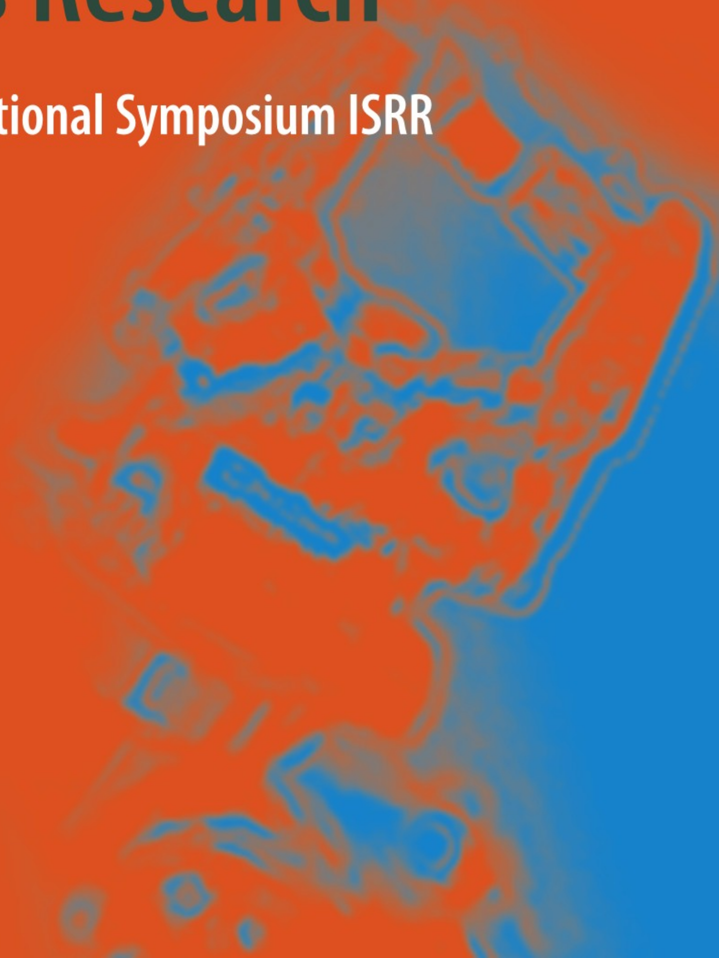
springer tracts in advanced robotics 70

Cédric Pradalier
Roland Siegwart
Gerhard Hirzinger (Eds.)

Robotics Research

The 14th International Symposium ISRR



Springer

Springer Tracts in Advanced Robotics

Volume 70

Editors: Bruno Siciliano · Oussama Khatib · Frans Groen

Cédric Pradalier, Roland Siegwart,
and Gerhard Hirzinger (Eds.)

Robotics Research

The 14th International Symposium ISRR

Springer

Editors

Dr. Cédric Pradalier
ETH Zürich
Institute of Robotics and Intelligent Systems
Autonomous Systems Lab
Tannenstrasse 3
8092 Zürich
Switzerland
E-mail: cedric.pradalier@mavt.ethz.ch

Prof. Dr. Roland Siegwart
ETH Zürich
Institute of Robotics and Intelligent Systems
Autonomous Systems Lab
Tannenstrasse 3
8092 Zürich
Switzerland
E-mail: rsiegwart@ethz.ch

Prof. Dr. Gerhard Hirzinger
German Aerospace Center (DLR)
Institute of Robotics and Mechatronics
Münchner Strasse 20
82234 Oberpfaffenhofen-Wessling
Germany
E-mail: gerd.hirzinger@dlr.de

ISBN 978-3-642-19456-6 e-ISBN 978-3-642-19457-3

DOI 10.1007/978-3-642-19457-3

Springer Tracts in Advanced Robotics ISSN 1610-7438

Library of Congress Control Number: 2011924480

Typeset & Cover Design: Scientific Publishing Services Pvt. Ltd., Chennai, India.

Printed on acid-free paper

5 4 3 2 1 0

springer.com

Editorial Advisory Board

STAR (Springer Tracts in Advanced Robotics) has been promoted under the auspices of EURON (European Robotics Research Network)

Foreword

By the dawn of the new millennium, robotics has undergone a major transformation in scope and dimensions. This expansion has been brought about by the maturity of the field and the advances in its related technologies. From a largely dominant industrial focus, robotics has been rapidly expanding into the challenges of the human world (human-centered and life-like robotics). The new generation of robots is expected to safely and dependably interact and work with humans in homes, workplaces, and communities providing support in services, entertainment, education, exploration, healthcare, manufacturing, and assistance.

Beyond its impact on physical robots, the body of knowledge that robotics has produced is revealing a much wider range of applications reaching across diverse research areas and scientific disciplines, such as: biomechanics, haptics, neurosciences, and virtual simulation, animation, surgery, and sensor networks among others. In return, the challenges of the new emerging areas are proving an abundant source of stimulation and insights for the field of robotics. It is indeed at the intersection of disciplines where the most striking advances happen.

The Springer Tracts in Advanced Robotics (STAR) is devoted to bringing to the research community the latest advances in the robotics field on the basis of their significance and quality. Through a wide and timely dissemination of critical research developments in robotics, our objective with this series is to promote more exchanges and collaborations among the researchers in the community and contribute to further advancements in this rapidly growing field.

As one of robotics pioneering symposia, the International Symposium on Robotics Research (ISRR) has established over the past two decades some of the field's most fundamental and lasting contributions. Since the launching of STAR, ISRR and several other thematic symposia in robotics find an important platform for closer links and extended reach within the robotics community.

This 14th edition of "Robotics Research," edited by Cédric Pradalier, Roland Siegwart and Gerd Hirzinger, brings a collection of a broad range of topics in robotics. The content of these contributions provides a wide coverage of the current state of robotics research: the advances and challenges in its theoretical foundation and technology basis, and the developments in its traditional and novel areas of applications.

In addition to the collection of papers presented in the diverse technical areas, the symposium hosted the *Blue Sky Session*, which was organized to highlight the challenges and new research directions in robotics research. The diversity, novelty, and span of the work reveal the field's increased maturity and its expanded scope. This 14th edition of ISRR culminates with this important reference on the current developments and new directions in the field of robotics – a true tribute to its contributors and organizers!

Stanford, California
October 2010

Oussama Khatib
STAR Editor

Preface

Cédric Pradalier, Roland Siegwart, and Gerd Hirzinger

This volume contains a collection of papers presented at the 14th International Symposium of Robotic Research (ISRR). ISRR is the biennial meeting of the International Foundation of Robotic Research (IFRR). It aims at covering the many facets of robotic research. The 14th ISRR took place from August 31st to September 3rd, 2009, in Lucerne, Switzerland.

Nearly 100 researchers from the major robotics research institutions around the world, representing all major research areas of robotics, attended the 14th ISRR. The technical program featured 48 regular papers, half of them invited, which were carefully selected to cover some of the most important and fast emerging research topics in robotics. The presentations of these papers were arranged in 12 thematic sessions, each of which was chaired by members of the Program Committee. Additionally, six invited presentation were given by Hugh Durrant-Whyte, Daniela Rus, Brad Nelson, Stefan Schaal, Sebastian Thrun and Makoto Kaneko during the Blue Sky session, hold on top of Mount Pilatus, a 2132-meter-high peak in the vicinity of Lucerne. These talks and the following discussions endeavored to identify and explore the challenges that robotic research will have to address in the coming years. As a tradition of the ISRR symposium, one evening was dedicated to an open video session, sponsored by Maxon Motors. Under the chairmanship of Oussama Khatib, the participants showed brief videos about their recent work and achievements, as well as amusing failures. The technical program of the 14th ISRR was complemented by a rich social program, which included a banquet on Mount Pilatus and a dinner on the shore of the lake of Lucerne with robotic demonstrations.

The scientific program was composed by the ISRR organizing committee with a strong support of the IFRR officers and reviewers. As for the previous symposia, ISRR 2009 followed up on the successful concept of a mixture of invited contributions and open submissions. Half of the 48 presentations were therefore invited contributions from outstanding researchers selected by the IFRR officers through a careful process, and half were chosen among the 66 submissions after peer review. This selection process resulted in a truly excellent technical program which, we believe, featured some of the very best of robotic research.

The final structure of this volume differs from that of the technical sessions during the conference. Out of the 48 total presentations, the 42 papers which were finally submitted for publication are organized in 8 sections that encompass the major research directions of robotics: Navigation, Control & Planning, Human-Robot Interaction, Manipulation and Humanoids, Learning, Mapping, Multi-Robot Systems, and Micro-Robotics. They represent an excellent snapshot of cutting edge research in robotics and outline future directions.

The symposium would not have been possible without the diligent work of a great number of people, including the various committee members and technical reviewers. Special thanks to Cornelia Della Casa, Eve Lasserre and Luciana Borsatti for assisting with the organization of the meetings and for running the registration desk. Ralf Kästner created the website and oversaw the submission process, which is gratefully acknowledged. Jérôme Maye wrote, adapted and performed the Robotic Song, which you can have the pleasure to read on the next pages. Oussama Khatib, President of IFRR, provided helpful advice all along and was once again a central driving force behind the scene. And finally, we thank all the participants of the 14th ISRR for making the symposium such an enjoyable and inspiring event, both from the technical and social point of view.

Dr. Cédric Pradalier
Prof. Dr. Roland Siegwart
Autonomous Systems Lab, ETH Zürich

Prof. Dr.-Ing. Gerd Hirzinger
DLR – Institute for Robotic und Mechatronics

The ISRR 2009 Committee

General Co-chairs

Roland Siegwart, ETH Zurich
Gerd Hirzinger, DLR Germany

Program Co-chairs

Tomomasa Sato (Asia/Oceania)
Henrik Christensen (Americas)
Raja Chatila (Europa/Africa)

Publication Chair

Cédric Pradalier

Web Master

Ralf Kästner

Conference Office

Cornelia Della Casa
Eve Lasserre
Luciana Borsatti

International Foundation of Robotics Research

President

Oussama Khatib

Executive Officers

Hirochika Inoue
Gerd Hirzinger
Takeo Kanade

Honorary Officers

Ruzena Bajcsy
Rod Brooks
George Giralt
Bernie Roth

Officers

Robert Bolles
Herman Bruyninckx
Raja Chatila
Henrik Christensen
Paolo Dario
Rudiger Dillmann
Shigeo Hirose
John Hollerbach
Ray Jarvis
Makoto Kaneko
Dan Koditschek
Tomomasa Sato
Yoshiaki Shirai
Roland Siegwart

The Robotics Song

Lyrics: Jérôme Maye
Music: Borrowed from “Yellow submarine”, The Beatles

Verse 1:

```
F      C        Bb      F
In the mountains of Lucerne
 Dm       Gm       Bb     C
There we meet to celebrate
F     C     Bb        F
Our dreams and our vision
Dm       Gm   Bb      C
For the world of tomorrow
```

Verse 2:

```
F       C       Bb     F
And we speak about robots
Dm        Gm        Bb      C
While we drink a glass or two,
F      C    Bb       F
What a duty, what a pain
Dm       Gm    Bb     C
For the beauty of science
```

Chorus:

```
F                     C
We all share our passion for robotics,
                                 F
love for robotics, love for robotics
F                     C
We all share our passion for robotics,
                                 F
love for robotics, love for robotics
```

Verse 3:

```
F       C    Bb               F
We all come from different lands
Dm             Gm  Bb          C
But we always find a common ground
F       C       Bb        F
And the piano starts to play...

Musical: C F C F
```

Chorus

Verse 4:

```
F      C    Bb      F
Our robots can navigate
Dm             Gm   Bb         C
In all kinds of unknown situations
F      C    Bb           F
They can learn new behaviors
Dm      Gm   Bb           C
Collaborate or even replicate.
```

Verse 5:

```
F        C   Bb               F
But one day, that's what we fear
Dm         Gm   Bb     C
They won't need us anymore
F     C    Bb           F
They will set up conferences
Dm          Gm   Bb        C
Where they will talk about us.
```

Chorus

Verse 6:

```
F        C     Bb            F
But of course this won't happen
Dm              Gm   Bb          C
We will keep for us all the pleasure
F     C      Bb          F
To gather every now and then
Dm       Gm   Bb       C
In some fancy destination.
```

Verse 7:

```
F          C     Bb         F
And we'll still enjoy the talks
Dm        Gm  Bb       C
Of some visionary fellows
F         C   Bb              F
And we'll wait for the banquet
Dm       Gm  Bb        C
Above all what we prefer
```

Chorus x2

Contents

Navigation

Control and Motion Planning

Human Robot Interaction

Robotic Manipulation and Humanoids

Learning

Mapping

Multi-Robot Systems

Micro-Robots

Navigation

Cédric Pradalier

Navigation is one of the core competencies of mobile robots. Navigating is usually understood as trying to go somewhere while keeping an estimate of its own location. This implicitly encompasses the planning of a path to the destination, the avoidance of obstacles on the way, and the control of the vehicle on the path. Furthermore, to know where it is – i.e. to localize – the robot often needs to build an online model of its environment, in effect solving the mapping problem.

- In "Reciprocal n-body Collision Avoidance", Van den Berg et al. address the control problem of multiple robots navigating while avoiding each other without communication. The key contribution is an algorithm based on velocity-obstacle providing guaranteed local collision-free motion for a large number of robots in a cluttered workspace.
- In "Unifying Geometric, Probabilistic and Potential Field Approaches to Multi-Robot Coverage Control", Schwager et al. also consider the case of multiple robots but focus on a coverage task. The importance of this work lies in the fact that it brings together various existing coverage algorithms in a common framework.
- In "Design and Development of an Optimal-Control-Based Framework for Trajectory Planning, Threat Assessment, and Semi-Autonomous Control of Passenger Vehicles and Hazard Avoidance Scenarios", Anderson et al. use an optimal control framework to optimize the path of a car in the context of a driving assistance scenario. An essential aspect of this work is the intelligent merging of control coming from the autonomous system on one side and a human driver on the other side.
- In "Autonomy through SLAM for an Underwater Robot", Folkesson and Leonard provide an exhaustive view of the navigation capabilities of an underwater robot equipped with a sonar system. This chapter provides an insight on the importance of field testing for achieving robust navigational autonomy.
- In "Sensing and Control on the Sphere ", Corke and Mahony, propose an overview of the mathematics of spherical geometry applicable to robotics, with a specific focus on sensor fusion, control, and structure from motion applications.
- In "Estimating Ego-Motion in Panoramic Image Sequences with Inertial Measurements", Schill et al. address a problem at the core of any navigation task for a flying robot, namely the ego-motion estimation. An im-

portant aspect of this work is the focus on robustness to real condition and experimental validation with real data.

Together, these papers show some of the key challenges of autonomous navigation for mobile robots in 2009: in particular, the focus on reliability and experimental validation through long duration experiments and real situations, and the management of a team of robots to achieve a joint task.

Reciprocal n-Body Collision Avoidance*

Jur van den Berg, Stephen J. Guy, Ming Lin, and Dinesh Manocha

Abstract. In this paper, we present a formal approach to *reciprocal n-body collision avoidance*, where multiple mobile robots need to avoid collisions with each other while moving in a common workspace. In our formulation, each robot acts fully independently, and does not communicate with other robots. Based on the definition of velocity obstacles [5], we derive sufficient conditions for collision-free motion by reducing the problem to solving a low-dimensional linear program. We test our approach on several dense and complex simulation scenarios involving thousands of robots and compute collision-free actions for all of them in only a few milliseconds. To the best of our knowledge, this method is the first that can guarantee local collision-free motion for a large number of robots in a cluttered workspace.

1 Introduction

Collision avoidance is a fundamental problem in robotics. The problem can generally be defined in the context of an autonomous mobile robot navigating in an environment with obstacles and/or other moving entities, where the robot employs a continuous cycle of sensing and acting. In each cycle, an action for the robot must be computed based on local observations of the environment, such that the robot stays free of collisions with the obstacles and the other moving entities, while making progress towards a goal.[1]

Jur van den Berg · Stephen J. Guy · Ming Lin · Dinesh Manocha
Department of Computer Science, University of North Carolina at Chapel Hill, USA
e-mail: {berg,sjguy,lin,dm}@cs.unc.edu

* This research is supported in part by ARO Contracts W911NF-04-1-0088, NSF award 0636208, DARPA/RDECOM Contracts N61339-04-C-0043 and WR91CRB-08-C-0137, Intel, and Microsoft.

[1] Note that the problem of (local) collision-avoidance differs from *motion planning*, where the global environment of the robot is considered to be known and a complete path towards a goal configuration is planned at once [18], and *collision detection*, which simply determines if two geometric objects intersect or not (see e.g. [17]).

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 3–19.
springerlink.com

The problem of collision avoidance has been well studied for one robot avoiding static or moving obstacles. In this paper, we address the more involved and less studied problem of *reciprocal n-body collision avoidance*, where collisions need to be avoided among multiple robots (or any decision-making entities). This problem has important applications in many areas in robotics, such as multi-robot navigation and coordination among swarms of robots [20]. It is also an key component in crowd simulation for computer graphics and VR [11, 21], modeling of non-player characters in AI, studying flocks of birds and fish in biology [23], and real-time (air) traffic control [16]. In this paper, we propose a fast and novel method that simultaneously determines actions for many (possibly thousands of) robots that each may have different objectives. The actions are computed for each robot independently, without communication among the robots or central coordination. Yet, we prove that our method guarantees collision-free motion for each of the robots.

We use a simplified robot model, where each robot is assumed to have a simple shape (circular or convex polygon) moving in a two-dimensional workspace. Furthermore, we assume that the robot is *holonomic*, i.e. it can move in any direction, such that the control input of each robot is simply given by a two-dimensional velocity vector. Also, we assume that each robot has *perfect* sensing, and is able to infer the exact shape, position and velocity of obstacles and other robots in the environment.

Main results. We present a rigorous approach for reciprocal n-body collision avoidance that provides a *sufficient* condition for each robot to be collision-free for at least a fixed amount of time into the future, only assuming that the other robots use the same collision-avoidance protocol. Our approach is *velocity-based*. That implies that each robot takes into account the observed velocity of other robots in order to avoid collisions with them, and also that the robot selects its own velocity from its *velocity space* in which certain regions are marked as 'forbidden' because of the presence of other robots. Our formulation, "optimal reciprocal collision avoidance", infers for each other robot a half-plane (in velocity-space) of velocities that are allowed to be selected in order to guarantee collision avoidance. The robot then selects its optimal velocity from the intersection of all permitted half-planes, which can be done efficiently using *linear programming*. Under certain conditions with densely packed robots, the resulting linear program may be infeasible, in which case we select the 'safest possible' velocity using a three-dimensional linear program.

We experimented with our approach on several complex simulation scenarios containing thousands of robots. As each robot is independent, we can fully *parallelize* the computation of the actions for each robot and report very fast real-time running times. Furthermore, our experiments show that our approach achieves convincing motions that are smooth and collision-free.

The rest of this paper is organized as follows. We start by discussing previous work in Section 2. In Section 3, we formally define the problem we address in this paper. We derive the half-plane of permitted velocities for optimal reciprocal collision avoidance of a robot with respect to another robot in Section 4, and show how this approach is used to navigate among multiple robots in Section 5. We report experimental results in Section 6 and conclude in Section 7.

2 Previous Work

The problem of collision avoidance has been extensively studied. Many approaches assume the observed obstacles to be static (i.e. non-moving) [2, 4, 6, 7, 13, 14, 24], and compute an immediate action for the robot that would avert collisions with the obstacle, in many cases taking into account the robot's kinematics and dynamics. If the obstacles are also moving, such approaches typically repeatedly "replan" based on new readings of the positions of the obstacles. This may work well if the obstacles move slower than the robot, but among fast obstacles (such as crossing a highway), the velocity of the obstacles need to be specifically taken into account. This problem is generally referred to as "asteroid avoidance", and approaches typically extrapolate the observed velocities in order to estimate the future positions of obstacles [8, 9, 12, 19, 22, 28].

The problem of collision avoidance becomes harder when the obstacles are not simply moving without considering their environment, but are also intelligent decision-making entities that try to avoid collisions as well. Simply considering them as moving obstacles may lead to *oscillations* if the other entity considers all other robots as moving obstacles as well [15, 26]. Therefore, the reactive nature of the other entities must be specifically taken into account in order to guarantee that collisions are avoided. Yet, the robot may not be able to communicate with other entities and may not know their intents. We call this problem *reciprocal collision avoidance*, and is the focus of this paper.

Velocity obstacles (VO) [5] have been a successful velocity-based approach to avoid collisions with moving obstacles; they provide a *sufficient* and *necessary* condition for a robot to select velocity that avoids collisions with an obstacle moving at a known velocity. This approach was extended for robot-robot collision avoidance with the definition of Reciprocal Velocity Obstacles (RVO) [10, 26], where both robots were assumed to select a velocity outside the RVO induced by the other robot. However, this formulation only guarantees collision-avoidance under specific conditions, and does not provide a sufficient condition for collision-avoidance in general.[2] In this paper, we present the principle of *optimal reciprocal collision avoidance* (ORCA) that overcomes this limitation; ORCA provides a sufficient condition for multiple robots to avoid collisions among one another, and thus can guarantee collision-free navigation.

We note that it is possible to provide a sufficient and necessary condition for multiple (say n) robots to select collision-avoiding velocities, by defining a *composite* velocity obstacle in the $2n$-dimensional space of velocities of all n robots [1]. However, this is not only computationally impractical, it also requires central coordination among robots. This is incompatible with the type of distributed multi-robot navigation we focus on in this paper, in which each robot independently and simultaneously selects its velocity from its own 2-D velocity space.

[2] In fact, both robots selecting a velocity *inside* each other's RVO is a sufficient condition to end up in a collision.

3 Problem Definition

The problem we discuss in this paper is formally defined as follows. Let there be a set of n robots sharing an environment. For simplicity we assume the robots are disc-shaped and move in the plane $\mathbb{R}^2$ (the definitions and algorithms we present in this paper can easily be extended to translating polygons, and also to higher dimensions). Each robot A has a current position $\mathbf{p}_A$ (the center of its disc), a current velocity $\mathbf{v}_A$ and a radius r_A. These parameters are part of the robot's external state, i.e. we assume that they can be observed by other robots. Furthermore, each robot has a maximum speed $v_A^{\max}$ and a preferred velocity $\mathbf{v}_A^{\text{pref}}$, which is the velocity the robot would assume had no other robots been in its way (for instance a velocity directed towards the robot's goal with a magnitude equal to the robot's preferred speed). We consider these parameters part of the internal state of the robot, and can therefore not be observed by other robots.

The task is for each robot A to *independently* (and simultaneously) select a new velocity $\mathbf{v}_A^{\text{new}}$ for itself such that *all* robots are *guaranteed* to be collision-free for at least a preset amount of time τ when they would continue to move at their new velocity. As a secondary objective, the robots should select their new velocity as close as possible to their preferred velocity. The robots are not allowed to communicate with each other, and can only use observations of the other robot's *current* position and velocity. However, each of the robots may assume that the other robots use the same strategy as itself to select a new velocity.

We name this problem "reciprocal n-body collision avoidance". Note that this problem cannot be solved using central coordination, as the preferred velocity of each robot is only known to the robot itself. In Section 4, we present a *sufficient* condition for each robot to select a velocity that is collision-free for (at least) τ time. In Section 5 we show how we use this principle in a continuous cycle for multi-robot navigation.

4 Reciprocal Collision Avoidance

4.1 Preliminaries

For two robots A and B, the *velocity obstacle* $VO_{A|B}^{\tau}$ (read: the velocity obstacle for A induced by B for time window τ) is the set of all *relative* velocities of A with respect to B that will result in a collision between A and B at some moment before time τ [5]. It is formally defined as follows. Let $D(\mathbf{p}, r)$ denote an open disc of radius r centered at $\mathbf{p}$;

$$D(\mathbf{p}, r) = \{\mathbf{q} \,|\, \|\mathbf{q} - \mathbf{p}\| < r\}, \tag{1}$$

then:

$$VO_{A|B}^{\tau} = \{\mathbf{v} \,|\, \exists t \in [0, \tau] :: t\mathbf{v} \in D(\mathbf{p}_B - \mathbf{p}_A, r_A + r_B)\} \tag{2}$$

The geometric interpretation of velocity obstacles is shown in Fig. 1(b). Note that $VO_{A|B}^{\tau}$ and $VO_{B|A}^{\tau}$ are *symmetric* in the origin.

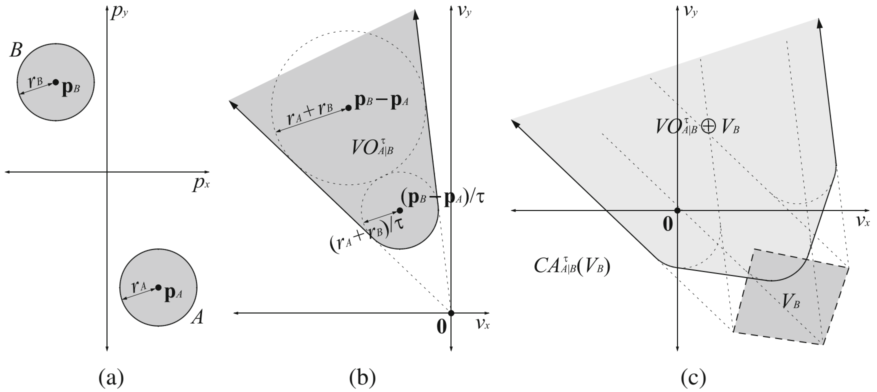


Fig. 1 **(a)** A configuration of two robots A and B. **(b)** The velocity obstacle $VO^{\tau}_{A|B}$ (gray) can geometrically be interpreted as a truncated cone with its apex at the origin (in velocity space) and its legs tangent to the disc of radius $r_A + r_B$ centered at $\mathbf{p}_B - \mathbf{p}_A$. The amount of truncation depends on the value of τ; the cone is truncated by an arc of a disc of radius $(r_A+r_B)/\tau$ centered at $(\mathbf{p}_B-\mathbf{p}_A)/\tau$. The velocity obstacle shown here is for $\tau = 2$. **(c)** The set of collision-avoiding velocities $CA^{\tau}_{A|B}(V_B)$ for robot A given that robot B selects its velocity from some set V_B (dark gray) is the complement of the Minkowski sum (light gray) of $VO^{\tau}_{A|B}$ and V_B.

Let $\mathbf{v}_A$ and $\mathbf{v}_B$ be current the velocities of robots A and B, respectively. The definition of the velocity obstacle implies that if $\mathbf{v}_A - \mathbf{v}_B \in VO^{\tau}_{A|B}$, or equivalently if $\mathbf{v}_B - \mathbf{v}_A \in VO^{\tau}_{B|A}$, then A and B will collide at some moment before time τ if they continue moving at their current velocity. Conversely, if $\mathbf{v}_A - \mathbf{v}_B \notin VO^{\tau}_{A|B}$, robot A and B are guaranteed to be collision-free for at least τ time.

More generally, let $X \oplus Y$ denote the Minkowski sum of sets X and Y;

$$X \oplus Y = \{\mathbf{x}+\mathbf{y} \,|\, \mathbf{x} \in X, \mathbf{y} \in Y\}, \tag{3}$$

then for any set V_B, if $\mathbf{v}_B \in V_B$ and $\mathbf{v}_A \notin VO^{\tau}_{A|B} \oplus V_B$, then A and B are guaranteed to be collision-free at their current velocities for at least τ time. This leads to the definition of the set of *collision-avoiding* velocities $CA^{\tau}_{A|B}(V_B)$ for A given that B selects its velocity from V_B (see Fig. 1(c)):

$$CA^{\tau}_{A|B}(V_B) = \{\mathbf{v} \,|\, \mathbf{v} \notin VO^{\tau}_{A|B} \oplus V_B\} \tag{4}$$

We call a pair of sets V_A and V_B of velocities for A and B *reciprocally collision-avoiding* if $V_A \subseteq CA^{\tau}_{A|B}(V_B)$ and $V_B \subseteq CA^{\tau}_{B|A}(V_A)$. If $V_A = CA^{\tau}_{A|B}(V_B)$ and $V_B = CA^{\tau}_{B|A}(V_A)$, we call V_A and V_B *reciprocally maximal*.

4.2 Optimal Reciprocal Collision Avoidance

Given the definitions above, we would like to choose sets of *permitted* velocities V_A for A and V_B for B such that $CA^{\tau}_{A|B}(V_B) = V_A$ and $CA^{\tau}_{B|A}(V_A) = V_B$, i.e. they are reciprocally collision-avoiding and maximal and guarantee that A and B are collision-free for at least τ time. Also, because A and B are individual robots, they should be able to infer their set of permitted velocities without communication with the other robot. There are infinitely many pairs of sets V_A and V_B that obey these requirements, but among those we select the pair that maximizes the amount of permitted velocities "close" to *optimization velocities* $\mathbf{v}^{\mathrm{opt}}_A$ for A and $\mathbf{v}^{\mathrm{opt}}_B$ for B.[3] We denote these sets $ORCA^{\tau}_{A|B}$ for A and $ORCA^{\tau}_{B|A}$ for B, and formally define them as follows. Let $|V|$ denote the measure (i.e. area in $\mathbb{R}^2$) of set V;

Definition 1 (Optimal Reciprocal Collision Avoidance). $ORCA^{\tau}_{A|B}$ and $ORCA^{\tau}_{B|A}$ are defined such that they are reciprocally collision-avoiding and maximal, i.e. $CA^{\tau}_{A|B}(ORCA^{\tau}_{B|A}) = ORCA^{\tau}_{A|B}$ and $CA^{\tau}_{B|A}(ORCA^{\tau}_{A|B}) = ORCA^{\tau}_{B|A}$, and such that for *all* other pairs of sets of reciprocally collision-avoiding velocities V_A and V_B (i.e. $V_A \subseteq CA^{\tau}_{A|B}(V_B)$ and $V_B \subseteq CA^{\tau}_{B|A}(V_A)$), and for *all* radii $r > 0$,

$$|ORCA^{\tau}_{A|B} \cap D(\mathbf{v}^{\mathrm{opt}}_A, r)| = |ORCA^{\tau}_{B|A} \cap D(\mathbf{v}^{\mathrm{opt}}_B, r)| \geq \min(|V_A \cap D(\mathbf{v}^{\mathrm{opt}}_A, r)|, |V_B \cap D(\mathbf{v}^{\mathrm{opt}}_B, r)|).$$

This means that $ORCA^{\tau}_{A|B}$ and $ORCA^{\tau}_{B|A}$ contain more velocities close to $\mathbf{v}^{\mathrm{opt}}_A$ and $\mathbf{v}^{\mathrm{opt}}_B$, respectively, than any other pair of sets of reciprocally collision-avoiding velocities. Also, the distribution of permitted velocities is "fair", as the amount of velocities close to the optimization velocity is equal for A and B.

We can geometrically construct $ORCA^{\tau}_{A|B}$ and $ORCA^{\tau}_{B|A}$ as follows. Let us assume that A and B adopt velocities $\mathbf{v}^{\mathrm{opt}}_A$ and $\mathbf{v}^{\mathrm{opt}}_B$, respectively, and let us assume that that causes A and B to be on collision course, i.e. $\mathbf{v}^{\mathrm{opt}}_A - \mathbf{v}^{\mathrm{opt}}_B \in VO^{\tau}_{A|B}$. Let $\mathbf{u}$ be the vector from $\mathbf{v}^{\mathrm{opt}}_A - \mathbf{v}^{\mathrm{opt}}_B$ to the closest point on the boundary of the velocity obstacle (see Fig. 2):

$$\mathbf{u} = \big(\operatorname*{arg\,min}_{\mathbf{v} \in \partial VO^{\tau}_{A|B}} \|\mathbf{v} - (\mathbf{v}^{\mathrm{opt}}_A - \mathbf{v}^{\mathrm{opt}}_B)\|\big) - (\mathbf{v}^{\mathrm{opt}}_A - \mathbf{v}^{\mathrm{opt}}_B), \tag{5}$$

and let $\mathbf{n}$ be the outward normal of the boundary of $VO^{\tau}_{A|B}$ at point $(\mathbf{v}^{\mathrm{opt}}_A - \mathbf{v}^{\mathrm{opt}}_B) + \mathbf{u}$. Then, $\mathbf{u}$ is the smallest change required to the relative velocity of A and B to avert collision within τ time. To "share the responsibility" of avoiding collisions among the robots in a fair way, robot A adapts its velocity by (at least) $\frac{1}{2}\mathbf{u}$ and assumes that B takes care of the other half. Hence, the set $ORCA^{\tau}_{A|B}$ of permitted velocities for A

[3] We introduce these optimization velocities to generalize the definition of ORCA. Nominally, the optimization velocities are equal to the current velocities, such that the robots have to deviate as little as possible from their current trajectories to avoid collisions. Other choices are discussed in detail in Section 5.2.

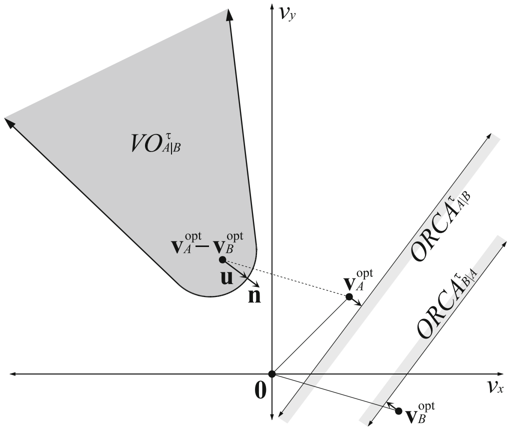

v_y
$VO^\tau_{A|B}$
$\mathbf{v}_A^{\text{opt}} - \mathbf{v}_B^{\text{opt}}$
$\mathbf{u}$
$\mathbf{n}$
$\mathbf{v}_A^{\text{opt}}$
$ORCA^\tau_{A|B}$
$ORCA^\tau_{B|A}$
$\mathbf{0}$
v_x
$\mathbf{v}_B^{\text{opt}}$

Fig. 2 The set $ORCA^\tau_{A|B}$ of permitted velocities for A for optimal reciprocal collision avoidance with B is a half-plane delimited by the line perpendicular to $\mathbf{u}$ through the point $\mathbf{v}_A^{\text{opt}} + \frac{1}{2}\mathbf{u}$, where $\mathbf{u}$ is the vector from $\mathbf{v}_A^{\text{opt}} - \mathbf{v}_B^{\text{opt}}$ to the closest point on the boundary of $VO^\tau_{A|B}$.

is the *half-plane* pointing in the direction of $\mathbf{n}$ starting at the point $\mathbf{v}_A^{\text{opt}} + \frac{1}{2}\mathbf{u}$. More formally:

$$ORCA^\tau_{A|B} = \{\mathbf{v} \,|\, (\mathbf{v} - (\mathbf{v}_A^{\text{opt}} + \frac{1}{2}\mathbf{u})) \cdot \mathbf{n} \geq 0\}. \tag{6}$$

The set $ORCA^\tau_{B|A}$ for B is defined symmetrically (see Fig. 2). The above equations also apply if A and B are *not* on a collision course when adopting their optimization velocities, i.e. $\mathbf{v}_A^{\text{opt}} - \mathbf{v}_B^{\text{opt}} \notin VO^\tau_{A|B}$. In this case, both robots each will take half of the responsibility to remain on a collision-free trajectory.

It can be seen that $ORCA^\tau_{A|B}$ and $ORCA^\tau_{B|A}$ as constructed above are in fact optimal according to the criterion of Definition 1. Agents A and B can infer $ORCA^\tau_{A|B}$ and $ORCA^\tau_{B|A}$, respectively, without communicating with each other, as long the robots can *observe* each other's position, radius, and optimization velocity. In Section 5.2, we discuss reasonable choices for the optimization velocity of the robots.

5 *n*-Body Collision Avoidance

In this section we show how to apply the ORCA principle as defined above to perform *n-body collision avoidance* among multiple moving robots, and discuss how we can incorporate static obstacles in this framework.

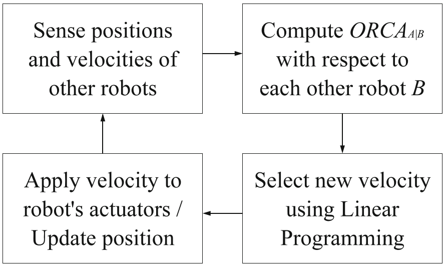

Sense positions and velocities of other robots
Compute $ORCA_{A|B}$ with respect to each other robot B
Select new velocity using Linear Programming
Apply velocity to robot's actuators / Update position

Fig. 3 A schematic overview of the continuous cycle of sensing and acting that is independently executed by each robot.

5.1 Basic Approach

The overall approach is as follows. Each robot A performs a continuous cycle of sensing and acting with time step Δt. In each iteration, the robot acquires the radius, the current position and the current optimization velocity of the other robots (and of itself). Based on this information, the robot infers the permitted half-plane of velocities $ORCA^{\tau}_{A|B}$ with respect to each other robot B. The set of velocities that are permitted for A with respect to *all* robots is the intersection of the half-planes of permitted velocities induced by each other robot, and we denote this set $ORCA^{\tau}_{A}$ (see Fig. 4):

$$ORCA^{\tau}_{A} = D(\mathbf{0}, v^{\max}_{A}) \cap \bigcap_{B \neq A} ORCA^{\tau}_{A|B}. \tag{7}$$

Note that this definition also includes the maximum speed constraint on robot A.

Next, the robot selects a new velocity $\mathbf{v}^{\text{new}}_{A}$ for itself that is closest to its preferred velocity $\mathbf{v}^{\text{pref}}_{A}$ amongst all velocities inside the region of permitted velocities:

$$\mathbf{v}^{\text{new}}_{A} = \operatorname*{arg\,min}_{\mathbf{v} \in ORCA^{\tau}_{A}} \|\mathbf{v} - \mathbf{v}^{\text{pref}}_{A}\|. \tag{8}$$

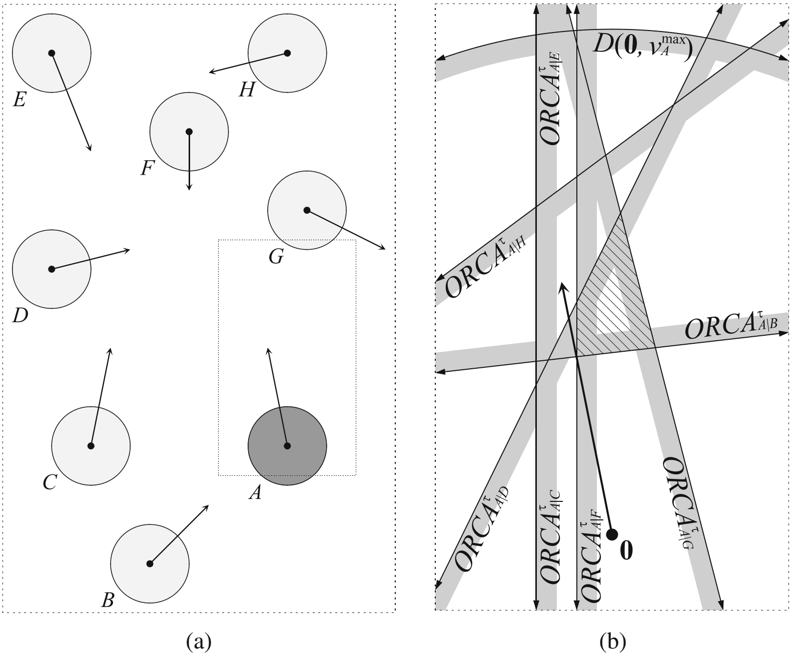
We will show below how to compute this velocity efficiently. Finally, the robot reaches its new position;

$$\mathbf{p}^{\text{new}}_{A} = \mathbf{p}_{A} + \mathbf{v}^{\text{new}}_{A} \Delta t, \tag{9}$$

and the sensing-acting cycle repeats (see Fig. 3).

The key step in the above procedure is to compute the new velocity $\mathbf{v}^{\text{new}}_{A}$ as defined by Equations (7) and (8). This can efficiently be done using *linear programming*, as $ORCA^{\tau}_{A}$ is a *convex* region bounded by linear constraints induced by the half-planes of permitted velocities with respect to each of the other robots (see Fig. 4). The optimization function is the distance to the preferred velocity $\mathbf{v}^{\text{pref}}_{A}$. Even though this is a quadratic optimization function, it does not invalidate the linear programming characteristics, as it has only one local minimum.

We use the efficient algorithm of [3], which adds the constraints one by one in random order while keeping track of the current optimal new velocity. The algorithm

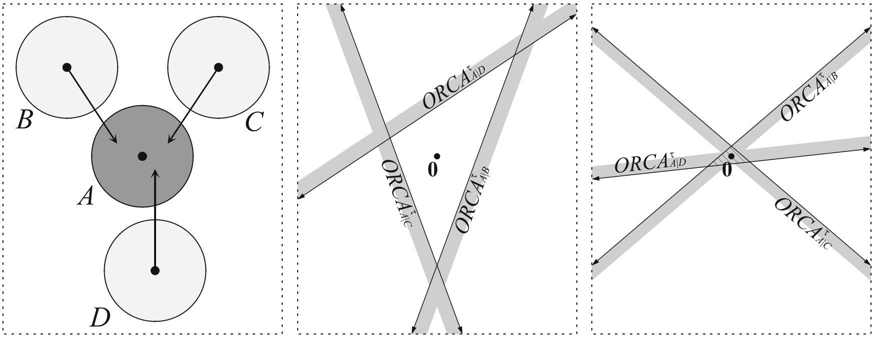


(a) (b)

Fig. 4 **(a)** A configuration with eight robots. Their current velocities are shown using arrows. **(b)** The half-planes of permitted velocities for robot A induced by each of the other robots with $\tau = 2$ and with $\mathbf{v}_*^{\text{opt}} = \mathbf{v}_*$ for all robots (i.e. the optimization velocity equals the current velocity). The half-planes of E and C coincide. The dashed region is $ORCA^{\tau}_{A}$, and contains the velocities for A that are permitted with respect to all other robots. The arrow indicates the current velocity of A.

has an expected running time of $O(n)$, where n is the total number of constraints in the linear program (which equals the number of robots in our case). The fact that we include a circular constraint for the maximum speed does not significantly alter the algorithm, and does not affect the running time. The algorithm returns the velocity in $ORCA^{\tau}_{A}$ that is closest to $\mathbf{v}_A^{\text{pref}}$, and reports failure if the linear program is infeasible, i.e. when $ORCA^{\tau}_{A} = \emptyset$. If the optimization velocities for the robots are chosen carefully (as we will discuss in Section 5.2), $ORCA^{\tau}_{A}$ will never be empty, and hence there will always be a solution that guarantees that the robots are collision-free for at least τ time.

We can increase the efficiency of selecting velocities by not including the constraints of all other robots, but only of those that are “close” by. In fact, robots B that are farther away from robot A than $(v_A^{\max} + v_B^{\max})\tau$ will never collide with robot A within τ time, so they can safely be left out of the linear program when computing the new velocity for robot A. A minor issue is that robot A does not know the maximum speed of other robots, but this can be worked around by “guessing” the maximum speed of other robots to be equal A’s own. We can efficiently find the set of close-by robots whose constraints should be included using a *kD-tree*.



(a) (b) (c)

Fig. 5 **(a)** A dense configuration with three robots moving towards robot A. The current velocities of the robots are shown using arrows; robot A has zero velocity. **(b)** The half-planes of permitted velocities for robot A induced by each of the other robots with $\tau = 2$ and $\mathbf{v}^{\text{opt}}_{*} = \mathbf{v}_{*}$ for all robots. The region $ORCA^{\tau}_{A}$ is empty, so avoiding collisions within τ time cannot be guaranteed. **(c)** The half-planes of permitted velocities for robot A induced by each of the other robots with $\tau = 2$ and $\mathbf{v}^{\text{opt}}_{*} = \mathbf{0}$ for all robots. The dashed region is $ORCA^{\tau}_{A}$.

5.2 Choosing the Optimization Velocity

One issue that we have left open is how to choose $\mathbf{v}^{\text{opt}}_{A}$ for each robot A. In order for the robots to infer the half-planes without communication, $\mathbf{v}^{\text{opt}}_{A}$ must be observable to other robots. Here, we discuss some reasonable possibilities:

- $\mathbf{v}^{\text{opt}}_{A} = \mathbf{0}$ for all robots A. If we set the optimization velocity to zero for all robots, it is *guaranteed* that $ORCA^{\tau}_{A}$ is non-empty for all robots A (see Fig. 5(c)). Hence, the linear programming algorithm as described above will find a velocity for all robots that guarantees them to be collision-free for at least τ time. This can be seen as follows. For any other robot B, the point $\mathbf{0}$ always lies outside the velocity obstacle $VO^{\tau}_{A|B}$ (for finite τ). Hence the half-plane $ORCA^{\tau}_{A|B}$ always includes at least velocity $\mathbf{0}$. In fact, the line delimiting $ORCA^{\tau}_{A|B}$ is perpendicular to the line connecting the current positions of A and B.
  A drawback of setting the optimization velocity to zero is that the behavior of the robots is unconvincing, as they only take into account the current positions of the other robots, and not their current velocities. In densely packed conditions, this may also lead to a global deadlock, as the chosen velocities for the robots converge to zero when the robots are very close to one another.
- $\mathbf{v}^{\text{opt}}_{A} = \mathbf{v}^{\text{pref}}_{A}$ (i.e. the preferred velocity) for all robots A. The preferred velocity is part of the internal state of the robots, so it cannot be observed by the other robots. Let us, for the sake of the discussion, assume that it is somehow possible to infer the preferred velocity of the other robots, and that the optimization velocity is set to the preferred velocity for all robots. This would work well in

low-density conditions, but, as the *magnitude* of the optimization velocity increases, it is increasingly more likely that the linear program is infeasible. As in most cases the preferred velocity has a constant (large) magnitude, regardless of the density conditions, this would lead to unsafe navigation in even medium density conditions.

- $\mathbf{v}_A^{\mathrm{opt}} = \mathbf{v}_A$ (i.e. the current velocity) for all robots A. Setting the optimization to the current velocity is the ideal trade-off between the above two choices, as the current velocity automatically adapts to the situation: it is (very) indicative of the preferred velocity in low-density cases, and is close to zero in dense scenarios. Also, the current velocity can be observed by the other robots. Nevertheless, the linear program may be infeasible in high-density conditions (see Fig. 5(b)). In this case, choosing a collision-free velocity cannot be guaranteed. Instead, we select the 'safest possible' velocity for the robot using a 3-D linear program (which we discuss in Section 5.3).

5.3 Densely Packed Conditions

If we choose $\mathbf{v}_A^{\mathrm{opt}} = \mathbf{v}_A$ for all robots A, there might not be a single velocity that satisfies all the constraints of the linear program in situations where the density of the robots is very high. In other words, the set $ORCA_A^{\tau}$ is empty (see Fig. 5(b)), and the algorithm of Section 5.1 returns that the linear program is infeasible. In this case, choosing a collision-free velocity cannot be guaranteed. Instead, we select the 'safest possible' velocity for the robot, i.e. the velocity that minimally 'penetrates' the constraints induced by the other robots. More formally, let $d_{A|B}(\mathbf{v})$ denote the *signed* (Euclidean) distance of velocity $\mathbf{v}$ to the edge of the half-plane $ORCA_{A|B}^{\tau}$. If $\mathbf{v} \in ORCA_{A|B}^{\tau}$, then $d_{A|B}(\mathbf{v})$ is negative. We now choose the velocity that minimizes the maximum distance to any of the half-planes induced by the other robots:

$$\mathbf{v}_A^{\mathrm{new}} = \operatorname*{arg\,min}_{\mathbf{v} \in D(\mathbf{0}, v_A^{\max})} \max_{B \neq A} d_{A|B}(\mathbf{v}). \tag{10}$$

Geometrically, this can be interpreted as moving the edges of the half-planes $ORCA_{A|B}^{\tau}$ perpendicularly outward with equal speed, until exactly one velocity becomes valid.

We can find this velocity using a *three-dimensional* linear program. For each other robot B, the distance function $d_{A|B}(\mathbf{v})$ is a plane in the three-dimensional $(\mathbf{v}, d)$ space. We now look for a point $(\mathbf{v}^*, d^*)$ that lies *above* all planes induced by the distance functions, and has a minimal d-value. Our new velocity $\mathbf{v}_A^{\mathrm{new}}$ is then set to $\mathbf{v}^*$.

We can use the same randomized algorithm as above to solve this 3-D linear program. It still runs in $O(n)$ expected time, where n is the number of other robots. In fact, we can project the problem down on the $\mathbf{v}$-plane, such that all geometric operations can be performed in 2-D. The 3-D linear program is always feasible, so it always returns a solution.

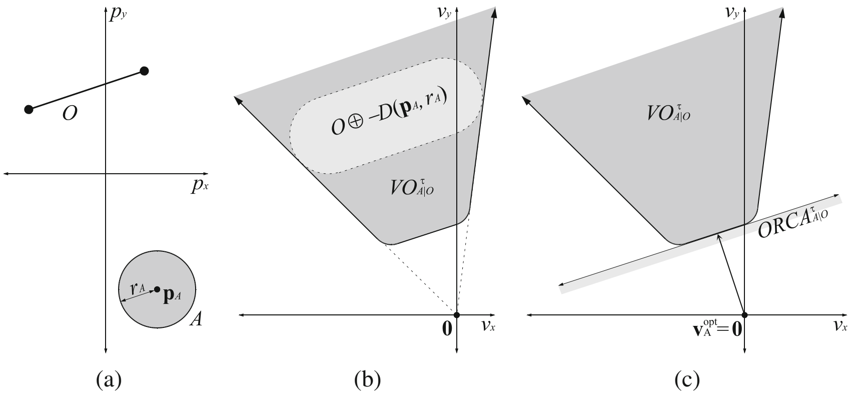


Fig. 6 **(a)** A configuration of a robot A and a line-segment obstacle O. **(b)** Geometric construction of the velocity obstacle $VO^{\tau}_{A|O}$ for $\tau = 2$. **(c)** The delimiting line of the half-plane $ORCA^{\tau}_{A|O}$ is tangent to $VO^{\tau}_{A|O}$ at the closest point on its boundary to $\mathbf{v}^{\text{opt}}_A$, which equals $\mathbf{0}$ in the case of obstacles.

Note that in these dense cases, the new velocity selected for the robot does not depend on the robot's preferred velocity. This means that the robot 'goes with the flow', and its behavior is fully determined by the robots surrounding the robot.

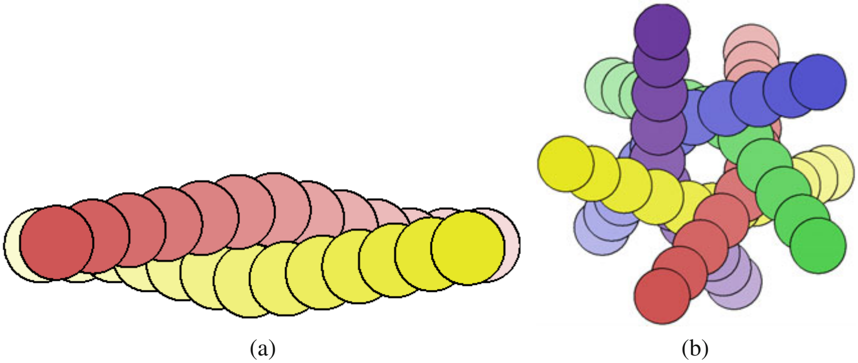
5.4 Static Obstacles

So far we have only discussed how robots avoid collisions with each other, but typical multi-robot environments also contain (static) obstacles. We can easily incorporate those in the above framework. We basically follow the same approach as above, with a key difference being that obstacles do not move, so the robots should take full responsibility of avoiding collisions with them.

We can generally assume that obstacles are modeled as a collection of line segments. Let O be one of such line segments, and let A be a robot with radius r_A positioned at $\mathbf{p}_A$. Then, the velocity obstacle $VO^{\tau}_{A|O}$ induced by the obstacle O is defined as (see Fig. 6(a) and (b)):

$$VO^{\tau}_{A|O} = \{\mathbf{v} \,|\, \exists t \in [0, \tau] :: t\mathbf{v} \in O \oplus -D(\mathbf{p}_A, r_A)\}. \tag{11}$$

Agent A will collide with obstacle O within τ time if its velocity $\mathbf{v}_A$ is inside $VO^{\tau}_{A|O}$, and it will be collision-free for at least τ time if its velocity is outside the velocity obstacle. Hence, we could define the region of permitted velocities for A with respect to O as the complement of $VO^{\tau}_{A|O}$. However, this would disallow us to use the efficient linear programming algorithm of Section 5.1, as the complement of $VO^{\tau}_{A|O}$ is a *non-convex* region. Therefore, we define the set of permitted velocities for A with respect to O as the half-plane $ORCA^{\tau}_{A|O}$ whose delimiting line is the *tangent* line to $VO^{\tau}_{A|O}$ at the closest point to $\mathbf{v}^{\text{opt}}_A$ on the boundary of $VO^{\tau}_{A|O}$ (see Fig. 6(c)).



(a) (b)

Fig. 7 Trace of robots in two small behavioral simulations. Robots are shown as colored disks which are light at their initial positions and darken as time progresses. **(a)** Trace of two simulated robots passing each other. **(b)** Trace of five simulated robots crossing each other to antipodal points in a circle.

In case of obstacles, we choose $\mathbf{v}_A^{\mathrm{opt}} = \mathbf{0}$ for all robots A. This guarantees that there always exists a valid velocity for the robot that avoids collisions with the obstacles within τ time. We can typically use a smaller value of τ with respect to obstacles than with respect to other robots, as robots should typically not be ‘shy’ to move towards an obstacle if this is necessary to avoid other robots. On the other hand, the constraints on the permitted velocities for the robot with respect to obstacles should be *hard*, as collisions with obstacles should be avoided at all cost. Therefore, when the linear program of Section 5.1 is infeasible due to a high density of robots, the constraints of the obstacles are not relaxed.

We note that the half-planes of permitted velocities with respect to obstacles as defined above only make sure that the robot avoids collisions with the obstacle; they do not make the robot move around them. The direction of motion around obstacles towards the robot’s goal should be reflected in the robot’s *preferred velocity*, which could be obtained using (efficient) global path planning techniques.

6 Experimental Results

To test our technique we ran several simulations. We performed both small-scale simulations to test local behavior and large-scale simulations to analyze performance scaling.

Behavioral Results. We first show two scenarios which highlight how robots smoothly avoid collisions with each other on the local level. In the first, shown in Fig. 7(a), two robots exchange position. When the robots notice that a collision is imminent (i.e. it will happen within τ time), they change velocities to smoothly avoid it. The second scenario shows five robots whose goal is to move across a circle to the antipodal position. As Fig. 7(b) shows, the robots smoothly spiral around each other to avoid collisions.

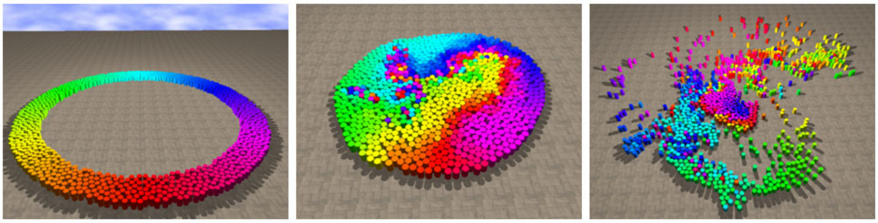


Fig. 8 Simulation of 1,000 agents trying to move through the center of a circle to antipodal positions. Robots smoothly move through the congestion that forms in the center.

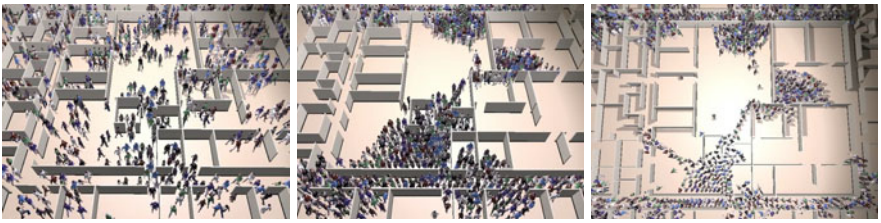
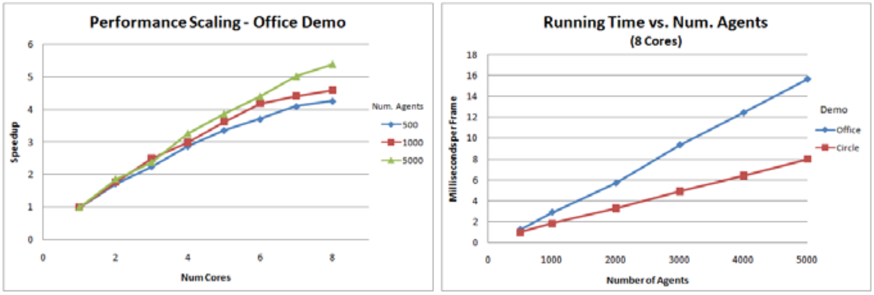


Fig. 9 Snapshots from simulation of 1,000 virtual agents evacuating an office as part of a crowd simulation.

Performance Results. In order to test the performance of our method we ran two large-scale simulations. The first test was a simulation of 1,000 agents in a large circle moving to antipodal positions as shown in Fig. 8. For the second test, shown in Fig. 9, we incorporated our optimal reciprocal collision avoidance formulation into the existing crowd simulation framework of [10]. In this simulation, virtual agents attempt to evacuate an office environment. The preferred velocity for each agent is set to follow a globally-planned path out of the office.

Because each agent makes independent decisions, we are able to efficiently parallelize the simulation by distributing the computations for agents across multiple processors. We used OpenMP multi-threading to parallelize key computations across eight Intel Xeon 2.66GHz (Clovertown) cores. Fig. 10(a) shows how our method scales across various numbers of cores in the Office scenario. There is a fairly good scaling in all scenarios – with best observed results in nearly linear scaling for a large number of agents where the constant system overhead becomes far less significant in the overall computation time.

In terms of absolute performance, Fig. 10(b) shows the running time for various numbers of agents for both simulations. For 5,000 agents on eight cores, it takes 8 ms (125 frames per second) to solve the collision-avoidance linear program for every agent in the large circle simulation, and 15.6 ms (64 frames per second) to update every agent in the office evacuation simulation.



(a) (b)

Fig. 10 Performance Graphs: **(a)** Performance scaling on the evacuation simulation for 1 to 8 cores. **(b)** Runtime for various number of agents on 8 cores (lower is better). Both simulations scale approximately linearly with the number of agents.

7 Conclusion and Future Work

In this paper, we have presented an efficient method that provides a sufficient condition for multiple robots to select an action that avoids collisions with other robots, though each acts independently without communication with others. Our approach to reciprocal *n*-body collision avoidance exhibits fast running times and smooth, convincing behavior in our experiments.

We have used a simple robot model, in which kinematics and dynamics are ignored. An important extension for future work is to take such constraints into account. We can either do this as a post-processing step, in which the computed new velocity is ‘clamped’ to what the kinematic and dynamic constraints allow. This would not strictly guarantee avoiding collisions anymore, but it may work well in practice [25]. A more thorough solution would be to take these factors intrinsically into account in the derivation of the permitted velocities for the robots. [27] and [19] provide some interesting ideas in this direction.

In this paper, we have demonstrated results for only 2-D environments. However, all definitions and the algorithm can be extended to 3-D. This may be interesting for applications such as autonomous aerial vehicles, or flocking simulation of birds or fish. Another important direction for future work is to implement the presented framework on real robots and incorporate sensing uncertainty. This has been done for reciprocal velocity obstacles in [25]. We believe that we can relatively easily replace the RVO formulation by our ORCA formulation in that implementation.

References

1. Abe, Y., Yoshiki, M.: Collision avoidance method for multiple autonomous mobile agents by implicit cooperation. In: IEEE RSJ Int. Conf. Intell. Robot. Syst., pp. 1207–1212 (2001)
2. Borenstein, J., Koren, Y.: The vector field histogram - fast obstacle avoidance for mobile robots. IEEE Journal of Robotics and Automation 7(3), 278–288 (1991)

3. de Berg, M., Cheong, O., van Kreveld, M., Overmars, M.: Computational Geometry: Algorithms and Applications. Springer, Heidelberg (2008)
4. Faverjon, B., Tournassoud, P.: A local based approach for path planning of manipulators with a high number of degrees of freedom. In: IEEE Int. Conf. Robot. Autom., pp. 1152–1159 (1987)
5. Fiorini, P., Shiller, Z.: Motion planning in dynamic environments using Velocity Obstacles. Int. Journal of Robotics Research 17(7), 760–772 (1998)
6. Fox, D., Burgard, W., Thrun, S.: The dynamic window approach to collision avoidance. IEEE Robot. Autom. Mag. 4, 23–33 (1997)
7. Fraichard, T., Asama, H.: Inevitable collision states - a step towards safer robots? Advanced Robotics 18(10), 1001–1024 (2004)
8. Fulgenzi, C., Spalanzani, A., Laugier, C.: Dynamic obstacle avoidance in uncertain environment combining PVOs and occupancy grid. In: IEEE Int. Conf. Robot. Autom., pp. 1610–1616 (2007)
9. Gil de Lamadrid, J.: Avoidance of Obstacles With Unknown Trajectories: Locally Optimal Paths and Periodic Sensor Readings. Int. Journal of Robotics Research 13(6), 496–507 (1994)
10. Guy, S., Chhugani, J., Kim, C., Satish, N., Dubey, P., Lin, M., Manocha, D.: Highly parallel collision avoidance for multi-agent simulation. In: ACM Symposium on Computer Animation (2009)
11. Helbing, D., Farkas, I., Vicsek, T.: Simulating dynamical features of escape panic. Nature 407, 487–490 (2000)
12. Hsu, D., Kindel, R., Latombe, J., Rock, S.: Randomized kinodynamic motion planning with moving obstacles. Int. J. Robot. Res. 21(3), 233–255 (2002)
13. Kanehiro, F., Lamiraux, F., Kanoun, O., Yoshida, E., Laumond, J.-P.: A local collision avoidance method for non-strictly convex polyhedra. In: Robotics: Science and Systems (2008)
14. Khatib, O.: Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. Int. Journal of Robotics Research 5(1), 90–98 (1986)
15. Kluge, B., Prassler, E.: Reflective navigation: Individual behaviors and group behaviors. In: IEEE Int. Conf. Robot. Autom., pp. 4172–4177 (2004)
16. Kuchar, J., Chang, L.: Survey of conflict detection and resolution modeling methods. In: AIAA Guidance, Navigation, and Control Conf. (1997)
17. Lin, M.: Efficient collision detection for animation and robotics. PhD thesis, University of California, Berkeley (1993)
18. LaValle, S.: Planning Algorithms. Cambridge University Press, Cambridge (2006)
19. Martinez-Gomez, L., Fraichard, T.: Collision avoidance in dynamic environments: an ICS-based solution and its comparative evaluation. In: IEEE Int. Conf. on Robotics and Automation (2009)
20. McLurkin, J., Demaine, E.: A Distributed Boundary Detection Algorithm for Multi-Robot Systems (2009) (under review)
21. Pettré, J., de Heras Ciechomski, P., Maïm, J., Yershin, B., Laumond, J.-P., Thalmann, D.: Real-time navigating crowds: scalable simulation and rendering. Computer Animation and Virtual Worlds 17, 445–455 (2006)
22. Petti, S., Fraichard, T.: Safe motion planning in dynamic environments. In: IEEE RSJ Int. Conf. Intell. Robot. Syst., pp. 2210–2215 (2005)
23. Reynolds, C.: Flocks, herds and schools: A distributed behavioral model. In: Int. Conf. on Computer Graphics and Interactive Techniques, pp. 25–34 (1987)
24. Simmons, R.: The curvature-velocity method for local obstacle avoidance. In: IEEE Int. Conf. on Robotics and Automation, pp. 3375–3382 (1996)

25. Snape, J., van den Berg, J., Guy, S., Manocha, D.: Independent navigation of multiple mobile robots with hybrid reciprocal velocity obstacles. In: IEEE/RSJ Int. Conf. Intell. Robot. Syst. (2009)
26. van den Berg, J., Lin, M., Manocha, D.: Reciprocal Velocity Obstacles for real-time multi-agent navigation. In: IEEE Int. Conf. on Robotics and Automation, pp. 1928–1935 (2008)
27. Wilkie, D., van den Berg, J., Manocha, D.: Generalized velocity obstacles. In: IEEE RSJ Int. Conf. Intell. Robot. Syst. (2009)
28. Zucker, M., Kuffner, J., Branicky, M.: Multipartite RRTs for rapid replanning in dynamic environments. In: IEEE Int. Conf. on Robotics and Automation, pp. 1603–1609 (2007)

Unifying Geometric, Probabilistic, and Potential Field Approaches to Multi-robot Coverage Control

Mac Schwager, Jean-Jacques Slotine, and Daniela Rus

Abstract. This paper unifies and extends several different existing strategies for multi-robot coverage control, including ones based on Voronoi partitions, probabilistic models, and artificial potential fields. We propose a cost function form for coverage problems that can be specialized to fit different distributed sensing and actuation scenarios. We show that controllers based on Voronoi partitions can be approximated arbitrarily well by simpler methods that do not require the computation of a Voronoi partition. The performance of three different controllers designed with the new approach is compared in simulation. We also formally delineate two classes of multi-agent problems: consensus problems and non-consensus problems. We show that coverage control is a non-consensus problem and that it requires the optimization of a nonconvex cost function.

1 Introduction

One of the fundamental problems of multi-robot control is how to deploy a group of robots to spread out over an environment to carry out sensing, surveillance, data collection, or distributed servicing tasks. This operation is called coverage control, and several methods have been proposed to accomplish it in a distributed and efficient way. In this paper we introduce a unifying principle that ties together a number of common ways of accomplishing coverage control. We show that many of the existing methods can be described as special instances of gradient descent on a cost

Mac Schwager · Daniela Rus
Computer Science and Artificial Intelligence Laboratory, MIT,
77 Massachusetts Avenue, Cambridge, MA 02139, USA
e-mail: schwager@mit.edu, rus@csail.mit.edu

Jean-Jacques Slotine
Nonlinear Systems Laboratory, MIT, 77 Massachusetts Avenue,
Cambridge, MA 02139, USA
e-mail: jjs@mit.edu

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 21–38.
springerlink.com

function. We propose a cost function form that specializes to give some common algorithms for coverage control, including Voronoi-based controllers, as in [7], controllers based on probabilistic models, as in [15], and artificial potential field-based controllers, as in [12].

Multi-robot coverage control serves as a prototype for many applications involving distributed sensing and distributed actuation. As examples of distributed sensing tasks, coverage control can be used to deploy underwater robots evenly over a coral reef to monitor coral health, or to deploy wheeled robots with cameras to spread out over a room for surveillance. Coverage control can also be used for multi-robot actuation tasks. For example, a coverage controller can be used to position oil clean-up robots over an oil spill so that they clean up the spill in minimum time. As another example, coverage control can be used to position de-mining robots to service a mine field in minimum time.

We describe a framework that is relevant to both sensing and actuation scenarios. We argue that Voronoi based methods are best suited to distributed *actuation* tasks, while a continuous approximation to the Voronoi decomposition is more appropriate for distributed *sensing* tasks. Furthermore, even in distributed actuation tasks, using a continuous approximation to the Voronoi cell improves the robustness and decreases the computational complexity of the controller. The continuous approximation is easy to compute regardless of the dimension of the space, while an exact Voronoi computation becomes unwieldy in higher dimensions.

As is typical for gradient based coverage control, the controllers we describe are provably convergent, robust to individual robot failures, and can adapt to environments that change slowly with respect to the speed of the robots. The controllers require that robots know the geometry of the environment and they know their own position in it using, for example, GPS or an indoor localization system. We also discuss how to accommodate the constraints of a communication network topology, but do not analyze this aspect of the problem in detail.

Related Work

Cortés et al. [7] introduced a controller for multi-robot coverage that works by continually driving the robots toward the centroids of their Voronoi cells. This inherently geometric strategy has seen many recent extensions to robots with a limited sensing radius in [6], to heterogeneous groups of robots and nonconvex environments in [18], and to incorporate learning of unknown environments in [21]. A recent text that presents much of this work in a cohesive fashion is [3] and an excellent overview is given in [16]. Coverage controllers also have been successfully implemented on robotic systems in [20, 19]. In this work we adopt notational conventions from the Voronoi based coverage control literature. Other common methods for coverage control take a probabilistic perspective. For example [15] proposes an algorithm for positioning robots to maximize the probability of detecting an event that occurs in the environment. Distributed dynamic vehicle routing scenarios are considered in [1, 17], in which events occur according to a random process and are serviced by the robot closest to them. Another common coverage control method is for robots to drive away from one another using artificial potential fields [12]. Despite the rather different models and objectives in these works, there are two common points which motivate us to find a unifying principle: 1) they all rely upon an optimization, and 2) they all use controllers that solve this optimization through the evolution of a dynamical system.

Some existing approaches do not fit under the framework we propose in this paper. A significant body of work has looked at coverage control as a motion planning problem. A survey of this work can be found in [5], and some significant contributions can be found in, for example, [4, 14] and the citations therein. Other authors have proposed information theoretic algorithms which consider placing sensors sequentially rather than driving them with a controller. Works such as [10, 13] position sensor nodes to maximize information for the sake of estimating a Gaussian random process in the environment.

Contributions

In the present work we focus on multi-agent deployment as an optimization problem. This is advantageous because it is amenable to geometric, probabilistic, and analytical interpretations, all of which have been seen in a separate light in the past. Our optimization approach ties together much of the existing literature on coverage control. Specifically, our contributions are: 1) We propose a cost function, putting particular emphasis on the role of a *mixing function*, a previously unrecognized component that captures critical assumptions about the coverage task. 2) We introduce a family of mixing functions with a free parameter, α, and show that different values of the parameter correspond to different assumptions about the coverage task, specifically showing that a minimum variance solution (i.e. a probabilistic strategy) is obtained with a parameter value of $\alpha = -1$, and Voronoi coverage (a geometric strategy) is recovered in the limit $\alpha \to -\infty$. 3) We prove a new result linking the convexity of a cost function to the multi-agent phenomenon of consensus. We show that coverage tasks are fundamentally different from consensus, and that they require the optimization of a nonconvex cost function. This suggests inherent limitations to gradient descent controller designs, which are pervasive in the coverage control literature.

The paper is organized as follows. In Section 2 we introduce the cost function, describing the purpose of each of its parts including the mixing function. We then produce a class of provably stable distributed coverage controllers by taking the gradient of the cost function. In Section 3 we derive three special cases of the controller; a Voronoi controller, a minimum variance controller, and a potential field controller. Section 4 presents our results on the relation between the convexity of a cost function, and multi-agent consensus. Simulation results are given in Section 5 and conclusions are in Section 6.

2 Generalized Coverage

In this section we introduce a general multi-agent cost function. We will use this cost function to define a new class of multi-agent coverage controllers by introducing a *mixing function*, which describes how information from different robots should be combined. We use the cost function to derive a stable gradient descent controller.

2.1 Coverage Cost Function

Let there be n robots[1], and let robot i have a position $p_i \in \mathscr{P} \subset \mathbb{R}^{d_p}$, where $\mathscr{P}$ is the state space of a robot, and d_p is the dimension of the space. The vector of all robot positions is denoted $P = [p_1^T, \dots, p_n^T]^T \in \mathscr{P}^n$, and we will call P the *configuration* of the robots. We want our robots to cover a bounded region $Q \subset \mathbb{R}^{d_q}$, which may or may not be related to the position space $\mathscr{P}$ of the robots. For example, the robots may be constrained to move in the space that they cover, so $\mathscr{P} = Q$ as in [7], or the robots may hover over a planar region that they cover with cameras, so $\mathscr{P} \subset \mathbb{R}^3$ and $Q \subset \mathbb{R}^2$, as in [19].

For each robot, a cost of sensing, or servicing, a point $q \in Q$ is given by a function $f(p_i, q)$. For simplicity of analysis we assume that $f(p_i, q)$ takes on only non-negative values, and that it is differentiable with respect to p_i.[2] The sensor measurements of the n robots are combined in a function $g(f(p_1, q), \dots, f(p_n, q))$, which we will call the *mixing function*. The mixing function embodies assumptions about the coverage task; that is, by changing the mixing function we can derive Voronoi based coverage control, probabilistic coverage control, and a variety of other kinds of distributed controllers.

Combining these elements, we propose to use a cost function of the form

$$\mathscr{H}(P) = \int_Q g(f(p_1, q), \dots, f(p_n, q))\phi(q)\, dq. \tag{1}$$

where $\phi : \mathbb{R}^{d_q} \mapsto \mathbb{R}_{>0}$ (we use the notation $\mathbb{R}_{>0}$ to mean the set of positive real numbers and $\mathbb{R}^d_{>0}$ the set of vectors whose components are all positive, and likewise for $\mathbb{R}_{\geq 0}$ and $\mathbb{R}^d_{\geq 0}$) is a weighting of importance over the region Q. Intuitively, the cost of the group of robots sensing at a single arbitrary point q is represented by the integrand $g(f(p_1, q), \dots, f(p_n, q))$. Integrating over all points in Q, weighted by their importance $\phi(q)$ gives the total cost of a configuration of the robots. We want to find controllers that stabilize the robots around configurations P^* that minimize $\mathscr{H}$. We will see in Section 4 that for coverage, and many other multi-agent problems, $\mathscr{H}$ is necessarily nonconvex, therefore gradient based controllers will yield *locally* optimal robot configurations. The cost function (1) will be shown to subsume

[1] We will use the term robot throughout, though the framework is suitable for general mobile sensing agents, including biological ones.

[2] This requirement can be generalized considerably as in [6] to the case where $f(p_i, q)$ is piece-wise continuous with a finite number of jump discontinuities. A finite sensor footprint can be modeled with a single jump discontinuity.

several different kinds of existing coverage cost functions. Drawing out the relations between these different coverage algorithms will suggest new insights into when one algorithm should be preferred over another.

2.2 Mixing Function

The mixing function $g_\alpha : \mathbb{R}^n_{\geq 0} \mapsto \mathbb{R}$ describes how information from different robots should be combined to give an aggregate cost of the robots sensing at a point q. This is shown graphically in Fig. 1 where the overlap of the two sensors is shown for illustrative purposes as the intersection of two circles. We propose a mixing function of the form

$$g_\alpha(f_1, \ldots, f_n) = \Big(\sum_{i=1}^{n} f_i^\alpha\Big)^{\frac{1}{\alpha}}, \tag{2}$$

with a free parameter $\alpha < 0$. The arguments $f_i \geq 0$ are real valued, and in our context they are given by evaluating the sensor function $f(p_i, q)$, hence the notation f_i. To be precise, the expression in (2) is undefined when $f_j = 0$ for some j, therefore in this case we define g_α by its limit, $g_\alpha(f_1, \ldots, 0, \ldots, f_n) = \lim_{f_j \to 0} \big(\sum_{i=1}^n f_i^\alpha\big)^{\frac{1}{\alpha}} = 0$.

This mixing function has several important properties. Firstly, notice that for $\alpha \geq 1$ it is the p-norm of the vector $[f_1 \cdots f_n]^T$. Specifically, it is convex for $\alpha \geq 1$ and as $\alpha \to \infty$, $g_\alpha(\cdot) \to \max_i(\cdot)$, which is the ℓ^∞ norm. However, we are interested in the regime where $\alpha < 0$. In this case $g_\alpha(\cdot)$ is not a norm because it violates the triangle inequality. In this regime it is also nonconvex, leading to a nonconvex cost function, which is a necessary attribute of coverage problems, as we will prove in Section 4.

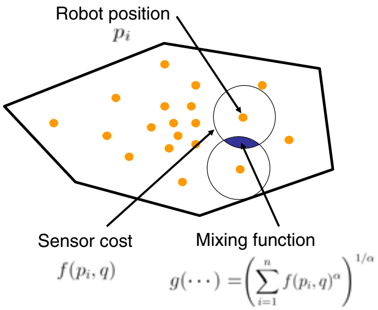
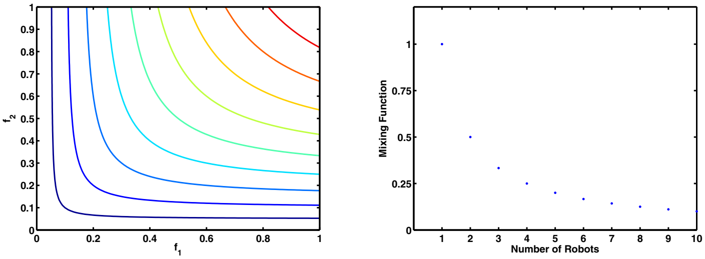

Robot position
p_i
Sensor cost
$f(p_i, q)$
Mixing function
$g(\cdots) = \left(\sum_{i=1}^{n} f(p_i, q)^\alpha\right)^{1/\alpha}$

Fig. 1 The mixing function is illustrated in this figure. The mixing function determines how information from the sensors of multiple robots is to be combined, shown graphically as the intersection of the two circles in the figure.

One can readily verify[3] that as $\alpha \to -\infty$, $g(\cdot) \to \min_i(\cdot)$. From an intuitive point of view, with $\alpha < 0$, $g_\alpha(\cdot)$ is smaller than any of its arguments alone. That is, the cost of sensing at a point q with robots at p_i and p_j is smaller than the cost of sensing with either one of the robots individually. Furthermore, the decrease in g_α from the addition of a second robot is greater than that from the addition of a third robot, and so on. There is a successively smaller benefit to adding more robots. This property is often called supermodularity, and has been exploited in a rather different way in [13]. Plots of level sets of $g_\alpha(f_1, f_2)$ for $\alpha = -1$ are shown in Fig. 2(a) and the decrease in $g_\alpha(\cdot)$ as the number of arguments grows is shown in Fig. 2(b). In this work we consider the number of robots to be fixed, but it is useful to illustrate the supermodularity property of the mixing function by considering the successive addition of new robots. Including this mixing function in the cost function from (1) gives

$$\mathscr{H}_\alpha = \int_Q \Big(\sum_{i=1}^{n} f(p_i, q)^\alpha \Big)^{\frac{1}{\alpha}} \phi(q)\, dq. \tag{3}$$

To model scenarios with a finite sensor footprint, we can also let $f(p_i, q)$ be infinite in some areas, in which case to keep the cost function bounded and differentiable it becomes necessary to include a prior w in the mixing function, yielding the variation $g_\alpha(f_1, \ldots, f_n) = \big(\sum_{i=1}^{n} f_i^\alpha + w^\alpha\big)^{1/\alpha}$. An application of this case was explored in [19] to design a controller for positioning multiple flying robots with downward facing cameras.



(a) Mixing Function Contours (b) Mixing Function Supermodularity

Fig. 2 The proposed mixing function with $\alpha = -1$ is shown in this figure. Fig. 2(a) shows a contour plot of the mixing function with two arguments. The nonlinear decrease in the function as more sensors are added, a property known as supermodularity, is shown in Fig. 2(b). The plot was generated with each robot fixed at 1 unit from the point q.

[3] We know $\lim_{\beta\to\infty}[\sum_i h_i^\beta]^{1/\beta} = \max_i h_i$. Write $\lim_{\alpha\to-\infty}[\sum_i f_i^\alpha]^{1/\alpha}$ as $\lim_{\beta\to\infty}[[\sum_i h_i^\beta]^{1/\beta}]^{-1}$ with $h_i = 1/f_i$ and $\beta = -\alpha$. We have $\lim_{\beta\to\infty}[[\sum_i h_i^\beta]^{1/\beta}]^{-1} = [\max_i h_i]^{-1} = [\frac{1}{\min_i f_i}]^{-1} = \min_i f_i$.

2.3 Gradient Control

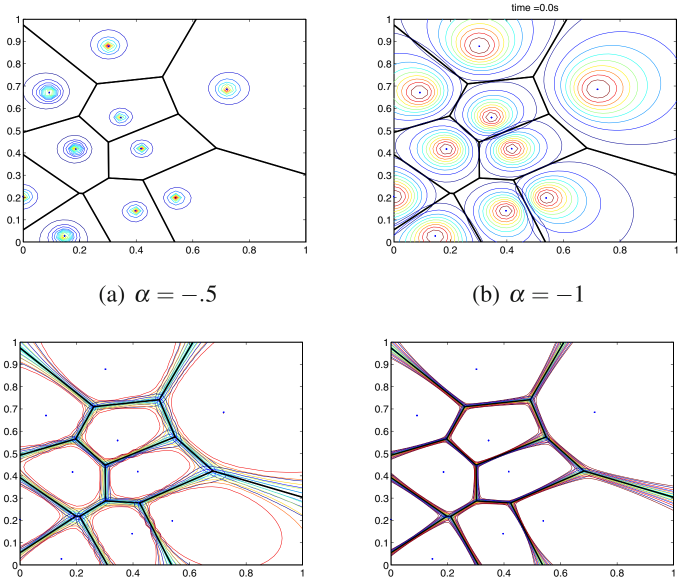
In order to derive a gradient descent controller, we take the derivative of the cost function $\mathscr{H}_\alpha$ with respect to the state of robot i to get

$$\frac{\partial \mathscr{H}_\alpha}{\partial p_i} = \int_Q \left(\frac{f(p_i,q)}{g_\alpha} \right)^{\alpha-1} \frac{\partial f(p_i,q)}{\partial p_i} \phi(q)\,dq.$$

To provide some intuition about the meaning of this function, notice that in the case that $f(p_i,q)$ is strictly increasing, the function inside the integral $(f(p_i,q)/g_\alpha)^{\alpha-1}$ gives an approximation to the indicator function[4] of the Voronoi cell of agent i, the approximation improving as $\alpha \to -\infty$. This is shown graphically in Fig. 3. It can be readily verified that this function is continuous, and that at $f(p_i,q) = 0$ it takes the value 1, and at $f(p_j,q) = 0$ and $j \neq i$ it takes the value 0.

For simplicity, we choose the function $f(p_i,q)$ to be

$$f(p_i,q) = \frac{1}{2}\|q - p_i\|^2, \quad \text{so that} \quad \frac{\partial f(p_i,q)}{\partial p_i} = -(q - p_i).$$



(a) $\alpha = -.5$

(b) $\alpha = -1$

(c) $\alpha = -5$

(d) $\alpha = -10$

Fig. 3 Contour plots of $(f(p_i,q)/g_\alpha)^{\alpha-1}$ are shown for a configuration of ten agent positions. The Voronoi tessellation is shown as well for comparison. As the parameter α approaches $-\infty$, $(f(p_i,q)/g_\alpha)^{\alpha-1}$ becomes closer to the indicator function of the Voronoi cell V_i.

[4] The indicator function for a set $S \subset Q$ returns 1 for $q \in S$, and 0 otherwise.

Other choices of $f(p_i,q)$ were investigated in [6] and could be used here as well, including functions with discrete jumps that model a finite senor footprint. This function represents the cost of a single robot i sensing at the position q. Therefore the quadratic form is appropriate for light based sensors, such as cameras or laser scanners. Light intensity drops off as the inverse square of the distance from the source, so it is reasonable for the cost to be proportional to the square of the distance. For tasks in which robots have to drive to a point q for servicing, and we want the cost to be proportional to the distance travelled, it would be more appropriate to use $f(p_i,q) = \|q - p_i\|$, for example.

We propose to use a gradient-based controller

$$\dot{p}_i = -k \frac{\partial \mathscr{H}_\alpha}{\partial p_i} = k \int_Q \left(\frac{f(p_i,q)}{g_\alpha} \right)^{\alpha-1} (q - p_i)\phi(q)\,dq, \tag{4}$$

where $k > 0$ is a positive control gain. We assume that the robots have integrator dynamics, $\dot{p}_i = u_i$, so we can control their velocity directly. We have found experimentally, in [20] for ground vehicles and [19] for quadrotor air vehicles, that this is a fair assumption as long as a fast inner control loop is in place to track the desired $\dot{p}_i$.

Stability can be proved by a direct application of the following result, the first part of which is stated in [11] Chapter 9, Section 4 as a corollary to Theorem 1.

Theorem 1 (Stability and Gradient Systems). *Let P^* be a critical point of $\mathscr{H}_\alpha$. Then P^* is a locally asymptotically stable equilibrium of the gradient system $\dot{P} = -\partial \mathscr{H}_\alpha / \partial P$ if and only if P^* is an isolated minimum of $\mathscr{H}_\alpha$.*

Corollary 1 (Coverage Control Stability). *For a group of robots with closed-loop dynamics*

$$\dot{p}_i = k \int_Q \left(\frac{f(p_i,q)}{g_\alpha} \right)^{\alpha-1} (q - p_i)\phi(q)\,dq$$

only configurations P^* for which $\mathscr{H}_\alpha(P^*)$ is an isolated local minimum are locally asymptotically stable.

Proof: The corollary follows directly from Theorem 1 and the fact that $\dot{P} = [\dot{p}_1^T \cdots \dot{p}_n^T]^T$ is the negative gradient of $k\mathscr{H}_\alpha$, which has the same critical points at $\mathscr{H}_\alpha$.

Remark 1 (Stability Vs. Convergence). Corollary 1 is somewhat different from typical stability results in the coverage control literature. It is more common to use LaSalle's invariance principle to prove convergence to a configuration where $\partial \mathscr{H}_\alpha / \partial P = 0$. The standard proof of this convergence applies also to the controller in (4). Unfortunately, such configurations are not necessarily local minima; they may be saddle points or local maxima. Our proof specifies that the system prefers local minima, not saddle points or maxima.

Remark 2 (Network Requirements). The computation of the controller requires that robot i knows the states of all the robots in the network. For this to be feasible there must either be a global supervisor or a fully connected network communication topology. It would be more useful if the controller depended only upon the states of robots with which it communicates. We suggest two methods to accomplish this, but we do not analyze them in detail in this paper. First, robot i can approximate its control law simply by computing (4) using only the states of the robots with which it is in communication. We expect this to give a good approximation because the function $(f(p_j,q)/g_\alpha)^{\alpha-1}$ depends weakly upon the states of agents that are not Voronoi neighbors, especially for small values of α, as evident from Fig. 3 . A rigorous stability analysis of this approximation scheme is difficult, however. A second option is for a robot i to use an estimated configuration vector, $\hat{P}$, in its calculation of the control law. The estimated configuration can be updated online using a standard distributed consensus algorithm (a so called "consensus estimator"). We expect that such a scheme may be amenable to a rigorous stability proof as its architecture is similar to adaptive control architectures. The investigation of these matters is left for future work.

3 Deriving Special Cases

In this section we show how the cost function 1 can be specialized to give three common kinds of coverage controllers, a Voronoi controller, which is geometric in nature, a minimum variance controller, which has a probabilistic interpretation, and a potential field controller. We conjecture that other coverage objectives beyond these three can be achieved with different choices of the mixing function parameter α.

Voronoi Coverage, $\alpha \to -\infty$

The Voronoi-based coverage controller described in [7] is based on a gradient descent of the cost function

$$\mathscr{H}_{\mathscr{V}} = \sum_{i=1}^{n} \int_{V_i} \frac{1}{2}\|q-p_i\|^2 \phi(q)\,dq,$$

where $V_i = \{q \in Q \mid \|q-p_i\| \leq \|q-p_j\|, \forall j \neq i\}$ is the Voronoi cell of robot i and the use of the subscript $\mathscr{V}$ is to distinguish it from $\mathscr{H}$ and $\mathscr{H}_\alpha$. The Voronoi partition can equivalently be written using the min function as

$$\mathscr{H}_{\mathscr{V}} = \int_Q \min_i(\frac{1}{2}\|q-p_i\|^2)\phi(q)\,dq,$$

because a point q is in the Voronoi cell V_i if and only if $\|q-p_j\|$ is minimized for $j=i$. As noted in Section 2.2, $\lim_{\alpha\to-\infty} g_\alpha(f_1,\ldots,f_n) = \min_i f_i$. Therefore $\mathscr{H}_{\mathscr{V}}$ is a special instance of (3) with the mixing function $g_{-\infty} = lim_{\alpha\to-\infty} g_\alpha$ and $f(p_i,q) = 1/2\|q-p_i\|^2$.

The choice of the min function for a mixing function now warrants some reflection. Consider a distributed actuation scenario in which we want to position robots so as to service an event that occurs randomly at some point in the environment q. Suppose any robot is equally capable of rendering the service, robots have to physically travel to the event to render the service, and our objective is to service an event as quickly as possible. Naturally, an event should be serviced by the robot that is closest to it, as it will reach the event the most quickly. In this case, the min function is the appropriate choice for a mixing function. By using the min function we are saying that the cost incurred by all the robots due to the event at q is the same as that incurred by the robot that is closest to q.

On the other hand, consider a sensing task in which an event of interest occurs randomly at a point q and is sensed at a distance by sensors located on the robots. In this case the use of the min function is more difficult to justify. Using the min function in this instance would imply that even though both p_i and p_j have some sensory information about the event, the cost function only counts the information from the one that is closest to q. This seems to be a poor choice of cost function for sensing, since in such cases we would want to capture the intuition that two sensors are better than one. The mixing function (2) captures this intuition. Furthermore, even in distributed actuation tasks, using a continuous approximation to the Voronoi cell improves the robustness of the controller. The discrete, geometric nature of the Voronoi computation combined with the continuous controller can lead to chattering, and small sensing errors can result in large changes in the control input. Fortunately, the Voronoi tessellation can be approximated arbitrarily well by choosing a small value of α, thereby preserving the Voronoi controller behavior while improving robustness.

Minimum Variance Coverage, $\alpha = -1$

We show in this section that setting the mixing function parameter to $\alpha = -1$ causes the robots to minimize the expected variance of their measurement of the location of a target of interest. As a side effect, we will formulate an optimal Bayesian estimator for the location of the target given the measurements of the agents.

Suppose our agents are equipped with sensors that give a noisy measurement of the position of a target in the environment. Let the target position be given by a random variable q that takes on values in Q, and agent i gives a measurement $y_i = q + w$, where $w \sim N(0, I_2\sqrt{f(p_i,q)})$ is a bi-variate normally distributed random variable, and where I_2 is the 2×2 identity matrix. The variance of the measurement, $f(p_i,q)$, is a function of the position of the sensor and the target. Intuitively one would expect a sensor to localize a target with more precision the closer the target is to the sensor. Then the measurement likelihood of agent i is $\mathbb{P}(y_i \mid q : p_i) = 1/(2\pi f(p_i,q))\exp\{-\|y_i - q\|^2/(2f(p_i,q))\}$, and the notation $\mathbb{P}(\cdot : p_i)$ is to emphasize that the distribution is a function of the agent position. Assume the measurements of different agents conditioned on the target position are independent. Also, let $\phi(q)$ be the prior distribution of the target's position. Then Bayes rule gives the posterior distribution,

$$\mathbb{P}(q \mid y_1,\ldots,y_n) = \frac{\prod_{i=1}^n \mathbb{P}(y_i \mid q : p_i)\phi(q)}{\int_Q \prod_{i=1}^n \mathbb{P}(y_i \mid q : p_i)\phi(q)\,dq}. \tag{5}$$

One can use the posterior to obtain a Bayesian estimate of the position of the event q given the measurements. For example, one may choose to estimate q using the mean, the median, or the maximum of the posterior in (5).

Our interest here, however, is not in estimating q. Instead we are interested in positioning the robots so that whatever estimate of q is obtained is the best possible one. To this end, we seek to position the robots to minimize the variance of their combined sensor measurements. The product of measurement likelihoods in the numerator of (5) can be simplified to a single likelihood function, which takes the form of an un-normalized Gaussian

$$\prod_{i=1}^n \mathbb{P}(y_i \mid q : p_i) = A \exp\left\{-\frac{\|\bar{y}-q\|^2}{2g_{-1}(\cdot)}\right\},$$

whose variance is equivalent to our mixing function $g_{-1}(\cdot) = \left(\sum_{i=1}^n f(p_i,q)^{-1}\right)^{-1}$. The values of A and $\bar{y}$ are not important in this context. If we want to position the robots so as to obtain the most decisive information from their sensors, we should move them to minimize this variance. Notice, however, that $g_{-1}(f(p_1,q),\ldots,f(p_n,q))$ is a random variable since it is a function of q. Taking the expectation over q of the likelihood variance gives our original cost function,

$$\mathscr{H}_{-1} = \mathbb{E}_q[g_{-1}(f(p_1,q),\ldots,f(p_n,q))] = \int_Q g_{-1}(f(p_1,q),\ldots,f(p_n,q))\phi(q)\,dq. \tag{6}$$

Thus we can interpret the coverage control optimization as finding the agent positions that minimize the expected variance of the likelihood function for an optimal Bayes estimator of the position of the target.

A more theoretically appealing criterion would be to position the agents to minimize the variance of the *posterior* distribution in (5). This gives a considerably more complicated cost function. The complication of this cost function and the fact that gradients can not be easily computed makes it a less practical option.

Potential Field Coverage, $\phi(q) = \sum_{i=1}^n \delta(\|q - p_i\|)$

The third type of coverage controller we consider is significantly different from the previous two in that it does not involve an integral over the environment. Instead it relies on the idea that robots should push away from one another to spread out over an environment, but should not move too far from one another or else they will become disconnected. Surprisingly, however, we will show that this rather different coverage philosophy can be reconciled with our generalized coverage cost function $\mathscr{H}$ in (1).

Let the importance function, $\phi(q)$, be given as a sum of delta-Dirac functions centered at each of the robot positions

$$\phi(q) = \sum_{i=1}^{n} \delta(\|q - p_i\|).$$

Substituting this for $\phi(q)$ in (1), the integral in $\mathscr{H}$ can then be evaluated analytically to give

$$\mathscr{H}_{\text{pot}} = \sum_{i=1}^{n} g(f(p_1, p_i), \ldots, f(p_n, p_i)),$$

and setting $g(f(p_1, p_i), \ldots, f(p_n, p_i)) = \sum_{j=1, j\neq i}^{n} f(p_j, p_i)$ gives a cost function for potential field based coverage (as well as models for flocking and herding)

$$\mathscr{H}_{\text{pot}} = \sum_{i=1}^{n} \sum_{j=1, j\neq 1}^{n} f(p_j, p_i), \tag{7}$$

where $f(p_j, p_i)$ can be interpreted as an inter-agent potential function. One choice for $f(p_j, p_i)$ is

$$f(p_j, p_i) = \frac{1}{6}\|p_j - p_i\|^{-2} - \|p_j - p_i\|^{-1}$$

which, taking the gradient of (7), yields the controller

$$\dot{p}_i = k \sum_{j=1, j\neq i}^{n} \left(\|p_j - p_i\|^{-2} - \frac{1}{3}\|p_j - p_i\|^{-3} \right) \frac{p_j - p_i}{\|p_j - p_i\|}. \tag{8}$$

Controllers similar to this one have been studied in a number of works, for example [12, 9, 22, 8]. There are numerous variations on this simple theme in the literature.

Computational Complexity

The gradient controllers described in this work must inevitably be discretized and implemented in a discrete time control loop. A common criticism of the Voronoi based coverage controller is that it is computationally intensive. At each iteration of the loop, a robot must re-compute its Voronoi cell. Additionally, the controller must compute spatial integrals over a region. In general, a discretized approximation must be used to compute the integral of $\phi(q)$ over the Voronoi cell, which is again computationally intensive. The two parameters that are important for computation time are the number of robots n and the number of grid squares in the integral computation, which we will call m. The typical decentralized algorithm for a single robot to compute its Voronoi cell (from [7]) runs in $O(n)$ time. The time complexity for computing a discretized integral is linear in the number of grid squares, and at each grid square requires a check if the center point is in the Voronoi cell, which is an $O(n)$ operation. Therefore the time complexity of the integral is in $O(nm)$. The Voronoi cell must be computed first, followed by the discretized integral, therefore the standard Voronoi controller has time complexity $O(n(m+1))$ at each step of the control loop.

Our controller in (4) does not require the computation of a Voronoi cell, but it does require the discretized spatial integral over the environment. We do not have to check if a point is in a polygon, but the integrand we evaluate, namely g_α is linear in n. Therefore the integral computation still has time complexity $O(nm)$, which is the time complexity of the controller at each step of the control loop. Yet as α decreases, the behavior of the controller approaches that of the Voronoi controller. The controller we propose in this paper is therefore significantly simpler in implementation (since it does not require the Voronoi computation), and it is faster computationally. Also, as the dimension of the state space increases, it becomes more difficult to compute and even to store the Voronoi decomposition, whereas the computation of g_α is straightforward in a space of any dimensionality.

4 Convexity and Consensus

Since we treat the multi-agent coverage problem as an optimization, it is natural to ask what sort of optimization we are dealing with, and what optimization tools can be brought to bear to solve it. We show in this section that the cost function in (3) is nonconvex, and that nonconvexity is a required feature of a large class of multi-agent problems, however undesirable this may be from an optimization perspective. Specifically, we demonstrate a link between the *convexity* of a cost function and the multi-agent phenomena known as *consensus*. For our purposes, consensus describes a multi-agent configuration in which all agents take on the same state, $p_1 = p_2 = \ldots = p_n$. Consensus is geometrically represented in the state space $\mathscr{P}^n$ as a d-dimensional hyperplane that passes through the origin (from the $d_p(n-1)$ independent equality constraints). This is illustrated by the diagonal line in Fig. 4 in a simplified 2D setting. We will prove, with some technical assumptions, that a multi-agent problem with a *convex* cost function admits at least one globally optimal consensus solution.

We begin with some basic definitions and facts from convex optimization which can be found in any standard text on the topic, for example [2]. A set $\Omega \subset \mathbb{R}^n$ is called convex if, for any two points in Ω, all points along the line segment joining them are also in Ω. An important consequence of the convexity of Ω is that any convex combination of points in Ω is also in Ω. A convex combination of m points $x_i \in \Omega$ is one of the form

$$x = \sum_{i=1}^{m} \alpha_i x_i \quad \text{where} \quad \sum_{i=1}^{m} \alpha_i = 1 \quad \text{and} \quad \alpha_i \geq 0 \quad \forall i.$$

A function $f : \Omega \mapsto \mathbb{R}$ is called convex if

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha) f(y) \quad \forall x, y \in \Omega \quad \text{and} \quad \forall \alpha \in [0,1].$$

This is equivalent to saying that the set of all points lying on or above the function $f(x)$ is a convex set (this set is known as the epigraph of $f(x)$). A function is called strictly convex if the '$\leq$' can be replaced with a '$<$' in the above relation. Also, we will use the word minimum to mean minimum or infimum if no minimum exists. We now state a theorem that follows from Weierstrass' Theorem and some well-known properties of convex functions.

Theorem 2 (Minima of Convex Functions). *For a continuous, convex function $f : \Omega \mapsto \mathbb{R}$, where the domain $\Omega \subset \mathbb{R}^n$ is convex, if any of the following are true:*

1. *Ω is bounded*
2. *There exists a scalar γ such that the level set $\{x \in \Omega \mid f(x) \leq \gamma\}$ is nonempty and bounded*
3. *f is such that $lim_{\|x\| \to \infty} f(x) = \infty$*

then the set of global minima of f is non-empty and convex.

We will apply this result to our multi-agent scenario. Consider a continuous multi-agent cost function $\mathscr{H} : \mathscr{P}^n \mapsto \mathbb{R}$. As before, an agent i has a state $p_i \in \mathscr{P} \subset \mathbb{R}^d$. It will be more convenient in this section to refer to a configuration of agents as a tuple $(p_1, \ldots, p_n) \in \mathscr{P}^n$, rather than the column vector notation used previously. Let us assume that agents are anonymous with respect to the cost function, by which we mean that the positions of any two agents can be interchanged without affecting the value of the cost function. This is formalized by the following assumption.

Assumption 1 (Anonymity of Agents) *The cost function $\mathscr{H}$ is such that*

$$\mathscr{H}(\ldots, p_i, \ldots, p_j, \ldots) = \mathscr{H}(\ldots, p_j, \ldots, p_i, \ldots) \quad \forall i, j \in \{1, \ldots, n\}.$$

Assumption 1 is in keeping with the ethos of complex, multi-agent systems, where the emphasis is on the global patterns that result from the interactions of many identical agents. Furthermore, let us assume that $\mathscr{H}$ and $\mathscr{P}^n$ satisfy at least one of the three properties in Theorem 2. Now we give the main result of this section.

Theorem 3 (Convexity and Consensus). *Under Assumption 1, if the cost function $\mathscr{H}(p_1, \ldots, p_n)$ is continuous and convex, $\mathscr{P}^n$ is convex, and one of the conditions in Theorem 2 is satisfied, then $\mathscr{H}(p_1, \ldots, p_n)$ has a global minimum such that $p_i = p_j$ $\forall i, j \in \{1, \ldots, n\}$.*

Proof: Our argument rests upon Assumption 1 and the fact from Theorem 2 that the set of minima of a convex function $\mathscr{H}$ is a convex set. Let h^* be the set of minima, and let $(\ldots, p_i^*, \ldots, p_j^*, \ldots)$ be an optimal solution in that set. By Assumption 1, $(\ldots, p_j^*, \ldots, p_i^*, \ldots)$ is also an optimal solution for any i and j. Therefore all permutations of components in $(p_1^*, \ldots, p_n^*)$ are optima. Then by convexity of h^*, all convex combinations of points in h^* are in h^*. In particular, the point $(\bar{p}, \ldots, \bar{p})$, where $\bar{p} = 1/n \sum_{i=1}^n p_i$ is an optimal solution (since it is a convex combination of permutations of $(p_1, \ldots, p_n)$). □

We show a geometric schematic of the proof argument in Fig. 4. The proof uses the fact that the convex set of minima must intersect the consensus hyperplane (the hyperplane where $p_i = p_j$ $\forall i, j$) at at least one point. A simple corollary follows.

Corollary 2 (Strict Convexity). *If the conditions of Theorem 3 are met and the cost function $\mathscr{H}(p_1, \ldots, p_n)$ is strictly convex, then the minimum is unique and is such that $p_i = p_j$ $\forall i, j \in \{1, \ldots, n\}$.*

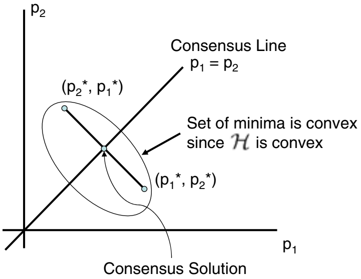


Fig. 4 This schematic shows the geometrical intuition behind the proof of Theorem 3 in a simplified 2D setting. Corollary 2 is proved by noticing that the set of minima is a single point (the consensus solution) if $\mathscr{H}$ is strictly convex.

Proof: A strictly convex function has at most one minimum over a convex domain. □

Remark 3 (Coverage is Nonconvex). Theorem 3 delineates two classes of multi-agent behaviors reminiscent of complexity classes in the theory of computation. One class, which we will call consensus behaviors, can be described as optimizing a convex cost function. The other class, which we will call non-consensus behaviors, is fundamentally different in that it can only be described with nonconvex cost functions. This is important because if we wish to design an optimization to solve a multi-agent problem, and we know that the problem cannot be solved satisfactorily by all the agents taking the same state, then we must use a nonconvex cost function. Likewise if we observe a multi-agent behavior in nature which cannot be described by all agents reaching the same state (the construction of a termite nest, for example), then an optimization-based explanation of this behavior must be nonconvex.

This is directly applicable to coverage problems. Indeed, coverage cannot be achieved with all agents moving to the same place, therefore coverage problems must involve the optimization of a nonconvex cost function. Our parameterized cost function $\mathscr{H}_\alpha$ from (3) is nonconvex for $\alpha < 1$ and is convex for $\alpha \geq 1$ because $\mathscr{H}_\alpha$ inherits the convexity properties of g_α. Correspondingly, we see that the robots become more highly aggregated as α approaches 1, and the robots all move to a common final position (i.e. consensus) when $\alpha \geq 1$. Theorem 3 explains why this is the case, and suggests that the search for a convex cost function for coverage control is futile.

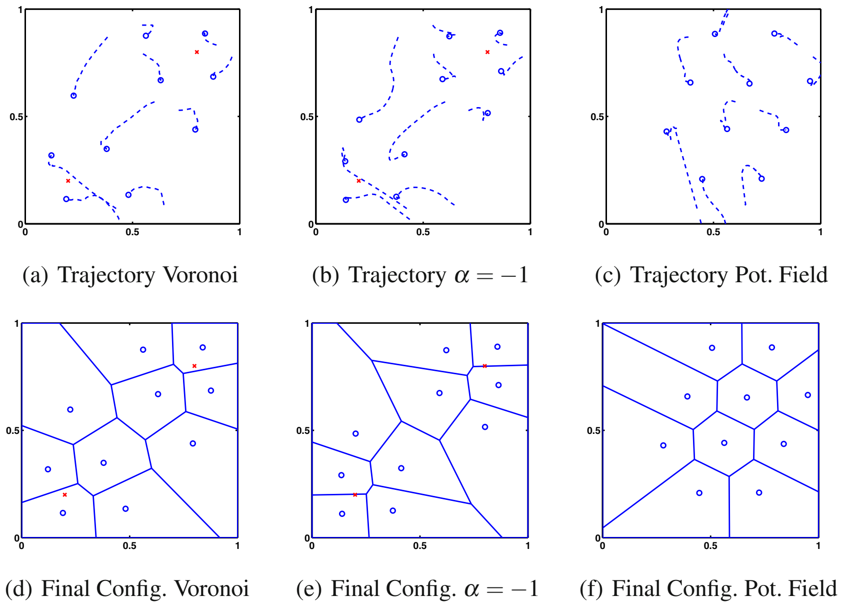
From an algorithmic point of view, however, this is unfortunate. Convex optimization has a powerful and well characterized tool set, but nonconvex optimization requires generally less efficient solution tools with looser guarantees. Distributed coverage controllers that use gradient methods (such as those in this paper) guarantee convergence to local optima, which is all one can expect for a nonconvex optimization. This points toward an open question for future work: are there nonconvex optimization methods not based on gradient descent that can be implemented in a multi-agent setting?

5 Simulation Results

The controller for the three scenarios described in Section 3 were simulated in a Matlab environment. The environment Q was taken to be a unit square, and the function $\phi(q)$ was set to be the sum of two Gaussian functions, one centered at $(.2,.2)$ and the other at $(.8,.8)$, both with variance .2. We expect to see a higher density of robots around areas of large $\phi(q)$. In our case, the robots group around the Gaussian centers.

The results of a simulation with ten robots using the Voronoi based controller, which corresponds to $\alpha \rightarrow -\infty$, is shown in Figs. 5(a) and 5(d). Similar plots are shown for the minimum variance controller, with $\alpha = -1$, in Figs. 5(b) and 5(e). Comparison of the two controllers shows that the Voronoi based controller causes the robots to spread out more, while as α increases, the robots group more closely together. When $\alpha > 1$, the cost function becomes convex, and the robots all move to the same position, which corroborates our results relating convexity to consensus (this is not shown in the plots).

The third scenario shown in Figs. 5(c) and 5(f) uses the potential field controller from [8]. This controller uses a sum of delta-Dirac functions for $\phi(q)$ rather than a sum of Gaussians, which causes the robots to arrange themselves in the close-packed lattice pattern.



(a) Trajectory Voronoi

(b) Trajectory $\alpha = -1$

(c) Trajectory Pot. Field

(d) Final Config. Voronoi

(e) Final Config. $\alpha = -1$

(f) Final Config. Pot. Field

Fig. 5 Trajectories and final configurations are shown for ten robots using the gradient control law with the Voronoi controller (5(a), 5(d)), the minimum variance controller (5(b), 5(e)), and a potential field controller (5(c), 5(f)). The Voronoi tessellation is shown for all scenarios for comparison, even though the right two controllers do not use the Voronoi cells for control.

6 Conclusion

In this paper we introduce a unifying optimization framework for multi-robot coverage control that brings together several different existing coverage algorithms. We point out that important properties of the underlying coverage objective are embodied in the way sensor information or actuator capabilities are combined from different robots. We propose a parameterized function to accomplish this combination, where different parameter values are shown to lead to different kinds coverage algorithms. Finally, we prove that for coverage problems the underlying optimization is necessarily nonconvex, making global optimization an unrealistic objective, especially for gradient descent controllers.

Our work invites an immediate extension, which is how to approximate the gradient controller over a communication graph. We outlined two methods for doing this. In the future the stability and robustness properties of these methods should be characterized and other methods should be investigated as well. Also our recognition that coverage problems stem from nonconvex optimizations suggests some new research directions. Gradient descent controllers, which are the most common type in the coverage control literature, can only be expected to find local minima. Therefore it is worthwhile to look for other nonconvex optimization methods that can be implemented in a multi-agent setting. We expect that these open questions will motivate new results for coverage control.

Acknowledgements. This work was done in the Distributed Robotics Laboratory at MIT. This work was supported in part by the MURI SWARMS project grant number W911NF-05-1-0219, NSF grant numbers IIS-0426838, CNS-0520305, CNS-0707601, EFRI-0735953, the MAST project, the ONR MURI SMARTS project, and The Boeing Company.

References

1. Arsie, A., Frazzoli, E.: Efficient routing of multiple vehicles with no explicit communications. International Journal of Robust and Nonlinear Control 18(2), 154–164 (2007)
2. Bertsekas, D., Nedić, A., Ozdaglar, A.E.: Convex Analysis and Optimization. Athena Scientific, Nashua (2003)
3. Bullo, F., Cortés, J., Martínez, S.: Distributed Control of Robotic Networks (2008) (manuscript preprint), Electronically, http://coordinationbook.info
4. Butler, Z.J., Rizzi, A.A., Hollis, R.L.: Complete distributed coverage of rectilinear environments. In: Proceedings of the Workshop on the Algorithmic Foundations of Robotics, Hanover, NH (March 2000)
5. Choset, H.: Coverage for robotics—A survey of recent results. Annals of Mathematics and Artificial Intelligence 31, 113–126 (2001)
6. Cortés, J., Martínez, S., Bullo, F.: Spatially-distributed coverage optimization and control with limited-range interactions. ESIAM: Control, Optimisation and Calculus of Variations 11, 691–719 (2005)
7. Cortés, J., Martínez, S., Karatas, T., Bullo, F.: Coverage control for mobile sensing networks. IEEE Transactions on Robotics and Automation 20(2), 243–255 (2004)

8. Dimarogonas, D.V., Kyriakopoulos, K.J.: Connectedness preserving distributed swarm aggregation for multiple kinematic robots. IEEE Transactions on Robotics 24(5), 1213–1223 (2008)
9. Gazi, V., Passino, K.M.: A class of repulsion/attraction forces for stable swarm aggregations. International Journal of Control 77(18), 1567–1579 (2004)
10. Hernandez, M.L., Kirubarajan, T., Bar-Shalom, Y.: Multisensor resource deployment using posterior Cramér-Rao bounds. IEEE Transactions on Aerospace and Electronic Systems 40(2), 399–416 (2004)
11. Hirsch, M.W., Smale, S.: Differential Equations, Dynamical Systems, and Linear Algebra. Academic Press, Inc., Orlando (1974)
12. Howard, A., Matarić, M.J., Sukhatme, G.S.: Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem. In: Proceedings of the 6th International Symposium on Distributed Autonomous Robotic Systems (DARS 2002), Fukuoka, Japan (June 2002)
13. Krause, A., Guestrin, C.: Near-optimal observation selection using submodular functions. In: Proceedings of 22nd Conference on Artificial Intelligence (AAAI), Vancouver, Canada (2007)
14. Latimer IV, D.T., Srinivasa, S., Shue, V.L., Sonneadnd, S., Choset, H., Hurst, A.: Towards sensor based coverage with robot teams. In: Proceedings of the IEEE International Conference on Robotics and Automation, vol. 1, pp. 961–967 (May 2002)
15. Li, W., Cassandras, C.G.: Distributed cooperative coverage control of sensor networks. In: Proceedings of the IEEE Conference on Decision ans Control and the European Control Conference, Seville, Spain (December 2005)
16. Martínez, S., Cortés, J., Bullo, F.: Motion coordination with distributed information. IEEE Control Systems Magazine 27(4), 75–88 (2007)
17. Pavone, M., Smith, S.L., Bullo, F., Frazzoli, E.: Dynamic multi-vehicle routing with multiple classes of demands. In: Proceedings of American Control Conference, St. Louis, Missouri (June 2009)
18. Pimenta, L.C.A., Kumar, V., Mesquita, R.C., Pereira, G.A.S.: Sensing and coverage for a network of heterogeneous robots. In: Proceedings of the IEEE Conference on Decision and Control, Cancun, Mexico (December 2008)
19. Schwager, M., Julian, B., Rus, D.: Optimal coverage for multiple hovering robots with downward-facing cameras. In: Proceedings of the International Conference on Robotics and Automation (ICRA 2009), Kobe, Japan (May 2009)
20. Schwager, M., McLurkin, J., Slotine, J.J.E., Rus, D.: From theory to practice: Distributed coverage control experiments with groups of robots. In: Proceedings of the International Symposium on Experimental Robotics (ISER 2008), Athens, Greece (July 2008)
21. Schwager, M., Rus, D., Slotine, J.J.: Decentralized, adaptive coverage control for networked robots. International Journal of Robotics Research 28(3), 357–375 (2009)
22. Tanner, H.G., Jadbabaie, A., Pappas, G.J.: Flocking in fixed and switching networks. IEEE Transactions on Automatic Control 52(5), 863–868 (2007)

Design and Development of an Optimal-Control-Based Framework for Trajectory Planning, Threat Assessment, and Semi-autonomous Control of Passenger Vehicles in Hazard Avoidance Scenarios

Sterling J. Anderson, Steven C. Peters, Tom E. Pilutti, and Karl Iagnemma*

Abstract. This paper describes the design of an optimal-control-based active safety framework that performs trajectory planning, threat assessment, and semi-autonomous control of passenger vehicles in hazard avoidance scenarios. This framework allows for multiple actuation modes, diverse trajectory-planning objectives, and varying levels of autonomy. A model predictive controller iteratively plans a best-case vehicle trajectory through a navigable corridor as a constrained optimal control problem. The framework then uses this trajectory to assess the threat posed to the vehicle and intervenes in proportion to this threat. This approach minimizes controller intervention while ensuring that the vehicle does not depart from a navigable corridor of travel. Simulation and experimental results are presented here to demonstrate the framework's ability to incorporate configurable intervention laws while sharing control with a human driver.

1 Introduction

Recent traffic safety reports from the National Highway Traffic and Safety Administration show that in 2007 alone, over 41,000 people were killed and another 2.5 million injured in motor vehicle accidents in the United States [1]. The longstanding presence of passive safety systems in motor vehicles, combined with the ever-increasing influence of active systems, has contributed to a decline in

Sterling J. Anderson · Steven C. Peters · Karl Iagnemma
Department of Mechanical Engineering, Massachusetts Institute of Technology,
77 Massachusetts Ave. Cambridge, MA 02139, USA
e-mail: {ster,scpeters,kdi}@mit.edu

Tom E. Pilutti
Ford Research Laboratories, Dearborn, MI 48124, USA
e-mail: tpilutti@ford.com

* Corresponding author.

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 39–54.
springerlink.com

these numbers from previous years. Still, the need for improved collision avoidance technologies remains significant.

Recent developments in onboard sensing, lane detection, obstacle recognition, and drive-by-wire capabilities have facilitated active safety systems that share steering and/or braking control with the driver [2,3]. These active safety systems operating with a “human in the loop” generally attempt to honor driver intentions, opposing them only when doing otherwise would lead to a collision or loss of control. Such modification of the driver’s intended trajectory requires that these systems assess the threat posed to a vehicle in order determine when and how strongly to intervene. Such systems should honor safe driver inputs and maneuvers while intervening when necessary to correct or override those deemed unsafe.

Among existing proposals for semi-autonomous vehicle navigation, lane-keeping systems using audible warnings [4], haptic alerts [5], steering torque overlays [6], and various combinations of these have been developed with mixed results [7]. In a recent subproject of the European PReVENT consortium, a lane-keeping system was designed to prevent lane departure by perceiving the environment, making heuristic-based trajectory planning decisions based on perceived threat, and implementing warning mechanisms or a slight steering torque overlay when the vehicle drifts from the desired trajectory [8].

Many of the navigation systems developed in previous work address only one piece of the active safety problem. While some use planning algorithms such as rapidly-exploring random trees [3], evolutionary programming [9] or potential fields analysis [10] to plan a safe vehicle path, others simply begin with this path presumed [11]. The threat posed by a particular path is seldom assessed by the controller itself and is often only estimated by a simple threat metric such as lateral or longitudinal acceleration required to avoid a road hazard [12]. Finally, hazard avoidance is commonly performed using one or more actuation methods (steering, differential braking, etc.) without explicitly accounting for the effect of driver inputs on the vehicle trajectory [8]. Such controllers selectively replace (rather than assist) the driver in performing the driving task.

Yu addressed this problem in mobility aids for the elderly by designing an adaptive shared controller which allocates control authority between the human user and a controller in proportion to the user’s performance [13]. These metrics and the associated intervention are designed to act on current and past user performance, however, and do not anticipate future states or performance. This reactive approach to semi-autonomy, while sufficient to control low-speed mobility aids, is not well suited for higher-speed applications with significant inertia effects and no pre-planned trajectory.

In this paper, a framework for passenger vehicle active safety is developed that performs vehicle trajectory planning, threat assessment, and hazard avoidance in a unified manner. This framework leverages the predictive and constraint-handling capabilities of Model Predictive Control (MPC) to plan trajectories through a preselected corridor, assess the threat this path poses to the vehicle, and regulate driver and controller inputs to maintain that threat below a given threshold.

The next section describes the semi-autonomous control framework and its associated trajectory prediction, control law, threat assessment, and intervention law. Simulation setup and results are then presented, followed by experimental setup and results, and the paper closes with general conclusions.

2 Framework Description

The framework described below leverages the predictive- and constraint-handling capabilities of MPC to perform trajectory planning, threat assessment, and hazard avoidance. First, an objective function is established to capture desirable performance characteristics of a safe or "optimal" vehicle path. Boundaries tracing the edges of the drivable road surface are assumed to have been derived from forward-looking sensor data and a higher-level corridor planner. These boundaries establish constraints on the vehicle's projected position. This constraint data, together with a model of the vehicle dynamics is then used to calculate an optimal sequence of inputs and the associated vehicle trajectory. The predicted trajectory is assumed to be a "best-case" scenario and used to establish the minimum threat posed to the vehicle given its current state and a series of best-case inputs. This threat is then used to calculate the necessary intervention required to prevent departure from the navigable region of travel and driver/controller inputs are scaled accordingly. Fig. 1 shows a block diagram of this system.

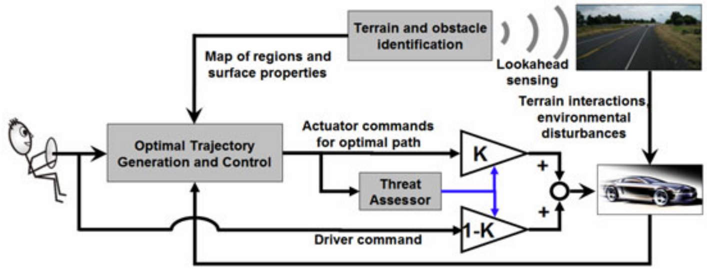

Terrain and obstacle identification
Map of regions and surface properties
Lookahead sensing
Terrain interactions, environmental disturbances
Actuator commands for optimal path
Optimal Trajectory Generation and Control
K
+
Threat Assessor
+
1-K
Driver command

Fig. 1 Diagram of an active safety system

In this paper it is assumed that road lane data is available and that road hazards have been detected, located, and mapped into a 2-dimensional corridor of travel. Existing systems and previous work in onboard sensing and sensor fusion justify this as a reasonable assumption [14]. Radar, LIDAR, and vision-based lane-recognition systems [3,15], along with various sensor fusion approaches [16] have been proposed to provide the lane, position, and environmental information needed by this framework.

Additionally, where multiple corridor options exist (such as cases where the roadway branches or the vehicle must circumnavigate an obstacle in the center of the lane), it is assumed that a high-level path planner has selected a single corridor through which the vehicle should travel.

2.1 Vehicle Path Planning

The best-case (or baseline) path through a given region of the state space is established by a model predictive controller. As described in later sections, metrics from this predicted path will be used to assess threat.

Model Predictive Control is a finite-horizon optimal control scheme that iteratively minimizes a performance objective defined for a forward-simulated plant model subject to performance and input constraints. Stated another way, MPC uses a model of the plant to predict future vehicle state evolution and optimize a set of inputs such that this prediction satisfies constraints and minimizes a user-defined objective function. At each time step, t, the current plant state is sampled and a cost-minimizing control sequence spanning from time t to the end of a control horizon of n sampling intervals, $t+n\Delta t$, is computed subject to inequality constraints. The first element in this input sequence is implemented at the current time and the process is repeated at subsequent time steps. The basic MPC problem setup is described in [17].

The vehicle model used in MPC accounts for the kinematics of a 4-wheeled vehicle, along with its lateral and yaw dynamics. Vehicle states include the position of its center of gravity [x, y], its yaw angle ψ, yaw rate $\dot{\psi}$, and sideslip angle β, as illustrated in Fig. 2. Table 1 defines and quantifies this model's parameters.

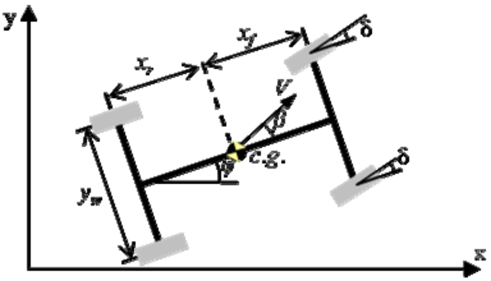


Fig. 2 Vehicle model used in MPC controller

Table 1 Vehicle model parameters

Symbol	Description	Value [units]
m	Total vehicle mass	2220 [kg]
I_{zz}	Yaw moment of inertia	3344 [kg·m^2]
x_f	C.g. distance to front wheels	1.43 [m]
x_r	C.g. distance to rear wheels	1.47 [m]
y_w	Track width	1.44 [m]
C_f	Front cornering stiffness	1433 [N/deg]
C_r	Rear cornering stiffness	1433 [N/deg]
μ	Surface friction coefficient	1

Tire compliance is included in the model by approximating lateral tire force (F_y) as the product of wheel cornering stiffness (C) and wheel sideslip (α or β for front or rear wheels respectively) as in

$$F_y = C\alpha \tag{1}$$

Linearized about a constant speed and assuming small slip angles, the equations of motion for this model are (where δ represents the steering angle input)

$$\dot{x} = V \tag{2}$$

$$\dot{y} = V(\psi + \beta) \tag{3}$$

$$\dot{\beta} = \frac{-(C_r + C_f)}{mV}\beta + \left(\frac{(C_r x_r - C_f x_f)}{mV^2} - 1\right)\dot{\psi} + \frac{C_f}{mV}\delta \tag{4}$$

$$\ddot{\psi} = \frac{(C_r x_r - C_f x_f)}{I_{zz}}\beta - \frac{(C_r x_r{}^2 + C_f x_f{}^2)}{I_{zz}V}\dot{\psi} + \frac{C_f x_f}{I_{zz}}\delta \tag{5}$$

where C_f and C_r represent the cornering stiffness of the lumped front wheels and the lumped rear wheels, and x_f and x_r are the longitudinal distances from the c.g. of the front and rear wheels, respectively.

2.1.1 Constraint Setup and Objective Function Description

As mentioned above, this framework assumes that the environment has been delineated previously. The boundaries of the navigable road surface at each timestep are then described by the constraint vectors

$$\begin{aligned} \mathbf{y}^{y}{}_{\max}(k) &= \left[y^{y}{}_{\max}(k+1) \quad \cdots \quad y^{y}{}_{\max}(k+p) \right]^T \\ \mathbf{y}^{y}{}_{\min}(k) &= \left[y^{y}{}_{\min}(k+1) \quad \cdots \quad y^{y}{}_{\min}(k+p) \right]^T \end{aligned} \tag{6}$$

In (6), $\mathbf{y}^{y}{}_{\max}$ and $\mathbf{y}^{y}{}_{\min}$ represent the upper and lower limits on the vehicle lateral position (y) and must satisfy

$$\mathbf{y}^{y}{}_{\max} - \mathbf{y}^{y}{}_{\min} > 0 \tag{7}$$

in order for the constraint space to remain feasible.

By enforcing vehicle position constraints at the boundaries of the navigable region of the road surface (i.e. the lane edges on an unobstructed road), the controller forces the MPC-generated path to remain within the constraint-bounded corridor whenever dynamically feasible. Coupling this lateral position constraint with input constraints $\mathbf{u}_{\text{min/max}}$, input rate constraints $\Delta\mathbf{u}_{\text{min/max}}$, and vehicle dynamic considerations, the corridor delineated by $\mathbf{y}^{y}{}_{\max}$ and $\mathbf{y}^{y}{}_{\min}$ translates to a safe operating region within the state space.

The controller's projected path through this constraint-imposed region is shaped by the performance objectives established in the MPC cost function. While many options exist for characterizing desirable vehicle trajectories, here, the total sideslip angle at the front wheels (α) was chosen as the trajectory characteristic to be minimized in the objective function. This choice was motivated by the strong influence front wheel sideslip has on the controllability of front-wheel-steered vehicles since cornering friction begins to decrease above critical slip angles. In [18] it is shown that limiting tire slip angle to avoid this strongly nonlinear (and possibly unstable) region of the tire force curve can significantly enhance vehicle stability and performance. Further, the linearized tire compliance model described here does not account for this decrease, motivating the suppression of front wheel slip angles to reduce controller-plant model mismatch. Finally, trajectories that minimize wheel slip also tend to minimize lateral acceleration and yaw rates, leading to a safer and more comfortable ride.

The MPC objective function with weighting matrices $R_{(\cdot)}$ then takes the form

$$J_k = \sum_{i=k+1}^{k+p} \frac{1}{2}\alpha_i^{\mathrm{T}} R_\alpha \alpha_i + \sum_{i=k}^{k+p-1} \frac{1}{2}\delta_i^{\mathrm{T}} R_\delta \delta_i + \sum_{i=k}^{k+p-1} \frac{1}{2}\Delta\delta_i^{\mathrm{T}} R_{\Delta\delta} \Delta\delta_i + \frac{1}{2}\rho_\varepsilon \varepsilon^2 \qquad (8)$$

where ε represents constraint violation and was included to soften select position constraints as $\mathbf{y}^j{}_{\min} - \varepsilon \mathbf{V}^j{}_{\min} \le \mathbf{y}^j \le \mathbf{y}^j{}_{\max} + \varepsilon \mathbf{V}^j{}_{\max}$.

In summary, the MPC controller uses vehicle position, input magnitude, and input rate constraints to satisfy safety requirements, while minimizing front wheel slip to maximize controllability.

2.2 Threat Assessment

The vehicle path calculated by the MPC controller is assumed to be the best-case or safest path through the environment. As such, key metrics from this prediction are used to assess instantaneous threat posed to the vehicle. By setting constraint violation weights (ρ_ε) significantly higher than the competing minimization weight (R_α) on front wheel sideslip, a hierarchy of objectives is created in order to force the optimal solutions to satisfy corridor constraints before minimizing front wheel sideslip. When constraints are not active, as illustrated by the gray vehicle in Fig. 3, front wheel sideslip – and the corresponding controllability threat – is minimized. When the solution is constrained, predicted front wheel sideslip increases with the severity of the maneuver required to remain within the navigable corridor.

The dark vehicle in Fig. 3 illustrates how the MPC-predicted optimal vehicle trajectory might appear as the tire slip angles and corresponding threat increase in the presence of an active constraint. As predicted sideslip approaches tire-cornering-friction-imposed limits, the threat of leaving the navigable corridor increases.

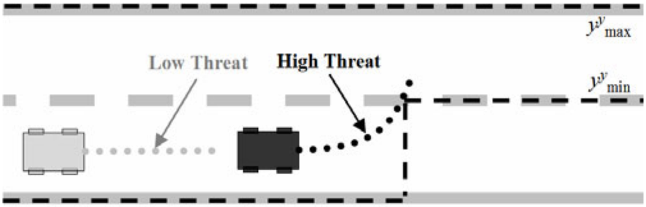


Fig. 3 Obstacle avoidance scenario showing MPC trajectory plans and corresponding threat

Various approaches are available to reduce the vector $\boldsymbol{\alpha}$ to a scalar threat metric Φ. In this work,

$$\Phi(k) = \max\left(\alpha_{k+1} \quad \alpha_{k+2} \quad \cdots \quad \alpha_{k+p}\right) \tag{9}$$

was chosen for its good empirical performance when used to regulate controller intervention (described in the next section).

2.3 Hazard Avoidance

Given a best-case vehicle path through the environment and a corresponding threat, desired inputs from the driver and controller are blended and applied to the vehicle. This blending is performed based on the threat assessment: a low predicted threat causes more of the driver's input and less of the controller's input to be applied to the vehicle, while high threat allows controller input to dominate that of the driver. This "scaled intervention" may thereby allow for a smooth transition in control authority from driver to controller as threat increases.

Denoting the current driver input by u_{dr} and the current controller input by u_{MPC}, the blended input seen by the vehicle, u_v, is defined as

$$u_v = K(\Phi)u_{MPC} + (1 - K(\Phi))u_{dr} \tag{10}$$

The intervention function K is used to translate predicted vehicle threat (Φ) into a scalar blending gain. This function is bounded by 0 and 1 and may be linear, piecewise-linear, or nonlinear. Linear and piecewise-linear forms of this function may be described by

$$K(\Phi) = \begin{cases} 0 & 0 \le \Phi \le \Phi_{eng} \\ \dfrac{\Phi_{aut} - \Phi}{\Phi_{aut} - \Phi_{eng}} & \Phi_{eng} \le \Phi \le \Phi_{aut} \\ 1 & \Phi \ge \Phi_{aut} \end{cases} \tag{11}$$

In (11), the shape of K is described by the threat level at which the semi-autonomous controller engages (Φ_{eng}) and the level at which it is given full control authority and effectively acts as an autonomous controller (Φ_{aut}).

Using predicted threat (Φ) as calculated in (9) with an appropriate cost function formulation of the form (8) ensures that 1) the threat metric regulating controller intervention is minimized in the path plan (and associated control calculation) and 2) the controller maintains full control authority when constraints are binding.

Increasing Φ_{eng} widens the "low threat" band in which the driver's inputs are unaffected by the controller. While this provides greater driver freedom for low-threat situations, this freedom comes at the cost of increasing the rate of controller intervention when Φ_{eng} is exceeded. This increased rate of intervention may adversely affect driver experience, as discussed in the results below.

Increasing the value of Φ_{aut}, on the other hand, delays complete controller intervention until more severe maneuvers are predicted. The friction-limited bounds on the linear region of the tire force curve (1) suggest a natural upper limit of $\Phi \leq 5$ degrees on surfaces with a friction coefficient of 1.0 in order to ensure that by the time the predicted maneuver required to remain within the safe region of the state space reaches this level of severity, the controller has full control authority and can – unless unforeseen constraints dictate otherwise – guide the vehicle to safety.

3 Simulation Setup

Controller performance was simulated using a vehicle plant model provided by researchers at Ford. This model describes longitudinal and lateral tire forces with the semi-empirical Pacejka tire model where the longitudinal and cornering forces are assumed to depend on the normal force, tire slip angle, surface friction, and longitudinal slip [19].

The vehicle model described by (2-5), with the parameters given in Table 1 was used in the receding horizon controller. Controller parameters are defined and quantified in Table 2 and vehicle velocity was chosen as 14 meters per second.

Table 2 Controller parameters

Symbol	Description	Value [units]
p	Prediction horizon	40
n	Control horizon	20
$R_y^{(\alpha)}$	Weight on front wheel slip	0.2657
R_u	Weight on steering input	0.01
$R_{\Delta u}$	Weight on steering input rate (Δ per Δt)	0.01
$u_{min/max}$	Steering input constraints	± 10 [deg]
$\Delta u_{min/max}$	Steering input rate (per Δt) constraints	± .75 [deg] (15 deg/s)
$y^y_{min/max}$	Lateral position constraints	Scenario-dependent
ρ_ε	Weight on constraint violation	1×10^5
$[\Phi_{eng}\ \Phi_{aut}]$	Thresholds for controller intervention	{[0 3],[1 3],[0 4],[2 4]}°
V	Variable constraint relaxation on vehicle position	[1.25, ···, 1.25, 0.01]

4 Simulation Results

Simulation results were obtained for various maneuvers, driver inputs, objective function configurations, and intervention laws. Results below are shown for double lane change maneuvers with a driver steer input $\delta_{driver} = 0$. This driver behavior was intended to simulate a drowsy or otherwise inattentive driver.

Semi-autonomous controllers using varying threat thresholds for controller engagement (Φ_{eng}) and full autonomy (Φ_{aut}) successfully satisfied safety constraints while allowing significant driver control whenever possible. In the figure legends below, sideslip thresholds Φ_{eng} and Φ_{aut} (in units of degrees) are labeled as [Φ_{eng} Φ_{aut}].

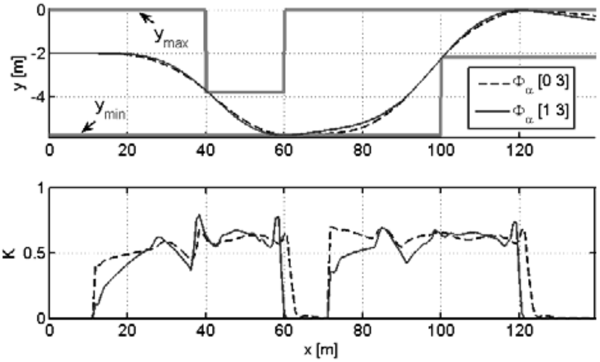


Fig. 4 Simulation results showing the effect of intervention thresholds ([Φ_{eng} Φ_{aut}] = [0 3], [1 3]) on semi-autonomous corridor-tracking performance

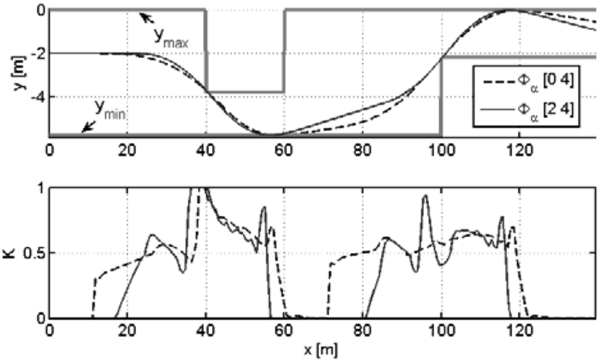


Fig. 5 Simulation results showing the effect of different intervention thresholds ([Φ_{eng} Φ_{aut}] = [0 4], [2 4]) on semi-autonomous corridor-tracking performance

As Fig. 4 and Fig. 5 illustrate, increasing Φ_{eng} delays controller intervention K at the cost of more rapid increases and more frequent saturation of the control authority allotment. This late intervention, while allowing the human driver greater autonomy far away from constraints/hazards, often requires a similar average control authority allotment as it must rapidly and forcefully regain control of the vehicle if the driver does not make the correction on their own. For example, increasing Φ_{eng} from 0 to 1 deg as shown in Fig. 4 ultimately decreased the average intervention K over the entire maneuver by only 12 %. Similar results were observed over the entire range of interest in Φ_{eng} and Φ_{aut} ($0 \leq \Phi_{eng} \leq 2$ and $2.5 \leq \Phi_{aut} \leq 5$), with average intervention K varying by less than 0.19. These results suggest that over the course of some maneuvers, this framework tends to average out controller intervention for various Φ_{eng} and Φ_{aut} settings, allowing for considerable driver preference tuning without dramatically changing average K.

5 Experimental Setup

Experimental testing was performed using a test vehicle and three human drivers. Driver and actuator steering inputs were coupled via an Active Front Steer (AFS) system. An inertial and GPS navigation system was used to measure vehicle position, sideslip, yaw angle, and yaw rate while a 1 GHz dSPACE processor ran controller code and interfaced with steering actuators.

Three common scenarios, including lane keeping, hazard avoidance, and multile hazard avoidance were used to analyze system performance. In each scenario, obstacles, hazards, and driver targets were represented to the driver by cones and lane markings and to the controller by a constrained corridor (with onboard sensing and constraint mapping assumed to have been performed previously by “virtual sensors” and high-level planners respectively).

Lane keeping experiments tested the threat assessment and intervention characteristics of the controller when the driver maneuvered inside and outside of the given lane. Six pairs of cones were set up along ~200 meters of the 3.35-meter-wide lane to guide the driver’s intended path. As shown in Fig. 6 (not to scale), the second and third sets of cones required the driver to steer the vehicle to the edge of the safe lane while the final two targets required that he attempt to leave the lane.

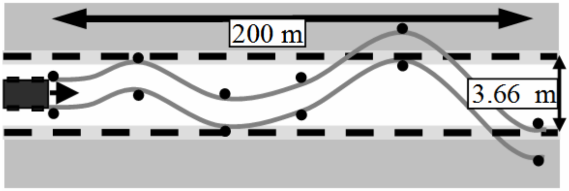


Fig. 6 Lane keeping test setup showing circles where cones were placed to guide the human driver’s (unassisted) path. Lane boundaries delineated by dashed lines were represented as constraints y^{y}_{min} and y^{y}_{max} to the controller

Hazard avoidance tests required that the vehicle avoid an obstacle in the current lane of travel. In these tests, the vehicle was driven at a constant velocity in the center of a lane with the driver holding the steering wheel at δ = 0 as if drowsy or inattentive. A row of cones blocked the vehicle's lane of travel, requiring the controller to: 1) plan a stable lane change maneuver around them, 2) assess the threat posed by that maneuver, and 3) intervene as necessary to avoid the hazard. Fig. 7 illustrates this test setup.

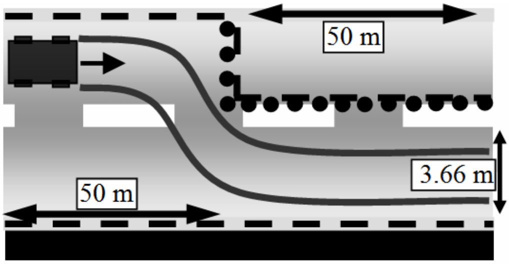


Fig. 7 Hazard avoidance test setup showing hazard cone placement (large circles) and lane boundaries (dashed) enforced by the controller

Multiple hazard avoidance experiments tested the controller's ability to navigate more complex road/hazard setups that required maneuvers with appreciable load transfer. In these tests (illustrated in Fig. 8), both lanes of travel were blocked at different locations, forcing the vehicle to change lanes to avoid the first hazard, then change lanes again to avoid the second as in a double-lane-change maneuver.

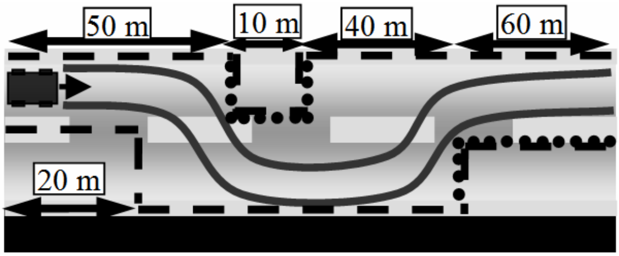


Fig. 8 Multiple hazard avoidance test setup showing hazard cone placement (circles) and lane boundaries (dashed)

Hazard avoidance and multiple hazard avoidance tests were conducted using two different types of driver inputs. Drowsy, inattentive, or otherwise impaired drivers were represented by a constant driver steer input of zero degrees. In these tests, the unassisted driver's path formed a straight line directly through the obstacle(s). To represent active driver steer inputs, the drivers were asked in separate tests to steer either around or into the obstacles. The urgency of these driver steer events was varied – sometimes avoiding the obstacle(s) with a smooth input and other times, steering at the last minute. Controller parameters were chosen as defined in Table 2. Experiments were conducted at 5, 10, and 14 meters per second.

6 Experimental Results

The semi-autonomous framework proved capable of keeping the vehicle within the navigable corridor for each of the maneuvers tested, with three different human drivers, and using multiple intervention laws. Additionally, the 50 ms sample time proved sufficient for control calculations.

Fig. 9 shows the results of lane-keeping tests.

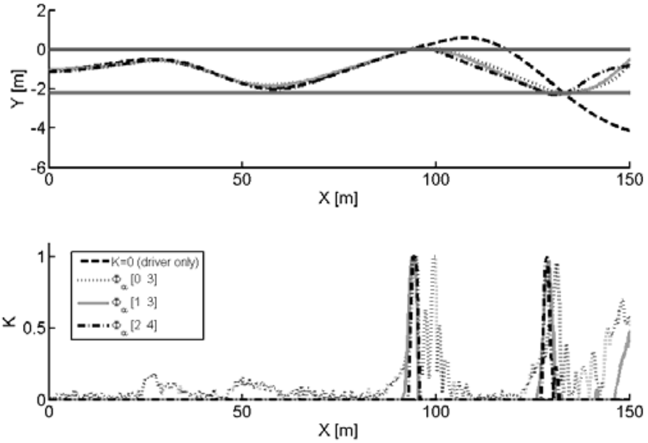


Fig. 9 Results of lane keeping tests with no controller action (dashed), and semi-autonomous controller intervention (dotted, solid, and dash-dot)

The dashed black line in Fig. 9 represents the vehicle trajectory under complete driver control ($K = 0$), and is shown here and in subsequent plots as a reference for the trajectory the driver would have followed had the semi-autonomous controller not engaged. Note that for multiple engagement (Φ_{eng}) and autonomous (Φ_{aut}) intervention thresholds, the semi-autonomous controller successfully kept the driver within the navigable corridor while allowing him significant control authority as long as he remained inside the navigable corridor (x~-20m to x~70m). Only when the vehicle was about to depart from the corridor did the controller intervene. In each case, this intervention was early and large enough to arrest departure. The slight intervention observed between x~10m and x~50m illustrates the framework's response to a predicted trajectory that required appreciable sideslip in order to remain within the lane. When the driver corrected the vehicle heading, *K* returned to approximately zero. Also note that the inclusion of a low-threat "deadband" ([1 3] and [2 4]) removed much of the noise seen for the experiment without a deadband ([0 3]).

Fig. 10 shows the results of multiple hazard avoidance experiments.

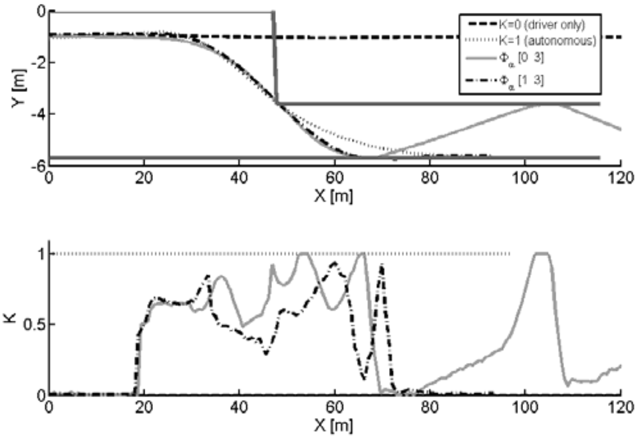


Fig. 10 Results of hazard avoidance tests with no controller action (dashed), fully-autonomous control (dotted), and semi-autonomous control (solid and dash-dot)

Similar to the lane keeping results shown in Fig. 9, controller intervention in hazard avoidance tests was sufficient to maintain the driver's natural/unassisted trajectory for as long as possible before taking control. When the framework did intervene, it allocated enough control authority to the controller to avert corridor departure or loss of control. Note that the trajectory oscillation observed in the [0 3] semi-autonomous experiment was a result of an overcorrection on the part of the controller at x~65m. The vehicle trajectory proceeded to rebound from of $y^y_{\max}$ because the driver's input remained at $\delta_{\text{driver}} = 0$. Were the driver more attentive as a result of the first intervention incident, the low levels of *K* directly following the initial rebound would have allowed him significant control authority to correct and straighten out the vehicle.

Fig. 10 also shows the results of an autonomous experiment in which the controller was given full control authority (*K*=1). Notice that for the given driver input ($\delta_{\text{driver}} = 0$), the vehicle path under semi-autonomous control closely resembles the "best case" path achieved using autonomous control while exerting only an average intervention level (*K*) of 0.34.

Fig. 11 compares a semi-autonomous multi-hazard-avoidance maneuver to an autonomous maneuver (*K*=1).

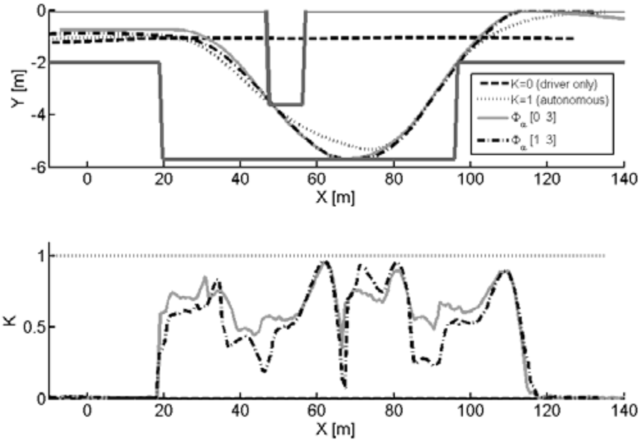


Fig. 11 Multiple hazard avoidance tests showing the similarity between vehicle trajectories under semi-autonomous (solid and dash-dot) and autonomous (dotted) control

Notice that both semi-autonomous controller configurations delayed intervention until the driver's inputs put the vehicle at risk of leaving the navigable road surface. When the framework did intervene, it allocated enough control authority to the controller to avert safe lane departure or loss of control. Even with average controller intervention K_{ave}=0.44, the ultimate vehicle trajectory using the semi-autonomous controller very closely resembles the "best case" trajectory taken by the autonomous controller. This arises from the selective nature of the semi-autonomous system – it intervenes only when necessary, then relinquishes control to the driver once predicted threat to the vehicle has been reduced.

Fig. 12 shows experiments in which the driver was instructed to swerve at the last minute to avoid hazards.

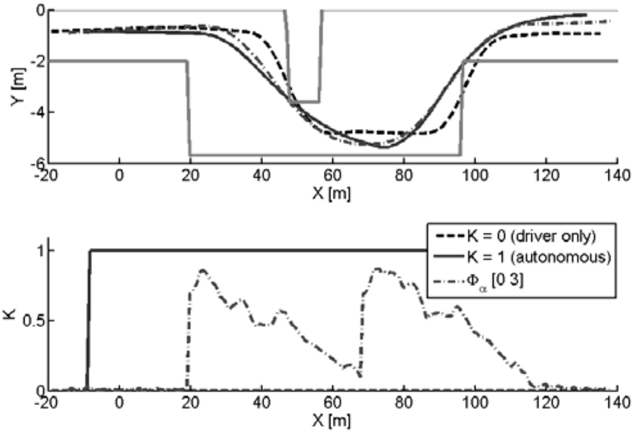


Fig. 12 Multiple hazard avoidance tests showing the vehicle trajectory with an unassisted driver input (dashed) and autonomous controller (solid), and semi-autonomous controller (dash-dot). In each case, the driver swerved to avoid hazards

As Fig. 12 shows, intervention by the semi-autonomous controller slightly preceded an otherwise-late driver reaction. The combined effect of both inputs was then sufficient to avoid both road hazards.

In each of experimental results shown above, the shared-adaptive controller behaves as a stable closed-loop system. While this was also true of all of the other simulated and experimental results conducted to date, no rigorous stability proof is presented in this paper.

7 Conclusions

This paper presented an optimal-control-based framework that performs trajectory planning, threat assessment, and semi-autonomous control of passenger vehicles in hazard avoidance. This framework has been shown in simulation and experiment to satisfy position, input, and dynamic vehicle constraints using multiple threat metrics and intervention laws. Additionally, this framework has been shown to provide significant autonomy to a human driver, intervening only as necessary to keep the vehicle under control and within the navigable roadway corridor. Simulation and experimental results have shown this control framework to be stable even in the presence of system-inherent time delays, though a rigorous stability proof is a topic of current investigation.

Finally, while human factors have not been studied in depth here, it is expected that with additional investigation, a best-case, or average driver-preferred intervention law may be described and intervention settings tuned accordingly. Further work is needed before this research is road-ready.

Acknowledgments. The authors would like to thank Eric Tseng, Matt Rupp, Mitch McConnell, Len Johnson, Steve Hermann, Kyle Carey, Reid Steigler, Tim Zwicky, Roger Trombley, Chris Wallis, and Jeff Rupp, all of Ford Motor Co. for their assistance in conducting the abovementioned experiments.

References

[1] National Highway Traffic Safety Administration (NHTSA), 2007 Traffic Safety Annual Assessment - Highlights, NHTSA National Center for Statistics and Analysis (2008)

[2] Weilkes, M., Burkle, L., Rentschler, T., Scherl, M.: Future vehicle guidance assistance - combined longitudinal and lateral control. In: Automatisierungstechnik, vol. 42, pp. 4-10 (January 2005)

[3] Leonard, J., How, J., Teller, S., Berger, M., Campbell, S., Fiore, G., Fletcher, L., Frazzoli, E., Huang, A., Karaman, S., Koch, O., Kuwata, Y., Moore, D., Olson, E., Peters, S., Teo, J., Truax, R., Walter, M., Barrett, D., Epstein, A., Maheloni, K., Moyer, K., Jones, T., Buckley, R., Antone, M., Galejs, R., Krishnamurthy, S., Williams, J.: A perception-driven autonomous urban vehicle. Journal of Field Robotics 25, 727–774 (2008)

[4] Jansson, J.: Collision avoidance theory with application to automotive collision mitigation. Doctoral Dissertation, Linkoping University (2005)

[5] Pohl, J., Birk, W., Westervall, L.: A driver-distraction-based lane-keeping assistance system. Proceedings of the Institution of Mechanical Engineers. Part I: Journal of Systems and Control Engineering 221, 541–552 (2007)

[6] Mobus, R., Zomotor, Z.: Constrained optimal control for lateral vehicle guidance. In: Proceedings of the 2005 IEEE Intelligent Vehicles Symposium, June 6-8, pp. 429–434. IEEE, Piscataway (2005)

[7] Alleyne, A.: A comparison of alternative obstacle avoidance strategies for vehicle control. Vehicle System Dynamics 27, 371–392 (1997)

[8] Netto, M., Blosseville, J., Lusetti, B., Mammar, S.: A new robust control system with optimized use of the lane detection data for vehicle full lateral control under strong curvatures. In: ITSC 2006: 2006 IEEE Intelligent Transportation Systems Conference, September 17–20, pp. 1382–1387. Institute of Electrical and Electronics Engineers Inc., Piscataway (2006)

[9] Vaidyanathan, R., Hocaoglu, C., Prince, T.S., Quinn, R.D.: Evolutionary path planning for autonomous air vehicles using multiresolution path representation. In: 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems, October 29-November 3, pp. 69–76. Institute of Electrical and Electronics Engineers Inc. (2001)

[10] Rossetter, E.J., Christian Gardes, J.: Lyapunov based performance guarantees for the potential field lane-keeping assistance system. Journal of Dynamic Systems, Measurement and Control, Transactions of the ASME 128, 510–522 (2006)

[11] Falcone, P., Tufo, M., Borrelli, F., Asgarit, J., Tseng, H.: A linear time varying model predictive control approach to the integrated vehicle dynamics control problem in autonomous systems. In: 46th IEEE Conference on Decision and Control 2007 (CDC), December 12–14, pp. 2980–2985. Institute of Electrical and Electronics Engineers Inc., Piscataway (2008)

[12] Engelman, G., Ekmark, J., Tellis, L., Tarabishy, M.N., Joh, G.M., Trombley, R.A., Williams, R.E.: Threat level identification and quantifying system. U.S. Patent US 7034668 B2 April 25 (2006)

[13] Yu, H., Spenko, M., Dubowsky, S.: An adaptive shared control system for an intelligent mobility aid for the elderly. Autonomous Robots 15, 53–66 (2003)

[14] Guo, L., Wang, J., Li, K.: Lane keeping system based on THASV-II platform. In: IEEE International Conference on Vehicular Electronics and Safety (ICVES 2006), December 13-15, pp. 305–308. Institute of Electrical and Electronics Engineers Computer Society, Piscataway (2006)

[15] McBride, J.R., Ivan, J.C., Rhode, D.S., Rupp, J.D., Rupp, M.Y., Higgins, J.D., Turner, D.D., Eustice, R.M.: A perspective on emerging automotive safety applications, derived from lessons learned through participation in the DARPA Grand Challenges. Journal of Field Robotics 25, 808–840 (2008)

[16] Zomotor, Z., Franke, U.: Sensor fusion for improved vision based lane recognition and object tracking with range-finders. In: Proceedings of the 1997 IEEE Conference on Intelligent Transportation Systems, ITSC, November 9–12, pp. 595–600. IEEE, Piscataway (1997)

[17] Garcia, C., Prett, D., Morari, M.: Model predictive control: theory and practice-a survey. Automatica 25, 335–348 (1989)

[18] Falcone, P., Borrelli, F., Asgari, J., Tseng, H.E., Hrovat, D.: Predictive active steering control for autonomous vehicle systems. IEEE Transactions on Control Systems Technology 15, 566–580 (2007)

[19] Keviczky, T., Falcone, P., Borrelli, F., Asgari, J., Hrovat, D.: Predictive control approach to autonomous vehicle steering. In: 2006 American Control Conference, June 14-16, pp. 4670–4675. Institute of Electrical and Electronics Engineers Inc., Piscataway (2006)

Autonomy through SLAM for an Underwater Robot

John Folkesson and John Leonard

Abstract. An autonomous underwater vehicle (AUV) is achieved that integrates state of the art simultaneous localization and mapping (SLAM) into the decision processes. This autonomy is used to carry out undersea target reacquisition missions that would otherwise be impossible with a low-cost platform. The AUV requires only simple sensors and operates without navigation equipment such as Doppler Velocity Log, inertial navigation or acoustic beacons. Demonstrations of the capability show that the vehicle can carry out the task in an ocean environment. The system includes a forward looking sonar and a set of simple vehicle sensors. The functionality includes feature tracking using a graphical square root smoothing SLAM algorithm, global localization using multiple EKF estimators, and knowledge adaptive mission execution. The global localization incorporates a unique robust matching criteria which utilizes both positive and negative information. Separate match hypotheses are maintained by each EKF estimator allowing all matching decisions to be reversible.

1 Introduction

The underwater ocean environment poses tremendous challenges to an autonomous underwater vehicle (AUV). Some of these are common to those faced by a remotely operated vehicle (ROV). Hull integrity under pressure, vehicle locomotion, power source, control, and stable vehicle attitude are all special constraints for underwater systems. Operating without physical, radio, or visual contact to the vehicle implies an increased degree of autonomy. To achieve autonomy further challenges must be

John Folkesson
Massachusetts Institute of Technology
e-mail: johnfolk@mit.edu

John Leonard
Massachusetts Institute of Technology
e-mail: jleonard@mit.edu

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 55–70.
springerlink.com

addressed that relate to the machine's decision making. These challenges are mainly in the areas of perception, localization, and reasoning. All three of these areas must accurately deal with uncertainties. Perception is limited by the physical properties of water; localization is made difficult by the presence of tides, wind driven currents, and wave surge, which can easily disorient a robot while underwater.

Underwater machine proprioception uses a vehicle model and the actuation signals, often implemented as the prediction step of an extended Kalman filter (EKF). Additional proprioception can be provided by an inertial sensor. Available exteroceptive sensors are depth, altitude, gravitation, and magnetic field. These sensors give absolute measurements of four of the six degrees of freedom (DOF) for the vehicle motion. The other two DOF are the position in the horizontal plane or local grid. When the motion of the vehicle in this plane is estimated solely using proprioceptive localization, the errors in this so called dead-reckoning estimate grow without bound.

There are sensors to give absolute measurements of these horizontal plane motions. One can augment the natural environment with acoustic beacons to set up an underwater system analogous to the satellite global positioning system (GPS) for open air systems. Recent state-of-the-art work with these long baseline (LBL) navigation systems is provided by Yoerger *et al.* [1]. Another common technique is to obtain Doppler velocity measurements using a Doppler Velocity Log (DVL), as illustrate by Whitcomb *et al.* [2]. These techniques are the solutions of choice for many types of missions in which low cost of the AUV is not a primary concern.

Errors in localization will lead to decision mistakes and failed missions. In some cases absolute localizations is less important than localization relative to some underwater features such as pipelines, cables, moored objects, or bottom objects. These cases require correct interpretation of exteroceptive sensors capable of detecting the features. The most useful underwater exteroceptive sensor for this process is sonar. Sound travels well under all conditions while light is useful only in clear water at short range. The sonar data can be formed into an image that can be processed using some of the same tools used to process camera images. The projective geometry and the interpretation of images is different so, for example, shadows are formed on the far side of objects relative to a sound source. These shadows give a great deal of information to a skilled human analyst.

There are a number of different types of sonar systems that produce images comparable to camera images. Side-scan sonar can form narrow sonar beams restricted to a plane perpendicular to the sonar array. The returns from these beams can be pieced together over the trajectory of the vehicle to give a detailed image of the sea floor. Forward looking sonars can generally see in a pie slice shaped wedge in front of the sonar. They usually have good angular resolution over a span of angles about an axis and good range resolution. The resolution in beam angles to the axis is relatively wide forming the wedge as the sound propagates.

Information from one sonar image is of limited use to the AUV. It is the repeated observation of the same objects that helps to remove disturbances from the motion estimate. It is also by tracking objects over many images that noise can be distinguished from the features of interest which can then be recognized as being relevant to the mission goals. Thus knowledge acquired from these observations can be compared to knowledge acquired prior to and during the current mission. As such, the state of the vehicle can depend on the observations and the mission execution becomes adaptive to these observations. Missions that would be impossible to carry out even with perfect absolute localization can now be envisioned as feasible with imperfect absolute localization. This is the power of the autonomous intelligent machine.

Our system is designed to carry out target reacquisition missions at low cost and without the use of pre-set acoustic beacons as positioning aids. The use of a DVL to correct for the effects of water currents is not part of the system, for cost considerations. The current state of the art for these types of reacquisition missions is to use human divers; this is impossible or undesirable for many situations, and an AUV-based system that could attach itself to a target offers many advantages. The system will find a specified moored target using an *a priori* map of the area's features. These features can be various moored or bottom objects that happen to be in the area. The *a priori* map is made using an AUV equipped with sophisticated sensors and a side-scan sonar. The mission critical decision centers on the position of the vehicle relative to this *a priori* map. The method to make this decision is to use SLAM to build a map of the sonar observations [3, 4, 5, 6, 7] and match this map to the *a priori* map in an EKF [8, 9, 10, 11]. Several EKF estimators are run in parallel allowing multiple match hypotheses to be considered. All matching decisions are reversible in that one of the hypotheses (known as the null hypothesis) makes no matches between the features; this null hypothesis can then be copied to any of the other estimators essentially resetting them and allowing them to be matched differently.

The local map building is done in a tracking filter. This computation is both accurately and efficiently carried out using a Square Root Smoothing SLAM algorithm. The tracking filter implements the algorithm as a graph over the vehicle states and feature positions. The information tracked over a number of frames is consolidated into composite measurements and used to update the bank of EKF estimators at a low frequency. Thus problems of complexity as the size of the slam graph grows are avoided by cutting off sections of the graph periodically and feeding them as single measurements of both features and AUV pose to the EKF estimators. One gets the accuracy of the graphical square root slam model that re-linearizes all measurements about the current solution without any explosion of complexity. At the same time one can use several EKF estimators updated with these accurate measurements at low frequency to allow multiple hypotheses and reversible decisions. As the matching is also done at this low frequency, the matching algorithm can be of higher complexity.

2 Mission Scenario

The mission starts with a survey of an undersea region containing moored and bottom features by an AUV equipped with sophisticated navigation and side-scan sonar.

This data is analyzed by an expert and an *a priori* map of features is produced. One of the moored features is identified as the target of interest to be reacquired. The *a priori* map and chosen target are input to our AUV system and then the vehicle is released from a distance of between 100 to 1,500 m from the field (Fig. 1). The AUV travels to a point about 70 m out and dives on the field. It surveys the field and builds a SLAM map of the features. When the AUV has achieved good localization relative to the *a priori* map, it will attempt to 'capture' the target. That is, still using the features for navigation, it will maneuver to drive its V shaped grippers into the mooring line of the target. At that point it will close the grippers tightly on the line and become attached.

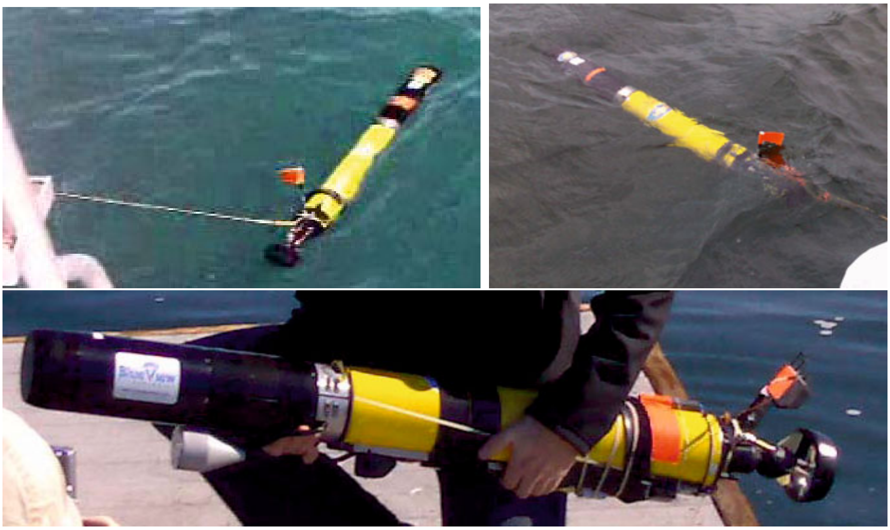


Fig. 1 The iRobot Ranger AUV is equipped with a Blueview Blazed Array sonar in the nose section. It is launched by hand.

3 System Components

The AUV is a Ranger manufactured by iRobot and equipped with depth sensor, altimeter, GPS, a 3D compass, and a Blueview Blazed Array forward looking sonar (FLS) as shown in Fig. 1.

The software system consists of 7 functional modules. A robust and fast detector finds virtually all the interesting features in the sonar images with a few false positives. These are fed to a tracking filter that is able to filter out the false positives from the real objects and form 'composite measurements' out of the sonar and motion measurements over a section of the path. These composite measurements are then input to the bank of EKF SLAM estimators. These try to match the *a priori* map to the SLAM map using a special matching criteria described in a later section.

The output of these estimators as well as information on the currently tracked features is passed to the interpreter which extracts information relative to the current mission objectives. This information is requested by the mission executor and used to make the high level decisions leading to a control output sent to the actuators. Final capture of the target is done using a sonar servo PID controller.

In addition, there is a dead reckoning service which provides motion estimates to the tracking filter, the EKF estimators, the mission executor, and the sonar servo control. This dead-reckoning service uses a detailed model of the robot and environment along with two separate EKF estimators to provide estimates of the motion with and without GPS. (GPS is of no use underwater but is used to initialize the position before dives.)

4 Sonar Feature Detector

The Blazed Array FLS has two sonar heads, a vertical and a horizontal, giving a position in 3D for features detected in both. The field of view of each head is 45 degrees by 30 degrees. The returns are given as range and angle in the plane of the 45 degree direction. Thus measurements have a large uncertainty perpendicular to this plane, extending outward in a 30 degree wedge shape. The effective range of the sonar is 40 m for good sonar reflectors and about 20 m for normal bottom objects such as rocks. Thus to get enough detections of normal bottom objects to be of real use the robot must pass within about 5 m of the object. This makes finding the objects of the *a priori* map a challenge as currents can easily throw the AUV off by more than this amount. The cruising speed of the Ranger is about 1.2 knot. The currents encountered during our validation testing were typically more than 0.5 knot and vary with time, depth, and in the horizontal plane.

The sonar horizontal head has been modified to tilt downward at 5 degrees to give a better view of the bottom targets. The formation of vertical and horizontal images out of the raw sonar data using the Blueview SDK library takes about 180 ms of processor time per ping. The feature detection algorithm then adds about 18 ms or 10% to this time. The steps used to process the images are as follows:

1. Form a 200 x 629 image of the vertical head sonar data, (in Fig. 2 the images are rotated 90 degrees).
2. Down-sample this to 50 columns (rows in Fig. 2).
3. Find sea bottom, altitude, and relative slope using line extraction.
4. If slope (pitch) is too far from level flight stop.
5. Form a 200 x 629 image of the horizontal head sonar data.
6. Down-sample this to 50 columns (rows in Fig. 2).
7. Use the altitude from step 3 to select one out of a number of averaged background images each for a specific altitude range.
8. Low pass filter the current horizontal image into the background image.
9. Subtract the background image (noise) from the horizontal image to form two separate images, one the normalized image and the other the shadow image.
10. Segment vertical image into bottom and open water.

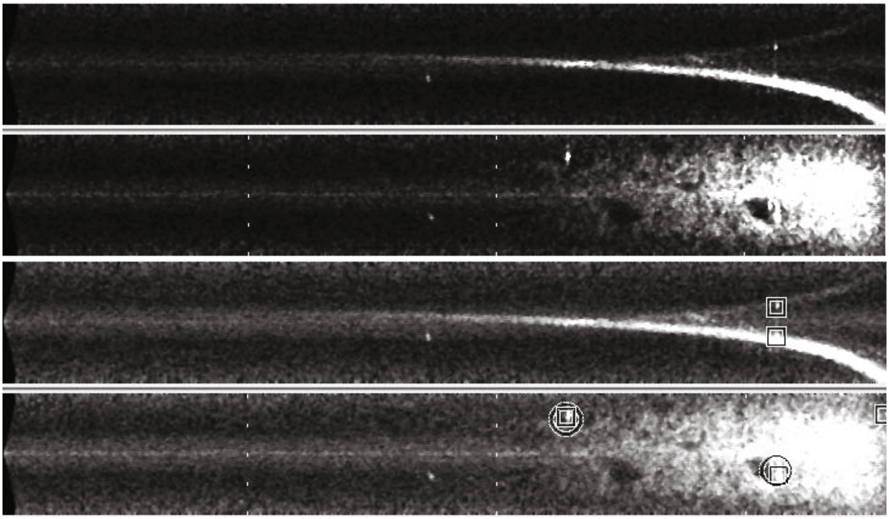


Fig. 2 Here is an example of the vertical (upper) and horizontal (lower) sonar head images. The top pair are the input to the detection module and the bottom pair show the output feature detections highlighted as circles for shadows and squares as edges. The images are rotated to show the range increasing from right to left. The white hash marks are spaced at 10, 20, and 30 m in the horizontal images for scale reference. The polar angle is shown in the as increasing downward in the images and has a total range of 45 degrees. The bright arc in the upper images is the sea bed and would appear as a straight line if plotted in Cartesian space instead of polar coordinates as shown here. The near-field region of open water is barely seen in these images due to the AUV pitch. It would normally be a wider darker region along the right edge of the horizontal image, here seen very narrow.

11. Segment horizontal image into three regions near field open water, ranges where the bottom is sufficiently bright to see shadows, and the remaining range out to 40 m.
12. Low pass filter the image segments pixels along each bearing (column) to smooth them eliminating very small features and noise.
13. Search for edges as gradients in each segment with thresholds based on the average background image level. (This gives some adaption to different bottom types.)
14. Search for shadows (below noise floor) in bottom segment coupled with an adjacent bright area closer to the sonar from the object that created the shadow.
15. Then select for output any feature found in both heads and the strongest remaining detections in each segment.

Step 3 allows the algorithm to process differently based on the altitude and pitch of the vehicle which is important as the esonification of the bottom will be dependent on these. Thus the image can be segmented into regions of open water and sea floor. The regions of open water are much less noisy and can therefore have lower detection thresholds. The down-sampling steps, 2 and 6, are crucial to fast processing without the loss of information. The sonar has an angular resolution of 1 degree so that 50 columns will be sufficiently fine for the 45 degree field of view. The averaged background images at each altitude are important for normalizing the image (step 9) and for detecting shadows (step 14). In step 9 the normalized pixels will be set to the difference when positive or 0 while the the shadow pixels will be set to minus the difference when this is positive or 0.

5 Mapping

The mapping approach utilizes the strengths of two SLAM methods each focused on separate aspects of the mapping problem. One aspect involves the tracking of feature detections and using them to improve the dead-reckoning over local map areas of 10-50 m in size. The other aspect involves piecing these local map estimates together to give an estimate of the map over the entire field and to then provide information on uncertainties to the map matching algorithm. Fig. 3 provides an aid to understanding the workings of the various parts of the mapping approach.

The feature tracking uses an accurate Square Root Smoother [12, 13] as previously described in [14, 15]. The inputs to this incremental Gaussian estimator are the sonar detections of features and the dead-reckoning estimates between the sonar measurements. The filter estimates the Gaussian Maximum Likelihood state along with a representation of its covariance. The state vector consists of the locations of the observed features and the all the poses of the AUV at these observation times. This representation allows us to delay initialization of features until we have gathered enough information on them. The initialization can then add all the previous measurements. Thus information is neither lost nor approximated. At any point the entire estimator can be re-linearized around the current state which improves consistency of the estimate. Thus the tracking filter avoids two of the problems encountered in SLAM estimation, consistency and loss of information. The trade off is increased computational complexity. This complexity is limited by the relatively small size of the local maps.

Periodically a section of the path along with all its measurements is cut out and formed into an independent estimator for just that subset of measurements. Then all variables for the intermediate pose states are eliminated by marginalizing them out forming a composite measurement with a state vector consisting of the starting and ending AUV poses and the feature locations. This mean and covariance estimate can then be used in an update of the bank of EKF estimators. The composite measurements are local maps of the features seen along a section of the robot path. These local maps are then joined using the bank of EKF estimators.

The EKF estimators have a state that consists of the AUV pose at the time of the last update and the locations of all features, both from the *a priori* map and the composite measurements. Thus the EKF SLAM filters first augment the state with the new pose and features, then do an EKF update step on the features locations and the two poses, then marginalize out the earlier pose and merge any features that were tracked between two adjacent local maps. There is no explicit prediction step.

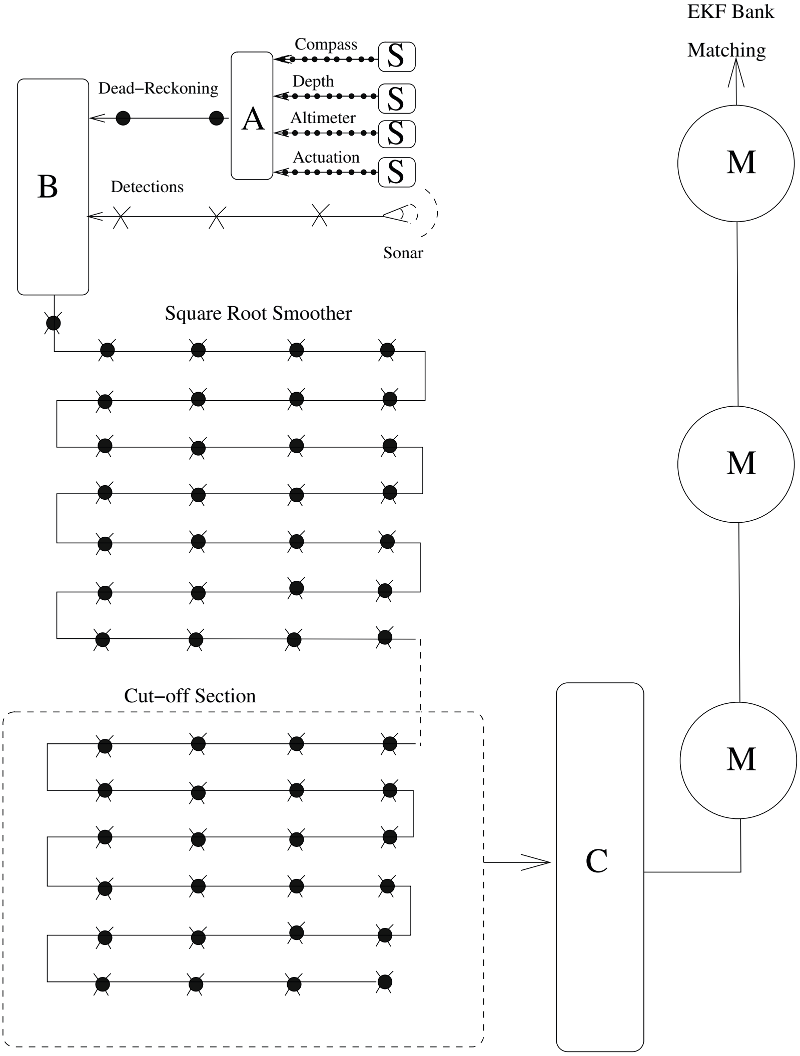


Fig. 3 We illustrate the compression of information in the various stages of mapping. We use an EKF in block (A) to combine the motion measurements of depth, altitude, compass, and actuation from sensors (S) into dead-reckoning estimates synchronized with the sonar feature detections. These are then used in block (B) to add nodes to the square root smoother graph. This smoother performs the feature tracking function eliminating false detections and forming maximum likelihood Gaussian estimates of the remaining detections. As this graph grows, sections are periodically cut-off reducing the number of nodes in the remaining graph. The cut-off section becomes an independent smoother graph. In block (C) we form composite measurements (M) out of the cut-off section of the graph and feed it to the bank of EKF estimators for matching to the *a priori* map. These composite measurements (M) of features and poses are local maps. The GPS measurements also update the EKF Bank. The EKF Bank also perform the current estimation. The current estimate from the most aggressive EKF is used by (A) to provide predictions to (B).

Each EKF estimator of the bank, numbered 0 to n-1, can match the features observed on the local maps to the features of the *a priori* map. These matches can be different for each estimator and the EKF will compute the maximum likelihood mean given the chosen match. The EKFs can also compute the Mahalanobis distance corresponding to the match. Thus the bank of estimators allows parallel match hypotheses to be continuously evaluated and updated. Any estimator's state can be copied into another estimator. Thus incorrect matches can be copied over once a better match is found. Estimator number 0, the most conservative, allows no matching at all and on each iteration is used to reset estimator number 1 to this state. All the estimators are then matched. The matching criteria are increasingly aggressive as the estimator number increases. If this then produces a match better (more likely) than any higher numbered estimator it is copied to those estimators. The likelihood is measured using the Unseen Match Scale described in Sect. 6.

6 Matching

Matching correctly to the *a priori* map is the most critical decision of each mission. We base our matching algorithm on a quantity we call the Unseen Match Scale (UMS). The goal is to capture the likelihood of a given match hypothesis h. The use of negative information is illustrated in Fig. 4. We see that only hypothesis (c) can explain the negative information of not having seen both previously mapped features (the x's) on the recent local map. We match one prior map feature to the recently observed feature (the dot) while the other prior map feature has not yet been scanned by the sonar. The UMS quantifies this information as follows:

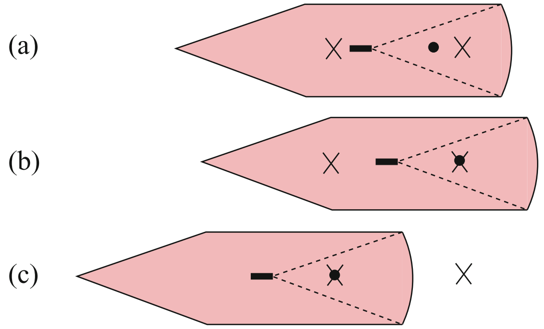


Fig. 4 Illustration of the Unseen Match Scale. The figure shows three match hypotheses. The shaded area shows the part of the environment scanned by the sonar on a recent local map with the dotted line showing the area of the last image. The small rectangle is the AUV at the last pose, the two x's are features from the *a priori* or earlier local map, and the dot is a feature on the most recent local map. Hypothesis (a) illustrates the unmatched situation (the null hypothesis). Hypothesis (b) cannot explain why one x was not seen. Hypothesis (c) explains why we have only one dot.

$$UMS(h) = -\Lambda N + D_h + U_h \tag{1}$$

The UMS has three terms. The first term is proportional to the number of matched feature pairs[1], N, where Λ is a parameter. The second is the Mahalanobis distance of the match as computed using the EKF covariance matrix. The third is the unseen feature energy defined as:

$$U_h = \ln(P(unseen|null)) - \ln(P(unseen|h)) \tag{2}$$

This expression is the negative of the log of the probability of all the unseen features normalized relative to the null hypothesis of making no new matches. Unseen features are features that were predicted to be in the robot's sensor field of view but were not initialized in the map. That is, these features are seen on one local map but are not seen on an overlapping local map. In Fig. 4, hypothesis (a) has three unseen features, (b) has one, and (c) has none.

We must compute the probability of the unseen features. For that we need several concepts. We form coarse grids over the area of each local map. Thus the same chain of local maps described in the last section will now be linked to a chain of grids summarizing the search for features by the sonar over the sections of path corresponding to the local maps. As the robot moves scanning the sea and sea floor with its sonar, we increment a counter in each scanned grid cell for each scan. This gives us a table containing the number of times each cell was scanned. We denote these numbers as s_{ig}, where i is the cell index and g is the index of the local grid.

We refer to the probability that the feature will be detected when the cell containing it is scanned as the feature's visibility, v. The feature will be initialized on the map if it is detected a number of times. We call this the detection threshold, n_d.

We define $Q_{fg}(h)$ as the probability of not initializing feature f predicted to lie in grid cell i of grid g. It can be computed from a binomial distribution.

$$Q_{fg}(h) = \sum_{j=0}^{n_d-1} \begin{pmatrix} j \\ s_{ig} \end{pmatrix} v^{(s_{ig}-j)}(1-v)^j. \tag{3}$$

The sum is over the number of times the feature may have been detected without being initialized.[2] We can form these sums for every feature and every local map on which it was not initialized. We can then compute the unseen feature energy of eq. (2) as the sum of these Q_{fg} over all the unseen features.

[1] This is the criteria used in the standard Joint Compatibility Branch and Bound algorithm[16] along with a threshold on the Mahalanobis distance.

[2] The cells i of eq. (3) are inferred based on the match of h. This is done by computing the new state generated by making the match. The new feature states will then imply a new transform to the local grid frames based on the features seen on the grids. This then is used to transform the unseen features to the grids and finally find the cell i that the feature falls in. See the appendix for other details.

$$-\ln(P(unseen|h)) = -\sum_{fg\varepsilon unseen} \ln Q_{fg}(h) \tag{4}$$

Notice we get a contribution here that can be directly traced to a particular feature on a particular local map. High values for the unseen feature energy contribution, $-\ln Q_{fg}$, indicate a prime candidate feature f for matching to features on the corresponding local grid g. This allows directed searches of the match hypothesis space.

Matched features are no longer unseen. Thus, the sum over unseen features will not include terms from the grids that had any feature matched to the feature we are considering. That gives a gain for matching that is not arbitrary or heuristic, but rather is precisely computed based on the simple statistics accumulated on the grids. The only parameter selected by hand is Λ. A probabilistic interpretation for Λ is presented in [17]. For the results presented here, good results have been obtained with Λ either set to zero or given a small value.

The complexity of calculating the Q_{fg} will be less than the number of features times the number of grids. Only features that fall on the local grid need to have Q_{fg} calculated. Thus the complexity is dependent on the path of the robot and the success of the matching. If the complexity were to become a problem, one could address this by considering only matches between features on the most recent grids and the rest of the grids. Thus we would give up trying to make matches that we failed to make on earlier iterations. That would then result in constant complexity as the map grew. Typically the complexity is not a problem so long as the matching is working properly. In that case the features are merged and no longer considered as unseen. Thus the number of Q_{fg} to be computed is reduced.

We must estimate the feature's visibility, v. We do this by examining the grids on which the feature was seen. We divide the number of detections by the sum of the s_{ig} for the feature.[3] We sum the s_{ig} using a Gaussian weighting over a window around the feature's predicted cell.

While the unseen feature energy, U_h of eq. (2), normally decreases as matches are added to the hypothesis, it does not *always* decrease, because the new state for the features given the match hypothesis h may imply a transformation which causes more overlap of local grids and a higher value for some Q_{fg}. Hence, some matches result in a decrease in the value of unseen feature energy.

All matches result in a decrease in the first term of the UMS eq. (1). The decrease in value may more than offset the inevitable increase from the second term of the UMS. The match with the lowest UMS is tested to see if it is ambiguous. If not, the match is made. In order to test ambiguity, the difference between the UMS for all matches, including null, and the UMS for the best match is checked. We threshold this difference with a parameter which we call the ambiguity parameter. We will make the match if this ambiguity parameter is less than the difference for all matches that conflict with the candidate match. Matches that agree with the candidate match, that is they give the same match pairs for all features in common, are

[3] In [18] they use the negative information of not observing a feature to remove spurious measurements from the map. This is another example of the usefulness of negative information. We could, in the same spirit, remove features with very low values of v.

not conflicting.[4] The null hypothesis is considered conflicting as are hypotheses that for some feature in common give a matched feature that is both different from and local to the matched feature of the best hypothesis. By local we mean that the two features are on the same local grid. Local features are always considered impossible to match to one another. If they were the same the local map builder should have matched them.

Using this measure we can say that the conflicting hypotheses have an energy or minus log likelihood that is higher than the one chosen by more than the ambiguity parameter.

By adjusting this ambiguity parameter we are able to set levels of aggressiveness in matching. We utilize this when running multiple hypotheses in the bank of parallel EKF SLAM estimators. Normally the more aggressive hypothesis will have a lower total energy as calculated by the accumulated total of the UMS for all matches merged. Occasionally the aggressive match will be wrong and then a more conservative hypothesis may eventually be able to find the true match lowering its energy below the aggressive one. When this happens we can copy the lower energy solution to the aggressive matcher's estimator and continue. In that way the aggressive matcher always has the most likely solution given the data and the implied search of all the parallel run estimators.

7 Field Testing

The system has been extensively tested and refined over 11 sea trials starting in 2006 each lasting from 2 to 3 weeks. We started in benign environments with unrealistic highly reflective moored targets that were easily detectable in the FLS from 60 m and progressed to less ideal sea conditions, normal moored targets, and bottom features including natural features not detectable past 20 m range. Finally control and attachment to the target were added.

The UMS matching algorithm was first validated in an A/B comparison test in St. Andrews Bay, Florida in June 2007 in which the UMS and Joint Compatibility Branch and Bound (JCBB) criteria were alternately used over 18 trials on a field of strong reflective moored targets. The bay has a tidal current that gave the AUV significant dead-reckoning errors on most trials. The AUV was released about 100 m from the field and from 8 directions. The AUV made a single pass of the field and had to decide on the match before finishing the pass. The trials were paired so as to give nearly equal starting conditions for the two algorithms. A success was defined as achieving a correct match to the field followed by the vehicle's successfully exiting the field and then re-approaching it again in a movement of mock capture toward the correct target. No actual attachment was done. The results of the live test are summarized in table 1. The difference in the frequencies is positive by 1.41

[4] The reason we need to introduce the concept of not conflicting is that a hypothesis that is correct but not complete might have a UMS very close to the correct and complete hypothesis. This is true if the left out pair(s) of the incomplete hypothesis do not change the energy by very much.

standard deviations. This gives a 91% significance to the difference and indicates that the UMS did outperform the simpler JCBB matching criteria.

To give an unambiguous comparison in the live test, we only used one match hypothesis. We later ran the data off line with 4 hypotheses and got the improved result shown in the table[5]. Of the four missions it could not match, two were UMS runs and two were JCBB runs. This gives some measure of the fairness of the random elements of each run.

Table 1 In the 2007 tests the Joint Compatibility criteria, JCBB, uses a threshold on the Mahalanobis distance of the multiple pair match and chooses the most pairs. This was compared to the UMS criteria and the difference in rates was considered as proof that the UMS performed better. In the 2008 tests we used both moored and bottom targets which were detectable from between 20 and 40 m. In the March 2009 test there was no matching done as we tested the control to attachment on a single moored target in a test pond. In the June 2009 tests we added better modeling and estimation of currents along with better feature modeling.

Selected Test Results

Match Criteria	Runs n	Successes	Frequency	$\sqrt{s_n^2/n}$
Bright Targets - June 2007:				
UMS	9	6	67%	17%
JCBB	9	3	33%	17%
UMS - JCBB			33%	24%
UMS Multi-hypothesis	18	14	78%	10%
Normal Targets - June 2008:				
UMS Multi-hypothesis	9	3	33%	17%
No Current - March 2009:				
One Feature	17	17	100%	
Normal Targets - June 2009:				
UMS Multi-hypothesis (all)	26	17	65%	9%

Next the detection of features in the sonar images was substantially improved from a simple intensity detector in the horizontal image to the detection algorithm described in Sect. 4. We added a PID control and grippers to the robot to enable actual capture of the mooring lines of the target. We tested in Narragansett Bay, Rhode Island in June 2008, on a field consisting of various man-made and naturally occurring objects on the sea bottom. Again the bay had a significant tidal current which gave us substantial dead-reckoning errors. We did nine capture runs. Two of the

[5] On one of the failed multi-hypothesis runs the solution was switched to the correct match after the robot had returned to the surface, too late to be considered a success.

runs resulted in the AUV hitting the mooring line and breaking off the gripper arm, which we considered a success for the software system. We captured the mooring line on the ninth run of the day for an overall success rate of 33%.

We then improved the final target capture with the addition of sonar servoing of heading and altitude for the PID control. In March 2009, we performed tests in a man made test pond to evaluate the terminal homing behavior with one target and no current. In 17 missions in which the SLAM algorithm locked on to the target, the vehicle successfully latched onto the line all 17 trials. This provides validation for the final capture phase.

The overall robustness of the system was further improved via two further refinements to the matching algorithm that are described in the appendix. We also improved the model and estimation of current variation with depth. In June of 2009 the entire system was tested in the Gulf of Mexico. 15 objects were placed with a nominal 12-15 m spacing and forming three parallel lines. Three of the objects were moored while the others were bottom objects. Over a two week period the system was evaluated by running 26 missions of which 17 resulted in attaching to the chosen target. We found that the success rate depended strongly on the initial approach to the field. By having the robot travel in a straight heading towards the center of the field along the direction of the current we achieved 16 of 18 successes with the other 8 trials using other strategies. Surface currents ranged from 0.4 to 1.0 knots.

Of the 9 failed missions 4 had no match and 5 miss-matched. No mission managed to make a successful second attempt after getting a new GPS fix but 2 timed out after making a correct match. The miss-matches typically occurred after initially missing the field and accumulating large dead-reckoning errors. Five of the successful runs had initial miss-matches that were corrected before capture was attempted. In all failures there were too large dead-reckoning errors on reaching the field. The dead-reckoning errors were attributed to disturbances in the water which can not be estimated and compensated for. We have succeeded in developing a model of the variation of ocean currents with depth but this only can remove the disturbances that matches the model. No variation in the horizontal plane can be estimated. The model parameters are estimated on the fly during GPS pop-ups during the initial approach and prior to the final dive into the field.

8 Conclusions

This paper has presented a summary of a field deployed AUV navigation system that achieves a high level of autonomy to perform a challenging real-world mission. We have worked methodically towards creating a robust system to reliably reacquire underwater targets, reducing the danger to the manned personnel that typically perform these missions. We first broke the problem into specialized functional modules and optimized each separately. By adding new functionality and refinements based on repeated testing at sea, incremental improvements have accumulated to give the AUV a robust autonomy. The value of the system has been comprehensively demonstrated in numerous field trials over a three-year period. Continued testing is in progress to improve the target capture success rate of the overall system for increasingly difficult ocean conditions.

Acknowledgements. This work was funded by the Office of Naval Research under grant N00014-05-1-0244, which we gratefully acknowledge. We would also like to thank the other team members, iRobot, Blueview, Naval Postgraduate School, SeeByte, and NSWC-PC for their assistance.

References

1. Yoerger, D., Jakuba, M., Bradley, A., Bingham, B.: Techniques for deep sea near bottom survey using an autonomous underwater vehicle. International Journal of Robotics Research 26(1), 41–54 (2007)
2. Whitcomb, L., Yoerger, D., Singh, H., Howland, J.: Advances in Underwater Robot Vehicles for Deep Ocean Exploration: Navigation, Control and Survery Operations. In: The Ninth International Symposium on Robotics Research. Springer, London (2000) (to appear)
3. Williams, S., Dissanayake, G., Durrant-Whyte, H.: Towards terrain-aided navigation for underwater robotics. Advanced Robotics-Utrecht 15, 533–550 (2001)
4. Ribas, D., Ridao, P., Neira, J., Tardos, J.: Slam using an imaging sonar for partially structured underwater environments. In: Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS 2006). IEEE, Los Alamitos (2006)
5. Leonard, J.J., Carpenter, R., Feder, H.J.S.: Stochastic mapping using forward look sonar. Robotica 19, 467–480 (2001)
6. Tena, I., de Raucourt, S., Petillot, Y., Lane, D.: Concurrent mapping and localization using sidescan sonar. IEEE Journal of Ocean Engineering 29(2), 442–456 (2004)
7. Fairfield, N., Kantor, G.A., Wettergreen, D.: Towards particle filter SLAM with three dimensional evidence grids in a flooded subterranean environment. In: Proceedings of ICRA 2006, May 2006, pp. 3575–3580 (2006)
8. Dissanayake, M.G., Newman, P., Clark, S., Durrant-Whyte, H., Corba, M.: A solution to the simultaneous localization and map building (SLAM) problem. IEEE Transactions on Robotics and Automation 17(3), 229–241 (2001)
9. Newman, P., Leonard, J., Rikoski, R.: Towards constant-time SLAM on an autonomous underwater vehicle using synthetic aperture sonar. In: Proc. of the International Symposium on Robotics Research, ISRR 2003 (2003)
10. Leonard, J., Rikoski, R., Newman, P., Bosse, M.: Mapping partially observable features from multiple uncertain vantage points. IJRR International Journal on Robotics Research 7(3), 943–975 (2002)
11. Hahnel, D., Burgard, W., Wegbreit, B., Thrun, S.: Towards lazy data association in SLAM. In: The 11th International Symposium of Robotics, vol. 15, pp. 421–431. Springer, Heidelberg (2005)
12. Dellaert, F.: Square root SAM: Simultaneous location and mapping via square root information smoothing. In: Robotics: Science and Systems (2005)
13. Dellaert, F., Kaess, M.: Square root SAM: Simultaneous location and mapping via square root information smoothing. International Journal of Robotics Reasearch 25(12), 1181–1203 (2006)
14. Folkesson, J., Leonard, J., Leederkerken, J., Williams, R.: Feature tracking for underwater navigation using sonar. In: Proc. of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2007), pp. 3678–3684 (2007)

15. Folkesson, J., Leederkerken, J., Williams, R., Patrikalakis, A., Leonard, J.: A Feature Based Navigation System for an Autonomous Underwater Robot. In: Field and Service Robots, pp. 105–114. Springer, Heidelberg (2008)
16. Neira, J., Tardós, J.D.: Data association in stocastic mapping using the joint compatibility test. IEEE Transaction on Robotics and Automation 17(6), 890–897 (2001)
17. Folkesson, J., Christensen, H.I.: Closing the loop with graphical SLAM. IEEE Transactions on Robotics and Automation, 731–741 (2007)
18. Montemerlo, M., Thrun, S.: Simultaneous localization and mapping with unknown data association using FastSLAM. In: Proc. of the IEEE International Conference on Robotics and Automation (ICRA 2003), vol. 1, pp. 1985–1991 (2003)
19. Singh, H., Roman, C., Pizarro, O., Eustice, R.: Advances in high-resolution imaging from underwater vehicles. In: International Symposium of Robotics Research, San Francisco, CA, USA (October 2005)

Appendix

A few refinements must be made to the simple formula of eq. (3) in the actual implementation. First, the prediction of the feature location and the grid number s_{ig} are uncertain. We therefore use a Gaussian window over the grid cells and sum the contributions from adjacent cells near the predicted feature location. This gives a more continuous scale. Second, the details of the local map formation are such that features can be initialized by being detected n_d times in total on adjacent local maps. So we need to sum over the local maps before and after as well.[6]

Third, a minor adjustment needs to be made to avoid numerical problems such as extremely small probabilities. The Q_{fg} are limited to avoid their becoming too small. This is done by performing the following transformation on the values computed as described above and in the main text:

$$Q_{fg} \leftarrow p * Q_{fg} + (1-p) \tag{5}$$

We call p the model probability as it is the probability that our model of the features is correct. We used $p = 0.99$.

Two additional refinements were added to the algorithm this year. A fourth refinement is to accumulate grids at a number of depths and then use the grid at the observed feature's depth of the computation. A fifth refinement accumulated grids at three ranges from the robot: 0-8 m, 8-20 m, and 20-40 m. This adjustment allowed us to model the fact that some features are not visible from the longer distances.

[6] This is straightforward but the resulting formulas are too complex to include here due to space limitations.

Sensing and Control on the Sphere

Peter Corke and Robert Mahony

Abstract. The advantages of a spherical imaging model are increasingly well recognized within the robotics community. Perhaps less well known is the use of the sphere for attitude estimation, control and scene structure estimation. This paper proposes the sphere as a unifying concept, not just for cameras, but for sensor fusion, estimation and control. We review and summarize relevant work in these areas and illustrate this with relevant simulation examples for spherical visual servoing and scene structure estimation.

1 Introduction

In the last few years there has been growing interest in image processing operations performed on the sphere stemming from a number of important developments. Firstly, a wide perceptual field is important for robotic path planning and collision avoidance and led researchers to adopt, or develop, wide-angle viewing systems [5, 32, 35]. Secondly, the mathematical techniques for spherical imaging have matured, for example the unified imaging model [16], spherical scale-space [3] and spherical SIFT [18]. Thirdly, as purely vision-based navigation becomes possible the ambiguity between rotation and translation which is problematic in a perspective camera image can be overcome by using wide angle imagery. Finally, there is a biological inspiration from small flying insects which use very wide angle eyes, sometimes combined with gyroscopic sensors to perform complex navigation tasks [6].

Pose estimation, including both position and attitude estimation, is a key requirement for autonomous operation of robotic vehicles. For aerial and underwater

Peter Corke
CSIRO ICT Centre
e-mail: peter.corke@csiro.au

Robert Mahony
Australian National University
e-mail: robert.mahony@anu.edu.au

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 71–85.
springerlink.com

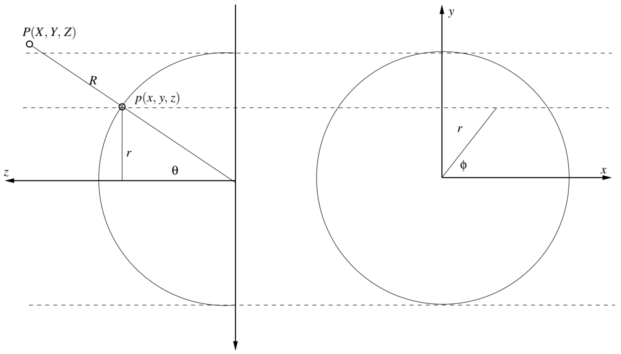


Fig. 1 The coordinate system. P is a world point, mapped to p on the surface of the unit sphere represented by the angles θ and ϕ.

robots, attitude estimation is especially important. Micro-Electro-Mechanical Systems (MEMS) technology has led to a range of low-cost and compact light-weight inertial measurement units that can be used to provide reliable measurements of angular velocity, direction of gravity and altitude. Sensors for magnetic field direction are also available. Good quality, lightweight and low-cost fisheye or catadioptric camera systems are available to provide additional information such as the orientation of vertical edges in the environment [28] or the plane of the horizon [6]. None of these sensors estimate full attitude, in SO(3), directly and it turns out that the attitude estimation problem is best tackled using sphere-based measurements.

Optical flow is an important low-level image cue that encodes both ego-motion and scene structure. We discuss the spherical optical-flow equation and its advantages for detecting key ego-motion parameters such as direction of motion. The optical flow equations can be inverted to create an image-based visual servoing (IBVS) system as well as a structure from motion estimator.

The next section, Section 2, derives the optical flow equation and image Jacobian for the sphere, and then in Section 3 we briefly recall the unified imaging model that can be used to create a spherical image from one or more cameras that could be perspective, fisheye or catadioptric. Section 4 describes sensor fusion on the sphere for the case of attitude and full-pose estimation. Section 5 inverts the optical flow equation to effect control on the spherical image plane and Section 6 outlines the advantages of the sphere for the structure from motion problem.

2 Spherical Optical Flow

As for the case of a perspective camera [21] we assume that the camera is moving with translational velocity $\mathbf{T} = (t_x,\, t_y,\, t_z)$ and angular velocity $\omega = (\omega_x,\, \omega_y,\, \omega_z)$ in the camera frame. A world point, $\mathbf{P}$, with camera relative coordinates ${}^c\mathbf{P} = (X,\, Y,\, Z)^T$ has camera-relative velocity

$$ {}^{c}\dot{\mathbf{P}} = -{}^{c}\omega_e \times {}^{c}\mathbf{P} + {}^{c}\mathbf{T}_e \tag{1}$$

which can be written in scalar form as

$$\dot{x} = z\omega_y - y\omega_z + t_x \tag{2}$$
$$\dot{y} = x\omega_z - z\omega_x + t_y \tag{3}$$
$$\dot{z} = y\omega_x - x\omega_y + t_z \tag{4}$$

The world point P is projected, Figure 1, to p on the surface of the unit sphere

$$x = \frac{X}{R},\ y = \frac{Y}{R}, \text{ and } z = \frac{Z}{R} \tag{5}$$

where the focal point is at the center of the sphere and the radial distance to the point is $R = \sqrt{X^2+Y^2+Z^2}$. The spherical surface constraint $x^2+y^2+z^2 = 1$ means that one of the Cartesian coordinates is redundant, and we will instead use a minimal spherical coordinate system comprising the angle of colatitude

$$\theta = \sin^{-1} r,\ \theta = [0,\ \pi) \tag{6}$$

and the azimuth angle

$$\phi = \tan^{-1}\frac{y}{x},\ \phi = [0,\ 2\pi) \tag{7}$$

yielding the point feature vector $\mathbf{f} = (\theta,\ \phi)$. Other spherical image features are possible such as lines or moments [37].

Taking the derivatives of (6) and (7) with respect to time and substituting (2) – (4) as well as

$$X = Z\tan\theta\cos\phi,\ Y = Z\tan\theta\sin\phi,\ Z = R\cos\theta. \tag{8}$$

we obtain, in matrix form, the spherical optical flow equation

$$\begin{bmatrix}\dot{\theta}\\ \dot{\phi}\end{bmatrix} = \mathbf{J}(\theta,\ \phi, R)\left[t_x\ t_y\ t_z\ \omega_x\ \omega_y\ \omega_z\right]^T \tag{9}$$

where

$$\mathbf{J}(\theta,\phi,R) = \begin{bmatrix} \frac{\cos(\phi)\cos(\theta)}{R(t)} & \frac{\sin(\phi)\cos(\theta)}{R(t)} & -\frac{\sin(\theta)}{R(t)} & \vdots & -\sin(\phi) & \cos(\phi) & 0 \\ -\frac{\sin(\phi)}{R(t)\sin(\theta)} & \frac{\cos(\phi)}{R(t)\sin(\theta)} & 0 & \vdots & -\frac{\cos(\phi)\cos(\theta)}{\sin(\theta)} & -\frac{\sin(\phi)\cos(\theta)}{\sin(\theta)} & 1 \end{bmatrix} \tag{10}$$

is the image feature Jacobian or optical flow equation in terms of the *spherical* point feature $\mathbf{f} = (\theta,\ \phi)$.

There are important similarities to the Jacobian derived for projective cameras in polar coordinates [8,22]. Firstly, the constant elements 0 and 1 fall at the same place, indicating that colatitude is invariant to rotation about the optical axis, that azimuth angle is invariant to optical axis translation, but equal to optical axis rotation. As for

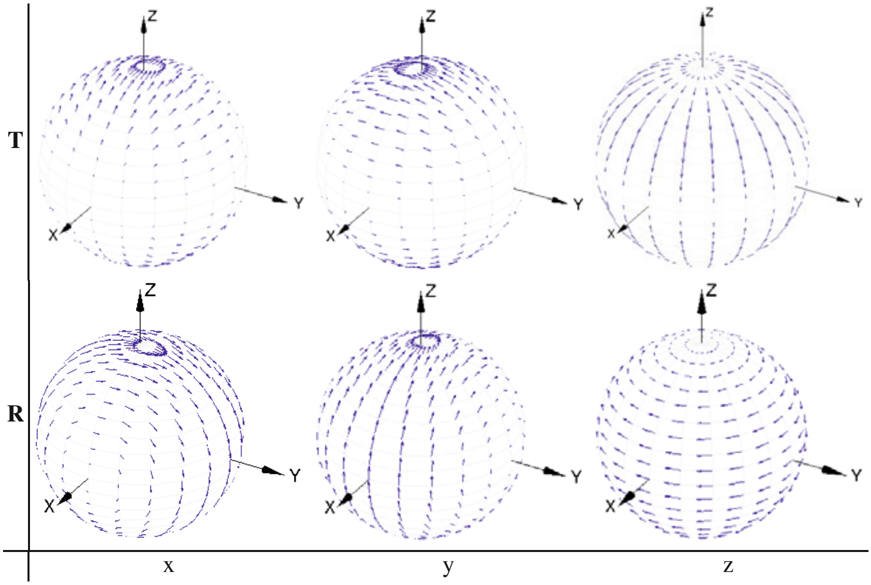


Fig. 2 Spherical optical flow patterns for canonical motion along and about x-, y- and z-axes

all image Jacobians the translational sub-matrix (the first 3 columns) is a function of point depth $1/R$. Note also that the Jacobian is not defined at the poles where $\sin\theta = 0$.

The optical flow patterns on the sphere, for canonical motions, are shown in Figure 2. We note the distinctly different flow patterns for each Cartesian velocity component. A perspective camera with its optical axis aligned with the z-axis has a field of view equivalent to a polar cap, and within that small region of the sphere we can see that the flow due to t_x and ω_y are close to indistinguishable, as are t_y and ω_x. This is a consequence of making a 2-DOF measurement (optical flow) of a 6-DOF phenomena (spatial velocity) leading to a null-space of order 4; in this some motions are linearly dependent (the indistinguishable flows) and some are unobservable (two are null at the pole).

We can also partition the Jacobian [10] into a translational and rotational part

$$\begin{bmatrix} \dot{\theta} \\ \dot{\phi} \end{bmatrix} = \frac{1}{R}\mathbf{J}_t(\theta, \phi) \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} + \mathbf{J}_\omega(\theta, \phi) \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{11}$$

which is important for both control and structure estimation. For points at infinity the first term will be zero.

Optical flow on the sphere can be calculated using a range of approaches, either on the sphere, or on the camera's image plane and mapped to the sphere as discussed in the next section.

3 Mapping Cameras to the Sphere

Recent and growing interest in wide-angle viewing systems [5, 32, 35] has been driven by application need for path planning and collision avoidance and also the availability of new cameras. For instance high-quality glass fisheye lenses with just over a hemispherical field of view have been used by researchers for several years now, and lower-quality low-cost plastic fisheye lenses suitable for small aerial robots have also been investigated [34]. Reasonable quality, lightweight and or catadioptric camera systems are also available.

The unified model of Geyer and Daniilidis [16] provides a convenient framework to consider very different types of cameras such as standard perspective, catadioptric and many types of fisheye lens. The projection model is a two-step process and the key concepts are shown in Figure 3. Firstly, the world point P is projected to the surface of the unit sphere with a focal point at the center of the sphere. The center of the sphere is a distance m from the image plane along its normal z-axis. Secondly the point p is re-projected to the image plane m with a focal point at a distance l along the z-axis, where l is a function of the imaging geometry.

Commonly used mirrors have a parabolic or hyperbolic cross-section, and for these $l = \varepsilon$ the eccentricity of the conic section: $l = 1$ for a parabola and $0 < \varepsilon < 1$ for a hyperbola. This class of mirrors result in a *central* camera with a single effective viewpoint, that is, all rays in space intersect at a single point. Mirrors commonly used in robotics, for example [12, 35], have an equiangular model and the viewpoint lies along a short locus (the *caustic*) within the mirror. In practice this difference in focal point is very small compared to the world scale and such mirrors are well

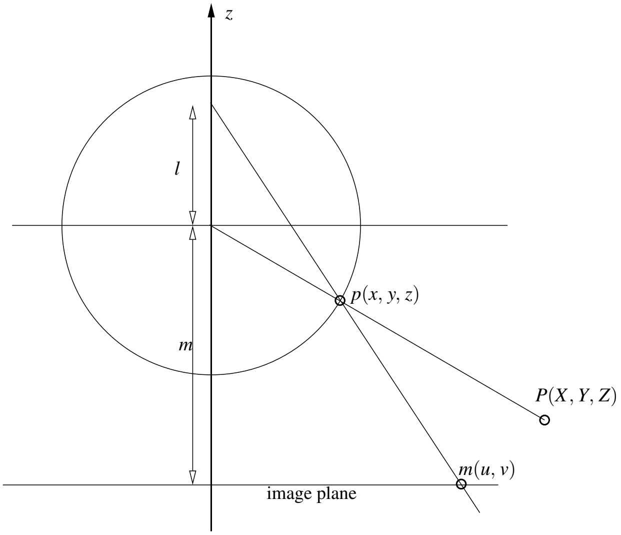


Fig. 3 Unified imaging model of Geyer and Daniilidis [16].

approximated by the central model. Subsequent work [40] showed that many fisheye cameras could also be included in this framework, generally with $l > 1$. For a perspective camera $l = 0$ and the first and second projection rays overlap.

We can invert the unified model and project an image, features or optical flow from a camera image plane (perspective, fisheye or catadioptric) to the sphere for subsequent processing. In [18] catadioptric and fisheye images were projected to the sphere for scale-space computation using a spherical equivalent of the Gaussian kernel. In [34] sparse optical flow was projected to the sphere for focus of expansion detection.

True spherical cameras are under development [25, 29] but until they become a reality we must be content with partial spherical views from a camera, or a mosaic view from multiple cameras (such as the Point Grey Ladybug camera). The spherical framework allows the mapping of multiple cameras with different orientations and different fields of view and projection models to the sphere from their individual image planes, and this is shown schematically in Figure 4 (left).

In practice the various images are not obtained from exactly the same viewpoint, but from slightly offset viewpoints caused by the physical separation of the individual cameras and this leads to parallax error as shown in Figure 4 (right). This is problematic in areas of camera overlap and exacerbated when the inter-camera distances are significant compared to the distance to the point. It can be resolved, that is, the point projected to O if the range to P can be estimated which is a stereo vision problem between the cameras centered at C_1 and C_2.

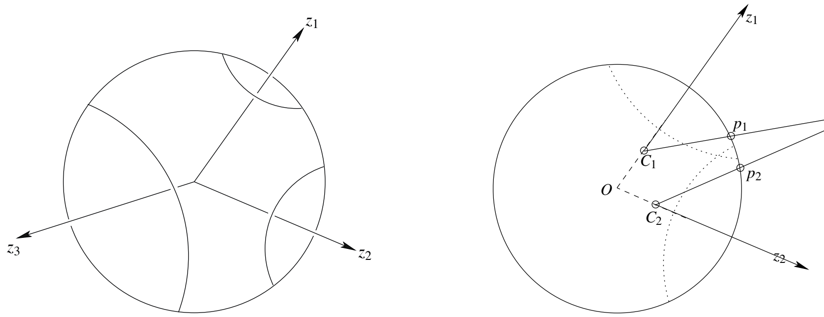


Fig. 4 (left) multiple camera views mapped to the sphere, (right) parallax on the sphere when camera centers, C_1 and C_2 are not coincident with the center of the sphere O.

4 Sensor Fusion on the Sphere

4.1 Attitude Estimation

There is a considerable body of work on attitude reconstruction for robotics and control applications (for example the review [13]). A standard approach is to use extended stochastic linear estimation techniques [26]. An alternative is to use deterministic complementary filter and non-linear observer design techniques [1, 39].

[15]. The recent interest in small low-cost aerial robotic vehicles has lead to a renewed interest in lightweight embedded IMU systems [1, 23, 36]. For the low-cost light-weight systems considered, linear filtering techniques have proved extremely difficult to apply robustly [33] and linear single-input single-output complementary filters are often used in practice [9]. The integration of inertial and visual information is discussed in [11].

Gyroscopes provide an estimate of angular velocity, Ω, corrupted by an offset which leads to drift in attitude after integration. We can constrain the estimate by minimizing the error angle between a reference direction and its estimate on the sphere in the body-fixed-frame. To fully constrain the solution at least two non-collinear reference directions are required. Each of these directions and estimates can be projected to the surface of the sphere as shown in Figure 5 as small circles. One such reference direction can be obtained from the gravity field vector — the the magnitude is irrelevant only its direction on the sphere is required. If the body is accelerating in the inertial frame this vector will be corrupted by inertial forces, however, for steady or hovering flight of a UAV the accelerometers approximately measure the gravitation field in the body-fixed-frame. Another reference direction can be provided by the Earth's magnetic field B at the location. Visual tracking of very distant feature points (at infinity) provide additional reference directions.

The natural framework to interpret these measurement is on the sphere in the body-fixed-frame of the mobile vehicle. Recent work in the development of non-linear observers for attitude estimation provides an ideal framework for the

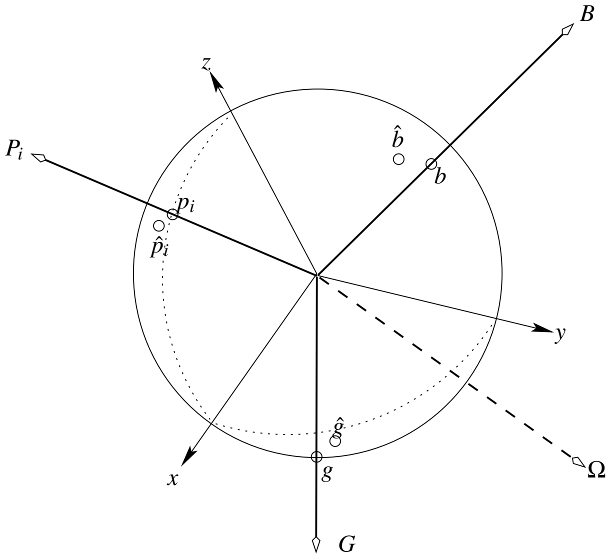


Fig. 5 Sensor integration on the sphere. The attitude of the body is indicated by the x-, y- and z-axes. The predicted and measured sensor vector projections are shown by small circles. Visual landmarks are also included.

construction of robust and efficient attitude filters based on vector measurements on the sphere [30,2,4,38,31].

More formally the attitude estimation problem can be expressed in terms of sphere based measurements [30] using the matrix Lie-group representation of $SO(3)$ — the group of rotations associated with the attitude of a rigid-body. The attitude kinematics is described by

$$\dot{R} = R\,^B\Omega_\times . \tag{12}$$

where $^B\Omega$ is the body-fixed-frame angular velocity of the rigid-body and $[]_\times$ denotes the skew-symmetric matrix. Consider measurements from $n \geq 1$ sensors in the body-fixed frame $a_i \in S^2_B$ $i = 1, 2, \ldots n$ each of which has an associated reference directions in the world frame $a_{0i} \in S^2_W$. These are expressed as vectors on the unit sphere in the inertial frame. Let $\hat{a}_i$ be the estimates of the sensor measurements in the body frame

$$\hat{a}_i = \hat{R}^\top a_{0i} \tag{13}$$

based on the estimated attitude $\hat{R}$.

The complementary observer proposed in [30] is

$$\dot{\hat{R}} = \hat{R}(\Omega_y - \hat{b} + \omega_{\text{mes}})_\times \tag{14}$$

$$\dot{\hat{b}} = -2k_I\omega_{\text{mes}} \tag{15}$$

$$\omega_{\text{mes}} = \sum_{i=1}^{n} k_i(a_i \times \hat{a}_i) \tag{16}$$

where Ω_y is the measured body-fixed-frame angular velocity obtained from the gyroscopes, $\hat{b}$ is the gyroscope bias, and $k_i \geq 0$. It is shown that the attitude estimate $\hat{R}$ will approach the true attitude R asymptotically. For implementation we use quaternions and the following algorithm:

1. Determine ω_{mes} from available sensor measurements according to (13) and (16)
2. Compute the quaternion velocity

$$\dot{\hat{q}} = \frac{1}{2}\hat{q} \otimes \mathbf{p}\left(\Omega_y - \hat{b} + k_P\omega_{\text{mes}}\right)$$

   where $\mathbf{p}$ converts a vector to a pure quaternion and $\otimes$ represents quaternion (Hamiltonian) multiplication.

3. Integrate and renormalize the quaternion

$$\hat{q}_{k+1} = \hat{q}_k + \delta_t\dot{\hat{q}}$$
$$\hat{q}_{k+1} = \hat{q}_{k+1}/||\hat{q}_{k+1}||$$

4. Update the gyroscope bias estimate

$$\hat{b}_{k+1} = \hat{b}_k + \delta_t\dot{\hat{b}}$$

The gains $k_i(t)$ can be set adaptively depending on confidence in particular sensors.

4.2 Pose Estimation

Pose estimation, including both position and attitude estimation, is a key requirement for autonomous operation of robotic vehicles, especially aerial robots. Modern Global Positioning Systems (GPS) are have decreasing cost, weight, and energy consumption while increasing quality and functionality. Carrier phase doppler shift can be extracted from GPS signals to provide inertial frame velocity estimates of a vehicle's motion [19] and rotated into the body-fixed-frame based on an attitude estimate.

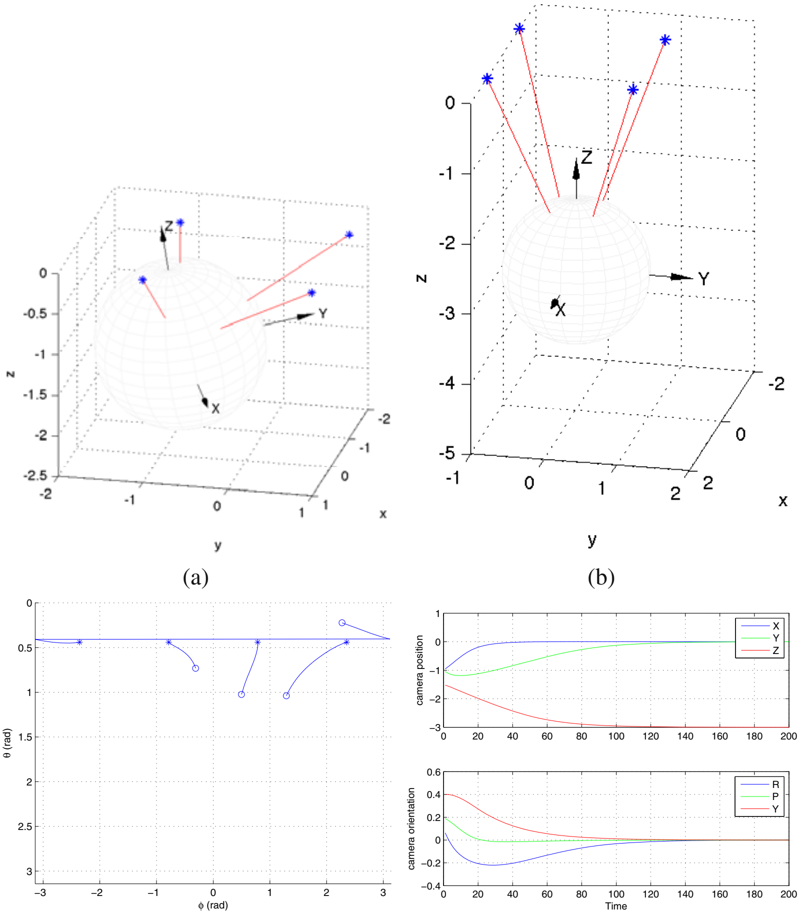
In the absence of GPS, due to denial or a multi-pathing environment, we can extract information about translational motion from the optical flow field. If optical flow due to rotational motion is estimated and subtracted, a process known as *derotation*, the remaining optical flow is due only to translation, corrupted with noise from the optical flow process itself and errors in the rotational estimate. A characteristic of the translational flow field is a focus of expansion on the sphere at the point defined by the vector $\mathbf{d}$ where $\mathbf{d} = \mathbf{t}/|\mathbf{t}|$. As discussed above the optical flow encodes translational motion with a scale uncertainty, that is $\mathbf{t}/R$ not $\mathbf{t}$ can be identified. Expressed another way, the direction of motion is a unit vector through the focus of expansion on the sphere. A corresponding focus of contraction will be found at the antipodal point. Recent approaches to determining the focus of expansion from optical flow are [27].

5 Control on the Sphere

Visual servoing is the use of information from one or more cameras to guide a robot so as to achieve a task [7, 21]. Image-Based visual servoing (IBVS) is a robust and efficient technique where the task is defined in terms of the desired view of the target and a control law is synthesized to move the camera toward that view. The goal is defined *implicitly* in the desired view. The pose of the target does not need to be known apriori, the robot moves toward the observed target wherever it might be in the workspace. Image-based control can be considered as an inverse problem to optical flow — given a current and desired view the required optical flow can be computed, the problem is to determine the motion in SE(3) to achieve it. Most of the visual servoing literature is concerned with perspective cameras and a Cartesian image plane. More recently polar coordinates have been used with perspective cameras, and spherical cameras [17, 22].

For control purposes we follow the normal procedure of computing one 2×6 Jacobian, (9), for each of N feature points and stacking them to form a $2N \times 6$ matrix

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\phi}_1 \\ \vdots \\ \dot{\theta}_N \\ \dot{\phi}_N \end{bmatrix} = \begin{bmatrix} \mathbf{J}_{\mathbf{p}1} \\ \vdots \\ \mathbf{J}_{\mathbf{p}N} \end{bmatrix} \mathbf{v} \tag{17}$$



(a) (b)

(c) (d)

Fig. 6 IBVS on the sphere. (a) initial pose, (b) final pose, (c) feature motion on the θ-ϕ plane, (d) pose evolution.

The control law is

$$\mathbf{v} = \mathbf{J}^{+} \dot{f}^{*} \tag{18}$$

where $\dot{f}^{*}$ is the desired velocity of the features. Typically we choose this to be proportional to feature error

$$\dot{f}^{*} = -\gamma(f \ominus f^{*}) \tag{19}$$

where γ is a positive gain, f is the current value of the feature vector, and f^{*} is the desired value, which leads to linear motion of features within the feature space.

$\ominus$ denotes modulo subtraction giving the smallest angular distance given that $\theta = [0, \pi)$ and $\phi = [-\pi, \pi)$.

Figure 6 presents simulation results for spherical IBVS for the case of general motion with translation and rotation in all axes. The target consists of four points arranged in the plane, and the servo task is to move to a pose where the camera's z-axis is normal to the plane and 3 m away. We can see that the velocity demand is well behaved and that the features have followed direct paths in the feature space. One feature has wrapped around in the azimuth direction.

If the attitude was servoed by a non-visual sensor such as gyroscope, accelerometer or magnetometer then we could use a partitioned IBVS scheme [10] where we would write (11) as

$$\frac{1}{R}\mathbf{J}_t(\theta,\phi)\begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} = \begin{bmatrix} \dot{\theta} \\ \dot{\phi} \end{bmatrix} - \mathbf{J}_\omega(\theta,\phi)\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{20}$$

and solve for translational velocity.

Classical visual servo control, as just described was principally developed for serial-link robotic manipulators [21] where all camera degrees of freedom are actuated. The dynamics of the system are easily compensated using a computed torque (or high gain) control design and the visual servo control may be derived from a first order model of the image dynamics [14]. Recent applications in high performance systems and under-actuated dynamic systems have lead researchers to consider full dynamic control design. Coupling of the camera ego-motion dynamics in the image plane proves to be a significant obstacle in achieving this goal and [24] proposed an asymptotically stable method for position regulation for fixed-camera visual servoing for a dynamic system. A further complication is encountered when an under-actuated dynamic system is used as the platform for the camera and [41] used a Lagrangian representation of the system dynamics to obtain an IBVS control for a blimp, an under-actuated, non-holonomic system.

An IBVS controller for a class of under-actuated dynamic systems has been proposed [17]that does not require accurate depth information for image features. Exploiting passivity they observe that

> *the only image geometry that preserves the passivity-like properties of the body fixed frame dynamics of a rigid object in the image space are those of a spherical camera.*

6 Structure and Motion Estimation on the Sphere

In the IBVS example of the previous section the values of R required to compute the image Jacobian were taken from the simulation engine, but in a real system they would not be known. Experience with IBVS shows that it is quite robust to errors in point depth, R, and typically the final depth value is assumed throughout the motion.

The point depth can also be estimated, by rewriting (11) in identification form as

$$\left(\mathbf{J}_t(\theta, \phi) \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \right) (1/R) = \begin{bmatrix} \dot{\theta} \\ \dot{\phi} \end{bmatrix} - \mathbf{J}_\omega(\theta, \phi) \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} \tag{21}$$

or

$$\mathbf{A}\theta = \mathbf{b} \tag{22}$$

where the camera motion $(T_x, T_y, T_z, \omega_x, \omega_y, \omega_z)$ is known, since IBVS commands it, and $(\dot{\theta}, \dot{\phi})$ is the optical flow which is observed during the motion. This a spherical structure from motion (SfM) system [20]. For the example of the previous section, the results of this estimator are shown in Figure 7 for one of the four target points. This is a very cheap estimator but we could also use a Kalman filter as is often used in structure from motion systems. Depth could be estimated for all tracked points in the scene, not just those used for servoing.

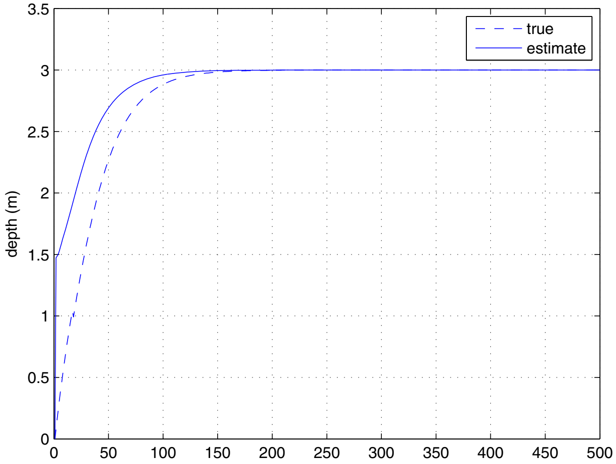


Fig. 7 Comparison of estimated and true point depth.

This leads to our final use for the sphere, the scene structure can be conveniently considered in camera-centric form as a spherical depth map $D : \mathbb{S}^2 \rightarrow \mathbb{R}$. This form makes it trivially easy to handle camera rotational motion, the depth map rolls around the sphere. For translational motion of the camera points move over the sphere according to the direction of translation and point depth. We believe that this structure is amenable to a particle filter, operating on the sphere, which is able to better model the non-Gaussian distribution of depth uncertainty.

7 Conclusions

In this paper we have presented the sphere as a unifying concept, not just for cameras, but for sensor fusion, estimation and control. For robots that move in $SE(3)$ such as UAVs or AUVs the advantages of the sphere are particularly strong.

The unified imaging model is of Geyer and Daniilidas [16] is well known and uses spherical projection to model catadioptric cameras with mirrors which are conics. However the model is also an excellent approximation for other shaped mirrors such as equiangular and can also model many fisheye lenses. This is convenient for robotics where the advantages of wide-angle cameras include wide perceptual field for path planning and collision avoidance and resolving the ambiguity between rotation and translation which is problematic in a perspective cameras. The unified model also provides a framework for mosaic creation through projection from camera image planes, and image processing operations such as scale-space feature detectors can be formulated on the sphere. Camera ego-motion results in the apparent motion of points, optical flow, in the spherical image. Points on the surface of a sphere are commonly described by a unit 3-vector but this description is redundant and we presented a formulation of the optical flow equation in terms of two angles: colatitude and azimuth. spherical imaging model are increasingly well known.

The problem of attitude estimation is very naturally treated on the sphere using the matrix Lie-group representation of $SO(3)$. An asymptotically stable estimator that can incorporate multiple sensors such as gyroscopes, magnetometers, accelerometers and visual landmarks is presented. Inverting the optical flow equation leads to a spherical image-based visual servoing scheme which exhibits the desirable properties of the more well-known planar IBVS scheme such as robustness to depth uncertainty, and like the polar-planar IBVS performs well for large rotational motions. Adding a simple state estimator based on the inverted optical flow equation leads to a structure from motion solution, and the consideration of a camera-centric spherical depth map as a convenient world representation.

References

1. Baerveldt, A.J., Klang, R.: A low-cost and low-weight attitude estimation system for an autonomous helicopter. Intelligent Engineering Systems (1997)
2. Bonnabel, S., Martin, P., Rouchon, P.: Invariant asymptotic observers. IEEE Transactions on Automatic Control (2008), http://arxiv.org/abs/math.OC/0612193 (accepted for publication)
3. Bülow, T.: Spherical diffusion for 3D surface smoothing. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(12), 1650–1654 (2004)
4. Campoloa, D., Kellerb, F., Guglielmellia, E.: Inertial/magnetic sensors based orientation tracking on the group of rigid body rotations with application to wearable devices. In: Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, Beijing, China, pp. 4762–4767 (2006)
5. Chahl, J.S., Srinivasan, M.V.: Reflective surfaces for panoramic imaging. Applied Optics 31(36), 8275–8285 (1997)
6. Chahl, J.S., Thakoor, S., Bouffant, N.L., Stange, G., Srinivasan, M.V., Hine, B., Zornetzer, S.: Bioinspired engineering of exploration systems: A horizon sensor/attitude reference system based on the dragonfly ocelli for mars exploration applications. J. Field Robotics 20(1), 35–42 (2003)
7. Chaumette, F., Hutchinson, S.: Visual servo control. i. basic approaches. IEEE Robotics & Automation Magazine 13(4), 82–90 (2006), doi:10.1109/MRA.2006.250573

8. Chaumette, F., Hutchinson, S.: Visual servo control. ii. advanced approaches [tutorial]. IEEE Robotics & Automation Magazine 14(1), 109–118 (2007), doi:10.1109/MRA.2007.339609
9. Corke, P.: An inertial and visual sensing system for a small autonomous helicopter. J. Robotic Systems 21(2), 43–51 (2004)
10. Corke, P., Hutchinson, S.A.: A new partitioned approach to image-based visual servo control. IEEE Trans. Robot. Autom. 17(4), 507–515 (2001)
11. Corke, P., Lobo, J., Dias, J.: An introduction to inertial and visual sensing. Int. J. Robotics Research 26(6), 519–536 (2007)
12. Corke, P., Strelow, D., Singh, S.: Omnidirectional visual odometry for a planetary rover. In: Proceedings International Conference on Intelligent Robots and Systems, pp. 4007–4012 (2004)
13. Crassidis, J.L., Markley, F.L., Cheng, Y.: Nonlinear attitude filtering methods. Journal of Guidance, Control,and Dynamics 30(1), 12–28 (2007)
14. Espiau, B., Chaumette, F., Rives, P.: A new approach to visual servoing in robotics. IEEE Transactions on Robotics and Automation 8(3), 313–326 (1992)
15. Gebre-Egziabher, D., Hayward, R.C., Powell, J.D.: Design of multi-sensor attitude determination systems. IEEE Transactions on Aerospace and Electronic Systems 40(2), 627–649 (2004)
16. Geyer, C., Daniilidis, K.: A unifying theory for central panoramic systems and practical implications. In: Vernon, D. (ed.) ECCV 2000. LNCS, vol. 1843, pp. 445–461. Springer, Heidelberg (2000)
17. Hamel, T., Mahony, R.: Visual servoing of an under-actuated dynamic rigid-body system: An image based approach. IEEE Transactions on Robotics and Automation 18(2), 187–198 (2002)
18. Hansen, P., Boles, W., Corke, P.: Spherical diffusion for scale-invariant keypoint detection in wide-angle images. In: DICTA 2008: Proceedings of the 2008 Digital Image Computing: Techniques and Applications, pp. 525–532 (2008)
19. Hatch, R.: Synergism of gps code and carrier measurements. In: Proceedings of the 3rd International Geodetic Sypomsium on Satellite Doppler Positioning, Las Cruces, New Mexico, vol. 2, pp. 1213–1232 (1982)
20. Huang, T., Netravali, A.: Motion and structure from feature correspondences: a review. Proceedings of the IEEE 82(2), 252–268 (1994), doi:10.1109/5.265351
21. Hutchinson, S., Hager, G., Corke, P.: A tutorial on visual servo control. IEEE Transactions on Robotics and Automation 12(5), 651–670 (1996)
22. Iwatsuki, M., Okiyama, N.: A new formulation of visual servoing based on cylindrical coordinate system with shiftable origin. In: IEEE/RSJ International Conference on Intelligent Robots and System, vol. 1, pp. 354–359 (2002), doi:10.1109/IRDS.2002.1041414
23. Jun, M., Roumeliotis, S., Sukhatme, G.: State estimation of an autonomous helicopter using Kalman filtering. In: Proc. 1999 IEEE/RSJ International Conference on Robots and Systems, IROS 1999 (1999)
24. Kelly, R.: Robust asymptotically stable visual servoing of planar robots. IEEE Transactions on Robotics and Automation 12(5), 759–766 (1996)
25. Krishnan, G., Nayar, S.K.: Towards a true spherical camera. In: Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, (SPIE) Conference Series, vol. 7240 (2009), doi:10.1117/12.817149
26. Lefferts, E., Markley, F., Shuster, M.: Kalman filtering for spacecraft attitude estimation. AIAA Journal of Guidance, Control and Navigation 5(5), 417–429 (1982)
27. Lim, J., Barnes, N.: Directions of egomotion from antipodal points. In: CVPR. IEEE Computer Society Press, Los Alamitos (2008)

28. Lobo, J., Dias, J.: Vision and inertial sensor cooperation using gravity as a vertical reference. IEEE Transactions on Pattern Analysis and Machine Intelligence 25(12), 1597–1608 (2003)
29. Maddern, W., Wyeth, G.: Development of a Hemispherical Compound Eye for Egomotion Estimation. In: Australasian Conference on Robotics & Automation (2008)
30. Mahony, R., Hamel, T., Pflimlin, J.M.: Non-linear complementary filters on the special orthogonal group. IEEE Transactions on Automatic Control 53(5), 1203–1218 (2008), doi:10.1109/TAC.2008.923738
31. Martin, P., Salaün, E.: An invariant observer for earth-velocity-aided attitude heading reference systems. In: Proceedings of the 17th World Congress The International Federation of Automatic Control, Seoul, Korea (2008)
32. Nayar, S.K.: Catadioptric omnidirectional camera. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, p. 482 (1997), http://doi.ieeecomputersociety.org/10.1109/CVPR.1997.609369
33. Roberts, J.M., Corke, P.I., Buskey, G.: Low-cost flight control system for small autonomous helicopter. In: Australian Conference on Robotics and Automation, Auckland, November 27-29, pp. 71–76 (2002)
34. Schill, F., Mahony, R., Corke, P., Cole, L.: Virtual force feedback teleoperation of the insectbot using optical flow. In: Kim, J., Mahony, R. (eds.) Proc. Australian Conf. Robotics and Automation (2008), http://www.araa.asn.au/acra/acra2008/papers/pap159s1.pdf
35. Strelow, D., Mishler, J., Koes, D., Singh, S.: Precise omnidirectional camera calibration. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, p. 689 (2001), http://doi.ieeecomputersociety.org/10.1109/CVPR.2001.990542, doi:10.1109/CVPR.2001.990542
36. Sukhatme, G.S., Roumeliotis, S.I.: State estimation via sensor modeling for helicopter control using an indirect Kalman filter. In: Int.Conf. Intelligent Robotics, IROS (1999)
37. Tahri, O., Mezouar, Y., Chaumette, F., Corke, P.: Generic decoupled image-based visual servoing for cameras obeying the unified projection model. In: IEEE International Conference on Robotics and Automation (ICRA 2009), pp. 1116–1121 (2009), doi:10.1109/ROBOT.2009.5152359
38. Vasconcelos, J., Silvestre, C., Oliveira, P.: A nonlinear observer for rigid body attitude estimation using vector observations. In: Proceedings of the 17th World Congress The International Federation of Automatic Control, Seoul, Korea, July 2008, pp. 8599–8604 (2008)
39. Vik, B., Fossen, T.: A nonlinear observer for GPS and INS integration. In: Proc. IEEE Conf. on Decisioin and Control, pp. 2956–2961 (2001)
40. Ying, X., Hu, Z.: Can we consider central catiodioptric cameras and fisheye cameras within a unified imaging model. In: Pajdla, T., Matas, J(G.) (eds.) ECCV 2004. LNCS, vol. 3021, pp. 442–455. Springer, Heidelberg (2004)
41. Zhang, H., Ostrowski, J.P.: Visual servoing with dynamics: Control of an unmanned blimp. In: Proceedings of the IEEE Internation Conference on Robotics and Automation, Detroit, Michigan, U.S.A., pp. 618–623 (1999)

Estimating Ego-Motion in Panoramic Image Sequences with Inertial Measurements

Felix Schill, Robert Mahony, and Peter Corke

Abstract. This paper considers the problem of estimating the focus of expansion of optical flow fields from panoramic image sequences due to ego-motion of the camera. The focus of expansion provides a measurement of the direction of motion of the vehicle that is a key requirement for implementing obstacle avoidance algorithms. We propose a two stage approach to this problem. Firstly, external angular rotation measurements provided by an on-board inertial measurement unit are used to de-rotate the observed optic flow field. Then a robust statistical method is applied to provide an estimate of the focus of expansion as well as a selection of inlier data points associated with the hypothesis. This is followed by a least squares minimisation, utilising only the inlier data, that provides accurate estimates of residual angular rotation and focus of expansion of the flow. The least squares optimisation is solved using a geometric Newton algorithm. For the robust estimator we consider and compare RANSAC and a *k*-means algorithm. The approach in this paper does not require explicit features, and can be applied to patchy, noisy sparse optic flow fields. The approach is demonstrated in simulations and on video data obtained from an aerial robot equipped with panoramic cameras.

1 Introduction

The estimation of the ego-motion of a camera from observation of a sequence of images is a classical problem in the computer vision literature. Algorithms based on eight or more point correspondences [18, 10] are well known, while for calibrated

Felix Schill · Robert Mahony
Department of Engineering, Australian National University, ACT, 0200, Australia
e-mail: {firstname.lastname}@anu.edu.au

Peter Corke
CSIRO ICT Centre, Pullenvale, Queensland, Australia
e-mail: peter.corke@csiro.au

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 87–101.
springerlink.com

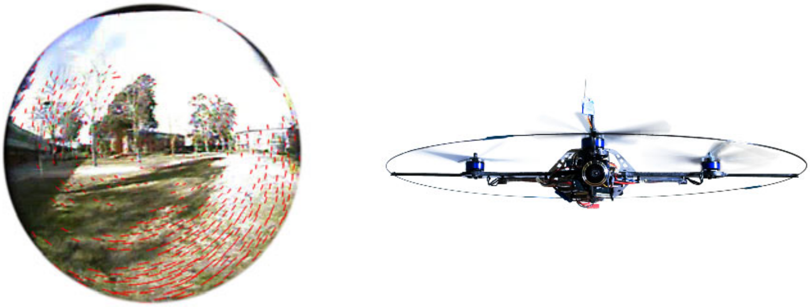


Fig. 1 Video image and optic flow (left) extracted from on-board video camera of the *Hummingbird* quad-rotor aerial robot (right)

cameras algorithms exist for as few as five point correspondences [15, 22]. In the case where high accuracy is required the bundle adjustment method can be used [26]. In addition to the classical methods, there have been a wide range of other methods considered in the literature [14, 23, 12, 11, 3, 21].

It is well known that for an image sequence with a small field-of-view it is difficult to distinguish between translation and rotation around orthogonal axes [2, 7]. In addition, there is is often a natural bias in solving the instantaneous epipolar constraint, the most common approach to recovering instantaneous motion, associated with grouping of image points in one small area [7]. Using panoramic or catadioptric cameras with a large field-of-view can substantially overcome this issue [7]. Due to the inherent ambiguity in velocity there have been a number of studies based on qualitative analysis of ego-motion methods [5, 6, 25] that utilise panoramic cameras, however, these methods often use explicit search routines to determine the best motion fit and are computationally expensive. In recent work Lim and Barnes have developed methods to compute ego-motion from antipodal pairs of optic flow vectors [16, 17]. Almost all the literature in this area has been developed based on the assumption that the camera is the only sensor. In robotic applications, especially those involving aerial robots, there is almost always an inertial measurement unit (IMU) on the vehicle that can provide a substantially correct estimate of rotation over short periods. However, the vision system for such applications often has poor quality optics, and if the video signal is being transmitted to ground there are artifacts due to signal interference. The authors know of no prior work that addresses the specific issues associated with ego-motion extraction for such a situation.

In this paper, we propose an algorithm for extracting ego-motion from a panoramic image sequence where the angular velocity can be measured using a separate sensor. We are primarily motivated by applications in aerial robotics where the vehicle is equipped with a wide angle fish-eye (or catadioptric) lens and inertial sensors. The vision sequences obtained from such vehicles often contain large regions where there is insufficient texture to generate optic flow, for example, regions of sky, or regions distant from the camera where the optics are of insufficient quality to generate good texture. This can make it difficult or impossible to find antipodal points. In addition, there are often extreme outliers in the optic flow field caused by errors in the optic flow algorithm induced by artifacts in the video due to signal interference and multi-path effects in video transmission. We propose a two stage approach. Firstly, optic flow is computed from the image sequence and then this flow is approximately de-rotated using the data from the gyroscopes (which are components of the vehicle's inertial measurement unit). This is achieved by subtracting the expected rotational optic flow (due to the measured angular velocity) from the measured optical flow. The resulting flow is almost entirely due to the translational ego-motion of the camera, except for errors and noise in IMU measurements and flow extraction, and has a simple structure that allows us to develop simple models to determine the unique focus of expansion (FoE), corresponding to the direction of motion of the camera. The focus of expansion estimate provides valuable information for object avoidance and vehicle guidance [20, 24].

We investigate two robust statistical methods, RANSAC and *k*-means, aimed at generating a reasonable hypothesis of the FoE of the flow and identify inlier and outlier optic flow measurements in the data. We then describe how an initial estimate of the focus of expansion can be refined in both translation and residual rotation by minimising a least squares cost based on the instantaneous epipolar condition posed on the sphere. We compute the geometric gradient and geometric Hessian and propose a Newton update step. The Newton minimisation is embedded in the RANSAC framework as the second model refinement step for each iteration. Given that the initial estimate provided by the first stage of the algorithm is moderately correct, this stage usually converges in at most three iterations. Moreover, the eigenvalues of the Hessian provide a measure of confidence in the estimate. A poor condition number for the Hessian indicates the likelihood of an unreliable estimate and the overall magnitude of eigenvalues is proportional to the distance scaled velocity of the vehicle. A common problem with Newton algorithms is poor convergence for badly chosen starting values. The presented problem is especially sensitive to rotation, however by using the rotation measurements from the onboard inertial sensors a reasonably close starting value can be computed.

The proposed algorithms are tested on synthetic data with outliers and noise, and demonstrated on video and inertial data obtained on a small aerial robotic vehicle.

2 Problem Formulation

In this section, we introduce some notation and develop the cost function that will be used to refine ego-motion estimates on the sphere.

We are interested in applications where there is a wide field of view fish eye or catadioptric video camera moving through a static world environment. We assume that the camera frame rate is fast compared to the relative optical velocity of the observed scene, i.e. pixel displacements are sufficiently small that perspective changes and lens distortions will not change significantly. Consequently the optic flow can be computed directly on the raw image sequence and the resulting flow vectors mapped back onto a spherical image plane based on a known calibration of the camera. The spherical optical flow field is denoted Φ and associates a flow vector $\Phi(\eta) \in T_\eta S^2$ in the tangent space of the sphere to a point $\eta \in S^2$ on the sphere. In practice, we are normally constrained to a sparse computation of optic flow, that is measurements at a finite number of points on the sphere indexed by $\{\eta_i\}$ for $i = 1, \ldots, n$, with n the number of optic flow vectors measured in a given reference image.

The optic flow can be split into translational and rotational components

$$\Phi(\eta) = \Psi(\eta) + \Theta(\eta). \tag{1}$$

Here $\Theta(\eta) := -\Omega \times \eta$, with $\Omega \in \{B\}$ the body-fixed frame angular velocity of the camera, is the contribution to the optic flow from the rotational motion of the camera, while the translation component of flow is given by $\Psi(\eta) := \frac{1}{\lambda(\eta)} \mathbb{P}_\eta v$, where $\mathbb{P}_\eta = (I_3 - \eta\eta^\top)$ is the projector onto $T_\eta S^2$ and $v \in \{B\}$ is the body fixed frame translational velocity of the vehicle.

In order to derive an optimisation-based method for identification of ego motion it is necessary to define a cost function. We propose to use a modified version of the instantaneous epipolar constraint. Let $\hat{w}$ denote the estimate of $w \in S^2$ of the true direction of motion of the vehicle. That is, set

$$w = \frac{v}{|v|}, \quad \text{for } v \neq 0,$$

and $\hat{w} \in S^2$ an estimate of $w \in S^2$. The direction of motion w is also the focus of expansion (FoE) of the translational flow field Ψ on the sphere.

For each individual optic flow vector $\Phi(\eta)$ measured at a point $\eta \in S^2$ the instantaneous epipolar constraint computed for estimates $\hat{w}$ and $\hat{\Omega}$ is

$$e_{\Phi(\eta)}(\hat{w}, \hat{\Omega}) := \langle \hat{w}, \big(\Phi(\eta) + \hat{\Omega} \times \eta\big) \times \eta \rangle. \tag{2}$$

Note that if $\hat{\Omega}$ is correct then $\Phi(\eta) + \hat{\Omega} \times \eta = \Psi(\eta)$ is the true translational optic flow. Taking the vector product of this with its base point η leads to a vector that is orthogonal to the direction of motion of the vehicle. Taking the inner product of the resulting vector with $\hat{w}$ is zero precisely when $\hat{w} = w$ is the true direction of motion. The instantaneous epipolar constraint is often written $\langle \hat{w} \times \eta, \big(\Phi(\eta) + \hat{\Omega} \times \eta\big) \rangle$, however, this can be transformed into (the negative of) Equation (2) using the properties of vector triple products and the form given above is more convenient for the gradient and Hessian computations undertaken later.

Since the optic flow is measured at a finite number of scattered points the cost considered is a sum

$$f(\hat{w}, \hat{\Omega}) := \sum_{i=1}^{n} e^2_{\Phi}(\eta)(\hat{w}, \hat{\Omega}) = \sum_{i=1}^{n} \langle \hat{w}, \big(\Phi(\eta) + \hat{\Omega} \times \eta\big) \times \eta \rangle^2 \tag{3}$$

It is clear that for ideal data the cost f is zero for the correct for $\hat{w} = w$ and $\hat{\Omega} = \Omega$. The cost f is a smooth function $f : S^2 \times \mathbb{R}^3 \to \mathbb{R}$ and can be optimised on this set using geometric concepts. A weakness of the cost proposed is that it is highly susceptible to perturbation by large magnitude outliers in the data. Optic flow algorithms often yield exactly this sort of error due to occasional mismatched point correspondences. Thus, direct minimisation of the cost f is likely to lead to poor ego-motion estimation. The next section applies robust statistical algorithms to overcome this issue.

3 Robust Estimation of Focus of Expansion

In this section we present two robust statistical methods for providing an estimate of focus of expansion of the image sequence. The approach is directly based on the application domain and we assume that a measurement of angular velocity of the vehicle is available, for example through the inertial measurement unit that is mounted on the aerial vehicles that we consider. As both the camera and the inertial measurement unit are mounted on the vehicle we will use the vehicle coordinate frame for all calculations. Direction estimates can easily be transferred into a world frame by applying a rotation that can be obtained from a complementary attitude filter [19][4].

Using the measured angular velocity, $\Omega_y \in \{B\}$, the measured optic flow can be de-rotated.

$$\Psi_{\Omega_y}(\eta_i) := \Phi(\eta_i) - \Theta_{\Omega_y}(\eta_i) = \Phi(\eta_i) + \Omega_y \times \eta_i.$$

The resulting estimate, $\Psi_{\Omega_y}(\eta_i)$, of translational flow is only defined at measured flow points η_i.

Any two measurements of translational flow $\Psi_{\Omega_y}(\eta_i)$ and $\Psi_{\Omega_y}(\eta_j)$ can be used to generate a hypothesis for the focus of expansion of the flow field. Since the flow vectors must lie in a plane containing the flow vector and the base point η then the intersection of these two planes provides an estimate of the focus of expansion for the flow field. Thus,

$$\hat{w}(\Omega_y, \eta_i, \eta_j) := \frac{(\Psi_{\Omega_y}(\eta_i) \times \eta_i) \times (\Psi_{\Omega_y}(\eta_j) \times \eta_j)}{|(\Psi_{\Omega_y}(\eta_i) \times \eta_i) \times (\Psi_{\Omega_y}(\eta_j) \times \eta_j)|}. \tag{4}$$

This hypothesis is sign indeterminate so we introduce a corrected hypothesis $\hat{w}_c$ is

$$\hat{w}_c(\Omega_y, \eta_i, \eta_j) := \hat{w}(\Omega_y, \eta_i, \eta_j) \cdot \mathrm{sign}(\langle \hat{w}, \Phi(\eta_1) \rangle) \tag{5}$$

There are potentially $n(n-1)/2$ (where n is the number of flow vectors measured) hypotheses that can be generated based on (4).

The k-means algorithm is applied by collecting a large number of of hypotheses and then clustering these hypotheses into classes. k-means clustering is a standard algorithm [9, 13] that is extensively used in the field of computer vision and we will not cover the details of the algorithm implementation. It is also possible to use different clustering algorithms. The distance between hypotheses for clustering purposes is the angle between two hypotheses. The estimate of the focus of expansion is provided by the centroid (the normalised mean) of the largest cluster of hypotheses,

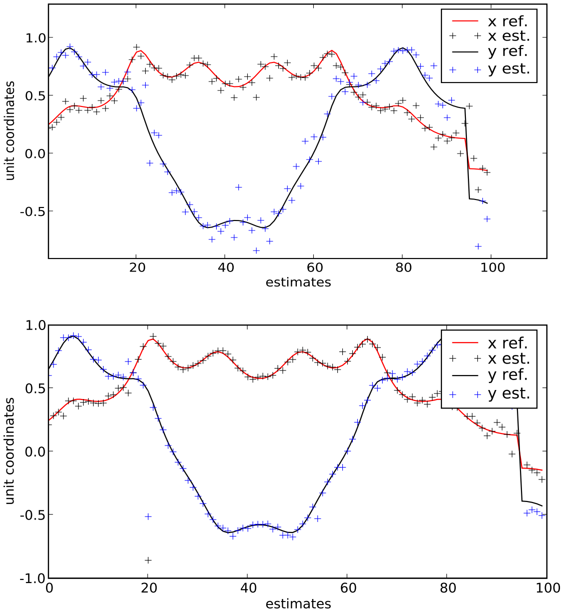


Fig. 2 Estimation results for *k*-means (top plot) and RANSAC (bottom plot), for synthetic flow, 30% outliers and 0.001 Gaussian noise (reference signal is the normalised true translation from the simulation) Note the significantly better performance of the RANSAC algorithm due to the Newton iteration step.

which also forms the inlier set. A Newton algorithm (described in Section 4) can be applied to the support set of the best cluster from the *k*-means algorithm to refine the best estimate, however it was found that this would often not converge if there are outliers present (see section 5). The described *k*-means algorithm can only generate an estimate for the focus of expansion, but not the residual rotation, and therefore relies on adequate removal of rotational flow.

To overcome these limitations, we merged the hypothesis generation of the *k*-means approach and a 6 DOF Newton model refinement into a more robust RANSAC framework. The RANSAC algorithm is based on consensus scoring of hypotheses and once again is a well known algorithm extensively used in computer vision applications [8, 10]. The algorithm is applied by randomly selecting pairs of flow vectors and generating hypotheses according to Equation 5 and using a normalised mean of the hypotheses generated. All flow vectors are then scored with regard to that hypothesis using the cost function (2) to determine the inlier or consensus set, consisting of all flow vectors where $e_{\Phi(\eta)} < t$ (t is the acceptance threshold). The hypothesis along with its consensus set is then used to initialise the

Newton iteration discussed in the next section and a refined estimate of both FoE and angular velocity is computed. Over several iterations, the refined estimate with the smallest residual is chosen as the best estimate from the algorithm.

4 Refining the Motion Estimate

In this section we present a geometric Newton algorithm that can be used to efficiently refine the estimates of ego-motion based on minimising the cost (3). The Newton algorithm requires a reasonable estimate of the local minima and identification of inlier flow vectors to provide a reliable estimate of ego-motion. The implementation of the Newton method also needs to respect the unit norm constraint on the focus of expansion estimate $\hat{w}$ in the optimisation problem. We achieve this by deriving the Newton algorithm with respect to the geometry of the constraint set. Details on geometric optimisation algorithms can be found in Absil *et al.* [1].

For the sake of simplifying notation we define

$$Z(\hat{\Omega}) := \big(\Phi(\eta) + \hat{\Omega} \times \eta\big) \times \eta. \tag{6}$$

The geometric gradient of f is an element of $T_{(\hat{w},\hat{\Omega})}S^2 \times \mathbb{R}^3$. It is obtained by differentiating $f(\hat{w},\hat{\Omega})$ in an arbitrary direction and then using the natural Riemannian metric to obtain a tangent vector;

$$\operatorname{grad} f(\hat{w},\hat{\Omega}) = \begin{pmatrix} \mathbb{P}_{\hat{w}} \left(\sum_{i=1}^{n} Z(\Omega) Z^\top(\Omega)\right) \hat{w} \\ -\sum_{i=1}^{n} \left((\hat{w}^\top Z(\Omega))\mathbb{P}_{\eta_i}\hat{w}\right) \end{pmatrix}, \tag{7}$$

recalling that $\mathbb{P}_v = I_3 - vv^\top$ is the projection onto $T_v S^2$.

It is possible to consider a gradient descent method to optimise the cost function f defined in (3). However, due to the inherent nature of the data, the cost function is several orders of magnitude more sensitive to change in the angular velocity estimate than the FoE estimate, leading to very poor convergence of naive gradient descent algorithms. In practice, effective implementation of a gradient descent algorithm would require preconditioning of the gradient. Since an initial guess of the local minimum is available from the k-means or RANSAC algorithm (Section 3) it is possible to overcome this difficulty by using a Newton algorithm directly.

The geometric Hessian for f can be written

$$\operatorname{Hess} f(\hat{w},\hat{\Omega}) = \tag{8}$$
$$\begin{pmatrix} \mathbb{P}_{\hat{w}} \left(\sum\limits_{i=1}^{n} Z(\Omega) Z(\Omega)^\top\right) \mathbb{P}_{\hat{w}} & -\mathbb{P}_{\hat{w}} \sum\limits_{i=1}^{n} \left((\hat{w}^\top Z(\Omega))\mathbb{P}_{\eta_i} + Z(\Omega)\hat{w}^\top \mathbb{P}_{\eta_i}\right) \\ -\sum\limits_{i=1}^{n} \left((\hat{w}^\top Z(\Omega))\mathbb{P}_{\eta_i} + \mathbb{P}_{\eta_i}\hat{w} Z(\Omega)^\top\right) \mathbb{P}_{\hat{w}} & \sum\limits_{i=1}^{n} \left(\mathbb{P}_{\eta_i}\hat{w}\hat{w}^\top \mathbb{P}_{\eta_i}\right) \end{pmatrix}$$

The Hessian is written as an element of $\mathbb{R}^{6\times 6}$ due to the identification of tangent vectors in $T_{\hat{w}}S^2$ with elements of $\mathbb{R}^3$. However, the vector $\hat{w}$ is normal to the tangent space $T_{\hat{w}}S^2$ and it follows that $v_0 := (\hat{w}, 0) \in \mathbb{R}^6$ is a zero eigenvector of the Hessian Hess$f$ in (8). The remaining five eigenvalues are associated with the quadratic structure of the cost f at the point $(\hat{w},\hat{\Omega})$. Due to the zero eigenvalue the inverse Hessian in the Newton algorithm has to be implemented with a pseudo inverse routine. In addition, the new estimate must be re-normalised onto the sphere at each iteration of the Newton algorithm [1]. For initial conditions close to the minimum of f each iteration of the Newton algorithm provides an additional two orders of magnitude of accuracy. In practice, at most two or three iterations are sufficient for the purposes of our calculations given that a suitable initial condition is available.

As an additional advantage of applying the Newton algorithm, it is a straightforward exercise to compute the condition number of the Hessian as the ratio of the magnitudes of largest to smallest eigenvalues of the five meaningful eigenvalues of Hess$f$ at the cost minimum. The condition number provides a reliability measure for the estimate of ego-motion of the system, a large condition number indicating that the minimisation is ill-conditioned. The eigenstructure of the Hessian can be used to identify directions of poor resolution of the ego-motion parameters.

5 Results

The combined algorithms were thoroughly tested with synthetically generated optical flow data, and on real video sequences obtained from a small-scale quad-rotor aerial vehicle. For the synthetic data, the true ego-motion of the vehicle is known and can serve as ground truth for comparisons. For the video sequences from the flying vehicle the inertial measurements from the on-board IMU were recorded. The measured rotations are used to de-rotate the spherical flow field. The trajectories of the vehicle can be compared qualitatively to the data obtained, however ground truth data was not available.

For the simulation tests the flow field was generated by creating a Gaussian-distributed point field (offset from origin: 18 along y-axis, sigma=10) for one-sided flow coverage. The offset of the point field simulates incomplete flow from a single camera, covering slightly less than half of the sphere. Tests for surrounding flow coverage where conducted on a Gaussian point field centered at the origin with the same variance.

Flow is created by translating and rotating the point field, and projecting start- and end position onto the sphere. A certain percentage of vectors (30%) is randomised to simulate large outliers, and Gaussian noise ($\sigma = 0.001$ or $\sigma = 0.002$) is added where appropriate in the simulations. Note that the Gaussian noise is applied directly to flow vectors on the unit sphere, which is independent of the pixel resolution of the camera. The value $\sigma = 0.002$ corresponds to 0.115 degrees on the sphere, which is approximately half a pixel for a PAL resolution camera with a 170 degrees field of view. The amount of outliers and noise approximately reflect the distortions found in real optical flow. For simplicity, and independence of specific camera parameters, the simulation is unitless. The spherical camera is simulated by projecting scene points onto a unit sphere. In reality a fish-eye lens can be used, or other means of generating a spherical panoramic image.

One-sided flow

outliers, noise →	0%	0.0	0%	0.001	30%	0.0	30%	0.001	30%	0.002
k-means (translation)	0.061	0.03	2.1	1.8	1.2	1.1	3.1	2.50	5.6	4.9
RANSAC (translation)	0.001	0.001	1.0	0.9	0.5	0.003	2.2	1.6	7.7	3.4
k-means (with rot.)	15.9	15.4	15.2	14.0	17.9	16.0	20.0	16.0	17.6	13.7
RANSAC (with rot.)	0.74	0.002	6.0	0.9	19.6	13.7	12.8	10.2	13.2	12.7

Surrounding flow

outliers, noise →	0%	0.0	0%	0.001	30%	0.0	30%	0.001	30%	0.002
k-means (translation)	0.03	0.015	1.50	1.0	1.4	1.1	2.5	2.0	4.0	3.0
RANSAC (translation)	0.002	0.001	0.5	0.4	0.2	0.002	0.9	0.7	1.8	1.3
k-means (with rot.)	13.9	11.7	14.8	10.6	17.0	11.7	13.7	10.7	12.8	11.9
RANSAC (with rot.)	3.3	0.001	1.2	0.5	12.9	6.1	11.6	4.8	14.1	7.1

Fig. 3 Mean error (left value) and median error (right value) of FoE in degrees for synthetically generated flow with one-sided flow (upper table) and surrounding flow (lower table) coverage

The results can be seen in figure 3. Parameters for the *k*-means algorithm were k=20, and 70 randomly picked pairs. The RANSAC algorithm uses 8 iterations, a threshold $t = 0.05$, 20 vector pairs for the initial hypothesis. The Newton algorithm was run for four iterations.

A key observation found was that the performance of the *k*-means algorithm was not sufficiently reliable to use as the initialisation for the Newton algorithm. In particular, the segmentation of the image flow vectors was not sufficiently robust and the Newton iteration was often undertaken with some outliers that significantly disrupted the performance of the algorithm. As a consequence, the *k*-means results are presented without any refinement step while the RANSAC algorithm contains the Newton refinement as an integral part. The relative performance of the *k*-means (without Newton) versus the RANSAC (with Newton) is clearly seen in figure 2. This can also be seen in the results shown in Table 3. Nevertheless, in the absence of noise all algorithms perform well, even when flow can only be obtained from one side of the sphere. As noise increases, one-sided flow extraction becomes increasingly unstable (notably it is more affected by noise than by outliers). If residual rotation of up to 15 degrees/sec is present in the flow field (i.e. due to imperfect inertial measurements - see lines 3 and 4 in table 3), the estimation results worsen — again more pronounced in the one-sided case.

To test the real-world performance of the algorithm a panoramic PAL camera with a 170 degree field of view fisheye lens was mounted on a linear horizontal rail. The experiment was carried out indoors; most objects were within 2-4 m of the camera. The camera and lens are identical to those used on the quadrotor flying robot presented later. Optic flow is computed using the sparse implementation of pyramidal Lucas-Kanade (from OpenCV) on a 10 by 10 grid which is equally distributed across the image. Due to weight constraints on flying platforms the lens is of relatively low quality which results in significant blur towards the edges of the image. Despite the loss of texture it is still possible to extract usable optic flow from the periphery, albeit with slightly reduced confidence. The mapping of flow to the sphere

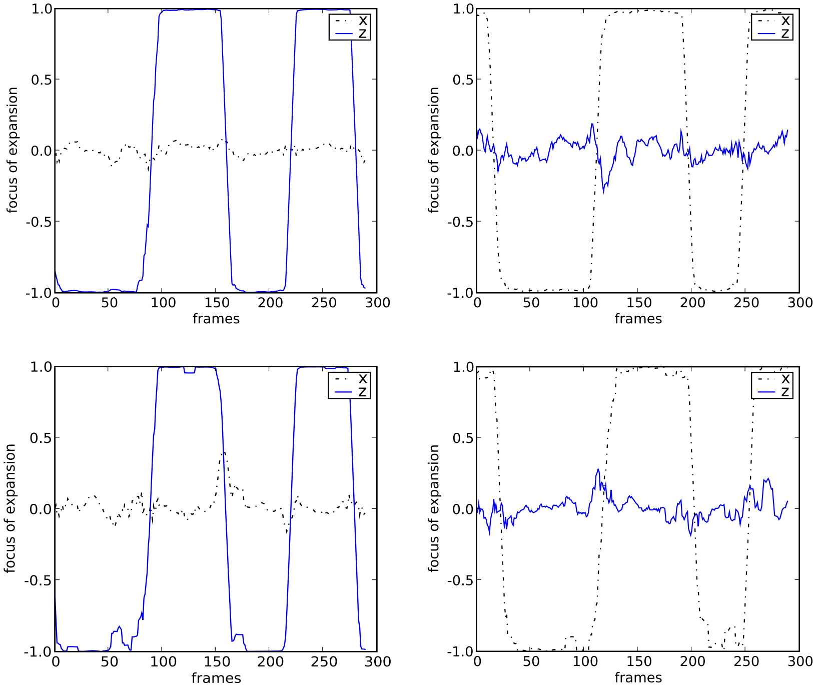


Fig. 4 FoE extraction for a camera mounted on a linear rail. The plots on the left show the results for the camera moving forwards and backwards along the optical axis (top: *k*-means, bottom: RANSAC), the plots on the right for the camera moving sideways perpendicular to the optical axis (again top: *k*-means, bottom: RANSAC).

assumes a constant angle per pixel rate. The mapping function is approximate, a formal calibration was not applied. This means that the algorithms have to be able to cope with measurement and calibration errors that can be expected in many applications. The *k*-means and the RANSAC algorithm were applied to this data and the results are shown in figure 4. The camera did not rotate in this experiment, therefore it no IMU was used, and no derotation was applied. Both algorithms perform very well. Due to the absence of any rotation the *k*-means algorithm performs well, and the RANSAC algorithm does not improve the results.

The algorithms were also applied to video sequences that were collected from a small quad rotor flying vehicle (see figure 1). This electronically stabilised vehicle is equipped with a forward looking PAL camera, an IMU, and radio systems that transmit the real-time video images (25 fps) and inertial measurements to the base station. The IMU is a *ThinIMU micro* by Scinamics with 500 deg/sec full scale range, 3 g accelerometer and 12 bit resolution. Gyro drift after bias calibration is typically 1-2 degrees per minute. The IMU data is transmitted to the ground via a 2.4 GHz *XBee* transmitter. Video is transmitted via a 5.8 GHz analog video link which

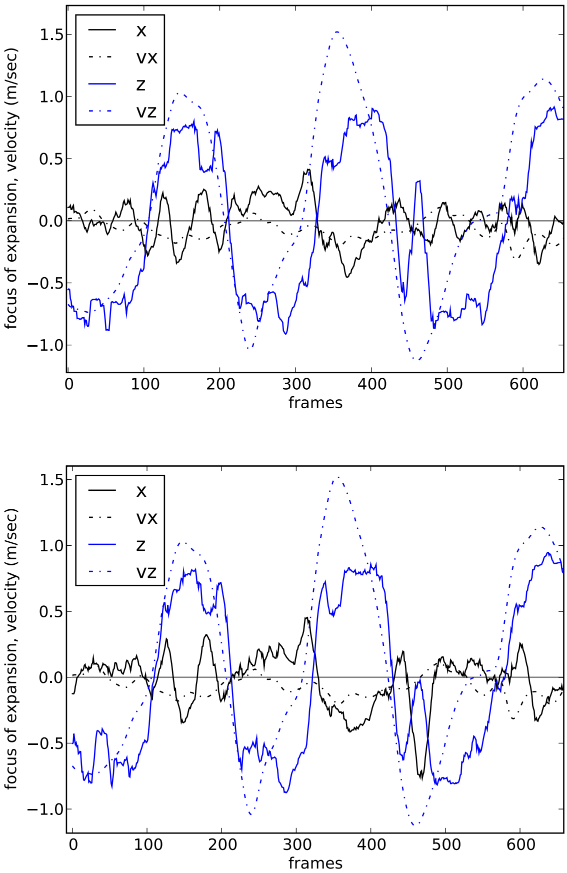


Fig. 5 FoE extraction for real world flyer scenarios, flying forwards and backwards along a straight line. The plot on the top shows the estimates from *k*-means, the lower plot shows the RANSAC results. The dashed lines are the approximate velocity data integrated from the inertial unit. Note that the FoE estimates are normalised, the velocity is not.

offers good video quality at full resolution without compression artifacts and negligible latency. Video images are captured at 720 × 576 pixels by a PCI framegrabber card, but the captured images are downsampled to half resolution before computing optical flow which provided sufficient flow information for motion reconstruction. Efforts have been made to reduce all signal latencies to a minimum. Both signals (video and inertial measurements) were synchronised and recorded on the base station. Approximately 100 flow vectors were computed from each image. The first test sequence consists of the vehicle flying repeatedly forwards and backwards in a straight line for approximately 7-10 m. In the second test sequence the flight pattern is roughly circular (manually controlled). The flight tests were conducted outdoors. A few trees were within 2-3 m of the flight path (sequence 1 only) which produced

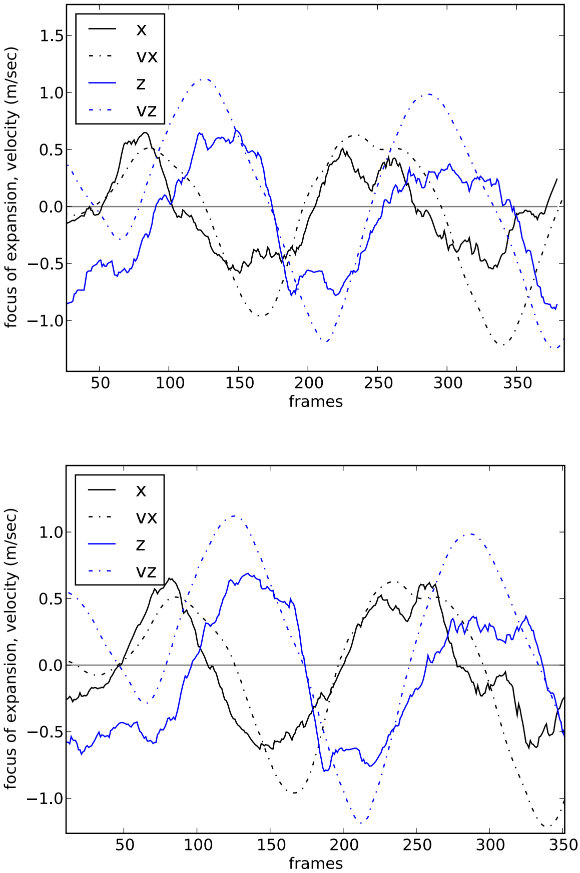


Fig. 6 The same as in figure 5, for the flyer following an approximate circle.

strong translational flow due to the short range. Altitude above ground was approximately 1-2 m, and airspeed reached over one meter per second. The results can be seen in figures 5 and 6. The *z*-axis denotes the forward direction of the flyer, the *x*-axis points to the right, and the *y*-axis points up.

Ground truth data was not available for the flight tests. However, for comparison the velocity estimate from the IMU is plotted as well. The velocity estimate was calculated by applying a complementary attitude filter with drift compensation, subtracting the estimated gravity vector from the accelerometer values, and integrating the residual acceleration. To avoid increasing drift errors a leaky integrator was used which pulls the velocity estimate back to zero over time. To reduce errors the IMU was carefully calibrated, and flights were kept short (20-30 seconds). Despite the imperfections the estimates allow a qualitative comparison between two very different sensor modalities.

Figure 5 shows the FoE estimates sequence 1 (linear flight). The estimates for the sequence 2 (circular flight pattern) are shown in figure 6. The estimates are in vehicle (or camera) coordinates, thus, banking or pitching motions may result in deviations of the estimate from a straight line in world coordinates – however, such motions are small. A sliding window filter of 10 frames was applied to the plots. The plots show that for real data, both the *k*-means and RANSAC algorithm deliver reasonably good estimates of the direction of travel. The reduced performance compared to the initial experiment with a rail mounted camera can be attributed to the imperfect estimation of rotation, synchronisation errors between IMU and video, artifacts from the wireless video transmission (noise, occasional drop-outs), and larger accelerations and velocities. Significant differences between the performance of the two algorithms cannot be concluded as they are obscured by the higher level of noise. It is likely that the performance will improve significantly with more complete coverage of the sphere (i.e. two back-to-back cameras) — especially the RANSAC algorithm should benefit as rotation and translation become easier to separate. Future work will analyse the eigenvalues of the Hessian to measure the sensitivity and confidence for various flow vector distributions.

6 Conclusions

Two methods were presented for estimating the focus of expansion from sparse panoramic optical flow fields, namely a *k*-means clustering method, and a RANSAC framework using a Newton iteration for model fitting. We introduced a cost function, gradient and Hessian for estimation of direction of travel and ego-rotation, which enables gradient descent methods and Newton methods for estimating ego-motion from spherical optic flow. The presented methods work with sparse, patchy flow fields, even if less than half the sphere is covered. Measurements from inertial sensors are used to provide a good initial value of rotation for the algorithms. The algorithms were evaluated and compared on synthetic data; it was found that the RANSAC algorithm performs better, but also that the *k*-means algorithm provides good results at much less computational cost. Tests on video and inertial measurements from a quad-rotor flying vehicle show that both algorithms can be applied to real data obtained from a single fish-eye camera, and provide a good estimate of the direction of travel. On real data the advantages of the RANSAC algorithm are negligible, as any possible improvements on estimation are masked by the higher sensitivity to noise. It is possible that the RANSAC algorithm as presented here will offer better performance is high quality cameras and well-calibrated quality optics are used; however this was not investigated. The presented algorithms can be used in robotics for obstacle avoidance — knowing the direction of travel relative to the world will indicate the point of potential impact. Divergence of optical flow can be computed in that direction to detect obstacles in the current path of the vehicles. The presented cost function and estimation framework can also be used for visual servoing or visual odometry, or as a bias correction input to inertial measurement filters.

Acknowledgements. This research was supported by the Australian Research Council through discovery grant DP0880509.

References

1. Absil, P.-A., Mahony, R., Sepulchre, R.: Optimization Algorithms on Matrix Manifolds. Princeton University Press, Princeton (2008)
2. Adiv, G.: Inherent ambiguities in recovering 3-d motion and structure from a noisy flow field. IEEE Transactions on Pattern Analysis and Machine Intelligence 11, 477–489 (1989)
3. Agrawal, A., Chellappa, R.: Ego-motion estimation and 3d model refinement in scenes with varying illumination. In: IEEE Workshop on Motion and Video Computing (MOTIONS 2005), vol. 2, pp. 40–146 (2005)
4. Baldwin, G., Mahony, R., Trumpf, J.: A nonlinear observer for 6 dof pose estimation from inertial and bearing measurements. In: IEEE International Conference on Robotics and Automation, ICRA 2009 (2009)
5. Brodsky, T., Fermuller, C., Aloimonos, Y.: Directions of motion fields are hardly ever ambiguous. International Journal of Computer Vision 26(1), 5–24 (1998)
6. Fermuller, C., Aloimonos, Y.: Qualitative egomotion. International Journal of Computer Vision 15, 7–29 (1995)
7. Fermuller, C., Aloimonos, Y.: Ambiguity in structure from motion: Sphere versus plane. International Journal of Computer Vision 28(2), 137–154 (1998)
8. Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24, 381–395 (1981)
9. Hartigan, J.A., Wong, M.A.: Algorithm as 136: A k-means clustering algorithm. Applied Statistics 28(1), 100–108 (1979)
10. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge (2000)
11. Irani, M., Rousso, B., Peleg, S.: Recovery of ego-motion using region alignment. IEEE Transactions on Pattern Analysis and Machine Intelligence 19(3), 268–272 (1997)
12. Jepson, A., Heeger, D.: Subspace methods for recovering rigid motion i: algorithm and implementation. International Journal of Computer Vision 7(2) (1992)
13. Kanungo, T., Mount, D.M., Netanyahu, N., Piatko, C., Silverman, R., Wu, A.Y.: An efficient k-means clustering algorithm: Analysis and implementation. IEEE Transactions on Pattern Analysis and Machine Intelligence 24, 881–892 (2002)
14. Koenderink, J., van Doorn, A.: Invariant properties of the motion parallax field due to the movement of rigid bodies relative to an observer. Optica Acta 22(9), 773–791 (1975)
15. Li, H., Hartley, R.: Five-point motion estimation made easy. In: 18th International Conference on Pattern Recognition (ICPR 2006), pp. 630–633 (2006)
16. Lim, J., Barnes, N.: Estimation of the epipole using optical flow at antipodal points. In: OMNIVIS (2007)
17. Lim, J., Barnes, N.: Directions of egomotion from antipodal points. In: Proceedings of CVPR (2008)
18. Longuet-Higgins, H.: A computer algorithm for reconstruction of a scene from two projections. Nature 293, 133–135 (1981)

19. Mahony, R., Hamel, T., Pflimlin, J.-M.: Complementary filter design on the special orthogonal group so(3). In: 44th IEEE Conference on Decision and Control, 2005 and 2005 European Control Conference (CDC-ECC 2005), December 2005, pp. 1477–1484 (2005)
20. Mahony, R., Schill, F., Corke, P., Oh, Y.S.: A new framework for force feedback teleoperation of robotic vehicles based on optical flow. In: Proceedings IEEE International Conference on Robotics and Automation, ICRA (2009)
21. Makadia, A., Geyer, C., Sastry, S., Daniilidis, K.: Radonbased structure from motion without correspondences. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2005) (June 2005)
22. Nister, D.: An efficient solution to the five-point relative pose problem. IEEE Transactions on Pattern Analysis and Machine Intelligence, 756–770 (2004)
23. Roach, J.W., Aggarwal, J.K.: Determining the movement of objects from a sequence of images. IEEE Transactions on Pattern Analysis and Machine Intelligence 2(6), 55–62 (1980)
24. Schill, F., Mahony, R., Corke, P., Cole, L.: Virtual force feedback teleoperation of the insectbot using optic flow. In: Proceedings of the Australasian Conference on Rotoics and Automation, Canberra, Australia (December 2008)
25. Silva, C., Santos-Victor, J.: Direct egomotion estimation. In: Procedings 13th International Conference on Pattern Recognition, vol. 1, pp. 702–706 (1996)
26. Triggs, B., McLauchlan, P.F., Hartley, R.I., Fitzgibbon, A.W.: Bundle adjustment - a modern synthesis. In: Triggs, B., Zisserman, A., Szeliski, R. (eds.) ICCV-WS 1999. LNCS, vol. 1883, p. 298. Springer, Heidelberg (2000)

Control and Motion Planning

Cédric Pradalier

In this part of the ISRR'09 proceedings, we have grouped together a collection of papers related to the planning and execution of motion for very different robots, from the complex hexapod of Haynes et al. to the simple model plane of Lussier Desbiens et al. In all cases, the challenges faced in these papers are related to the management of environment and sensing of uncertainties as well as the incompleteness of the system models.

- In “Gait Transitions for Quasi-Static Hexapedal Locomotion on Level Ground”, Haynes et al. address the complex problem of synchronizing actuation of multiple limbs for legged robots. The introduction of the notion of “Young Tableaux” describes the periodicity and proximity of various kinds of gaits and their transitions in an innovative way.
- In “Stable Dynamic Walking over Rough Terrain”, Manchester et al. also deal with a walking system but in this case focus on exploiting the natural dynamic of an under-actuated system. The key contribution of the approach is to provide a systematic way to stabilize such a system to a target orbit, with local exponential stability.
- In “Design and analysis of hybrid systems with applications to robotic aerial vehicles”, Gillula et al. consider the transition between basic motion primitives and use the Hamilton-Jacobi differential game formulation to find transitions which are provably safe.
- In “Motion Planning under Uncertainty for Robotic Tasks with Long Time Horizons”, Kurniawati et al. focus on another source of complexity in robotic planning, namely the management of uncertainties. By proposing a new point-based POMDP solver, they succeed in planning for long-time-horizon tasks which were not solvable with existing planners.
- In “Scansorial Landing and Perching”, Lussier Desbiens et al. present a very different approach to control for a small airplane trying to land on a vertical wall. By integrating intelligence into the mechanical design, they can simplify the sensing requirements and maneuver planning complexity to a minimum, while still being able to successfully field test the resulting system.
- In “Anthropomorphic Soft Robotics - from Torque Control to Variable Intrinsic Compliance”, Albu-Schaeffer et al. describe the effort of the German Aerospace Centre (DLR) to develop anthropomorphic robots, combining movement precision and intrinsic safety. The central element of

this work is the use of Variable Impedence Actuator, which creates new challenges in terms of control and motion planning.

The core idea when collocating these papers into a control and motion planning section was to show how researchers around the world are proposing solutions to handle the multiple sources of uncertainties inherent to any real-world robotics. Uncertainties stem out from the system complexity: the uncertainties in the perception systems, the limited system actuation, or the introduction of elastic elements into the mechanical design. Among these papers, we believe that the embodied intelligence described by Lussier Desbiens et al. is a good reminder that many control and planning problems can be solved by an intelligent mechanical design in the first place.

Gait Transitions for Quasi-static Hexapedal Locomotion on Level Ground

Galen C. Haynes, Fred R. Cohen, and Daniel E. Koditschek

Abstract. As robot bodies become more capable, the motivation grows to better coordinate them—whether multiple limbs attached to a body or multiple bodies assigned to a task. This paper introduces a new formalism for coordination of periodic tasks, with specific application to gait transitions for legged platforms. Specifically, we make modest use of classical group theory to replace combinatorial search and optimization with a computationally simpler and conceptually more straightforward appeal to elementary algebra.

We decompose the space of all periodic legged gaits into a cellular complex indexed using "Young Tableaux", making transparent the proximity to steady state orbits and the neighborhood structure. We encounter the simple task of transitioning between these gaits while locomoting over level ground. Toward that end, we arrange a family of dynamical reference generators over the "Gait Complex" and construct automated coordination controllers to force the legged system to converge to a specified cell's gait, while assessing the relative static stability of gaits by approximating their stability margin via transit through a "Stance Complex". To integrate these two different constructs—the Gait Complex describing possible gaits, the Stance Complex defining safe locomotion—we utilize our compositional lexicon to plan switching policies for a hybrid control approach. Results include automated gait transitions for a variety of useful gaits, shown via tests on a hexapedal robot.

1 Introduction

Gait transitions are ubiquitous among legged animals and essential for robots. Whereas there is a long and still lively debate about the reason for their value to

G.C. Haynes · D.E. Koditschek
Electrical & Systems Engineering, University of Pennsylvania
200 S. 33rd St, Philadelphia, PA 19104
e-mail: {gchaynes,kod}@seas.upenn.edu

F.R. Cohen
Mathematics Department, University of Rochester
RC Box 270138, Rochester, NY 14627
e-mail: cohf@math.rochester.edu

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 105–121.
springerlink.com

animal runners (optimized joint loads? [4]; optimized energetics? [5]; muscle function or bone strain? [21]), the more limited capabilities of legged robots ensure for years to come that different maneuvers in different environments at different speeds under varied loading conditions will require the adoption of distinct locomotion patterns, along with necessitating the ability to transition between them safely and efficiently. The great variety of gaits found in nature—quadrupedal walking, trotting, pacing, and galloping; hexapedal wave gaits and alternating tripods [15, 11, 22]; and so on—persuades us of the importance in building a general framework to identify and produce reliable transitions amongst all gaits a robot can use.

A variety of methods have been proposed for switching gaits in legged robots, however most have not considered underactuated systems, in which legs do not have full control over the timing of stance and recirculation throughout a full stride. Examples of underactuated legged robots include the RHex robotic hexapod [17] and the RiSE climbing robot [19], legged machines respectively capable of running and climbing on many unstructured terrains. In the case of RHex, each leg contains a single actuator, thus modification of gait timing must occur during recirculation, so as to not produce inconsistent toe velocities during stance. For RiSE this is even more imperative when climbing a wall because inconsistencies of toe velocities while attached to a climbing surface can cause a robot to lose grip and fall. For this reason, we have developed a variety of prior methods using only gait timing modification during leg recirculation [10, 9] in order to change gaits during locomotion.

This paper focuses upon methods of merging low-level regulation control of gaits, as described above, with high-level task planning, in which hybrid control of various different gait strategies is necessary. We address the problem of producing safe, efficient gait control for underactuated robots via switching policies amongst families of gait limit cycle attractors. We do so by exploiting the algebraic structure of two distinct symbolic decompositions of the limb phase space: the Gait Complex, introduced in Section 2; and the Stance Complex, introduced in Section 3. Our techniques in this paper build upon basic ideas presented in [12], but we introduce methods that are more general and more comprehensive in scope, particular to the application of gait switching. Our specific contributions lie in the introduction of these two cellular decompositions of the phase space that we use to

i. enumerate the allowable gaits of a legged system;
ii. design a mixed planning/control method to navigate amongst them;
iii. execute these transitions in real-time during continuous legged locomotion.

Initial results are presented for a walking hexapod, while future applications include feedback-driven general terrain locomotion for walking, running, and climbing robots, while requiring minimal sensory information and computational power.

2 Hybrid Control over the Gait Complex

2.1 The Gait Complex, $\mathsf{Gaits}_n[\mathbb{T}]$

In [2], we endow $\mathbb{T}^{n+1}/\mathbb{T} \approx \mathbb{T}^n$ with the structure of a cell complex, denoted $\mathsf{Gaits}_n[\mathbb{T}]$, the disjoint union of its j-skeleta

$$\mathsf{Gaits}_n[\mathbb{T}] = \coprod_{j=0}^{n} \mathsf{Gaits}_n[\mathbb{T}]^j,$$

collections of j-dimensional submanifolds assembled in $\mathsf{Gaits}_n[\mathbb{T}]^j$, with appropriate "gluing" identifications at their boundaries [6]. Although the cardinality of this cell complex must grow combinatorially in the degrees of freedom, n, it is sufficiently regular to enjoy the additional structure of a *Delta Complex* wherein each cell of each of the skeleta is the image of a standard unit simplex whose boundaries are formally associated via a family of "characteristic maps" [6].

We find it useful to index the various cells of the complex, $\mathsf{Gaits}_n[\mathbb{T}]$, by means of equivalence classes of Young Tabloids [16], $\mathsf{T} \in \mathcal{T}^{n+1}_{k+1}$, arrays of (typically) unevenly long strings of integers taken from the set $\{1, \ldots, n+1\}$ with no replacement, each of whose $k+1$ rows denotes a "virtual leg" (a subset of legs that is locked in steady state at the same relative phase for the gait being described), and whose row order corresponds to the cyclic order of virtual legs in the gait. We show formally in [2] that a certain quotient (that is, a complete transversal of left cosets [16] arising from a particular subgroup) of the permutation group $\Sigma_{n+1} \times \Sigma_{k+1}$ is in one-to-one correspondence with the gait complex $\mathsf{Gaits}_n[\mathbb{T}]^k$, but for purposes of this paper it suffices to provide the following intuitive characterization of the equivalence classes as follows. Two tabloids, $\mathsf{T}, \mathsf{T}' \in \mathcal{T}^{n+1}_{k+1}$, index the same cell in the k-skeleton, $\mathsf{Gaits}_n[\mathbb{T}]^k$ if and only if: (i) there is a bijection between their rows (each considered as an unordered collection of integers); and (ii) the bijection is some power of the "full shift", $\zeta \in \Sigma_{k+1} : (1, 2, \ldots, k, k+1) \mapsto (k+1, 1, 2, \ldots, k)$.

In this paper, we make twofold use of the Young Tabloids. First, each Tabloid provides an algorithmic specification of a gait generator over the cell it indexes. We will sketch the nature of this algorithm in Section 2.2 and provide some illustrative examples in Table 2. Second, a computationally simple Tabloid operator, $\partial : \mathcal{T}^{n+1}_{k+1} \to 2^{\mathcal{T}^{n+1}_{k}}$ computes the set of tabloids indexing the boundary cells in $\mathsf{Gaits}_{n+1}[\mathbb{T}]^k$ of the cell in $\mathsf{Gaits}_{n+1}[\mathbb{T}]^{k+1}$ indexed by its argument. We will use this operator as the key computational component in the transition planner presented in Section 4. Given length constraints, it does not seem possible in this paper to present any more formal an account of these ideas (which are formally developed in [2]) and we seek rather to provide an intuitive feeling for what the machinery offers through the use of examples and the informal pictures and tables in the Appendix.

2.2 The Gait Fields

While not required for qualification as a Delta Complex, we find in this application the need for a family of "normal" maps $n_{\mathsf{T}} : \mathbb{T}^{n+1} \to \mathbb{T}^{n-k}$ associated with each cell indexed by its tabloid $\mathsf{T} \in \mathcal{T}^{n+1}_{k+1}$ whose Jacobian yields the normal bundle, $Tp_{\mathsf{T}}{}^{\perp}$ defined by the corresponding inclusion map. Specifically, we use them here to build gradient vector fields that "force" the resulting flows toward the designated cell wherein the flow of the desired reference field is known to produce a desired gait. To do so, first observe (as shown formally in [2]) that

$$\nu_m : \mathbb{T}^m \to [0,1] : (r_1, ..., r_m) \mapsto 1 - \frac{1}{m} \sum_{i=1}^{m} \cos \angle r_i \tag{1}$$

is a perfect Morse function with critical points in $\{0, \pi\}^m$ each of which have Morse index specified by number of π entries. It follows that the flow associated with the gradient vector field, grad $\nu_{\ m}$, takes almost all initial conditions in $\mathbb{T}^m$ to the identity, $(e^{2\pi i 0}, \dots, e^{2\pi i 0})$. Thus, given a tabloid, $\mathsf{T} \in \mathcal{T}^{n+1}_{k+1}$, the Morse function $\nu_{\mathsf{T}} := \nu_{n-k} \circ n_{\mathsf{T}}$ defines a gradient vector field whose flow brings almost every initial condition in $\mathbb{T}^{n+1}$ to the cell in $\mathbb{T}^{n+1}/\mathbb{T}$ that T indexes. Examples of the normal maps associated with each cell of the three-legged complex, $\mathsf{Gaits}_2[\mathbb{T}]$ are listed in Table 1.

Denote by $R^1_{BC} : \mathbb{T}^1 \to T\mathbb{T}^1$ the "Buehler Clock" reference generator first introduced in [18] that encodes a one dimensional circulation flow undergoing a phase interval of slow "stance" motion corresponding to the behavior we presume appropriate when a leg is in contact with the ground, followed by a complementary phase interval of fast "recirculation" corresponding to the time interval over which a leg will be lifted off from the ground and returned ready for its next stance phase. This simple rhythm generator can be "pushed forward" to $\mathbb{T}^2$ via the inclusion (5) as $R^2_{BC} := P_{\begin{array}{|c|c|}\hline 1&2\\\hline\end{array}} \cdot R^1_{BC} \circ p^{\dagger}{}_{\begin{array}{|c|c|}\hline 1&2\\\hline\end{array}}$. The sum, $F^2_{PR} = R^2_{BC} - \text{grad}\, \nu_{\begin{array}{|c|c|}\hline 1&2\\\hline\end{array}}$, which can be written in angular coordinates (see footnote 3) as

$$F^2_{PR}(x_1, x_2) = R^1_{BC}(x_1) \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \sin(x_1 - x_2) \begin{bmatrix} 1 \\ -1 \end{bmatrix}. \tag{2}$$

induces a flow under which almost every pair of phases is brought to a bipedal "pronk"—a limit cycle characterized by both legs recirculating in phase—a cycle over the cell in $\mathsf{Gaits}_1[\mathbb{T}]$ indexed by $\begin{array}{|c|c|}\hline 1&2\\\hline\end{array} \in \mathcal{T}^2_1$.

In contrast, let us construct the alternating phase bipedal gait generator displaying the archetype of a gait in the "antiphase" cell of $\mathsf{Gaits}_1[\mathbb{T}]$ indexed by $\begin{array}{|c|}\hline 1\\\hline 2\\\hline\end{array} \in \mathcal{T}^2_1$. By conjugation, $R^2_{AP} := Dh^2_{TR} \cdot F^2_{PR} \circ \left(h^2_{TR}\right)^{-1}$, through the translation, $h^2_{TR} : \mathbb{T}^2 \to \mathbb{T}^2 : (x_1, x_2) \mapsto (x_1, \pi + x_2)$, we define a new vector field

$$R^2_{AP}(x_1, x_2) = R^1_{BC}(x_1) \begin{bmatrix} 1 \\ 1 \end{bmatrix} - \sin(x_1 - x_2 + \pi) \begin{bmatrix} 1 \\ -1 \end{bmatrix}. \tag{3}$$

that shifts the roles of the two invariant submanifolds of the pronking field, F^2_{PR}.

Table 2 provides a detailed listing of the various intermediate fields required to construct two of the most familiar hexapedal gaits: the alternating tripod, R^6_{AP} [18], and the stair climbing gait, R^6_{Stair} [14]. All of the gaits used in the experiments reported here are generated in a similar manner.

3 The Task of Locomotion and the Stance Complex

The Gait Complex, $\mathsf{Gaits}_{n-1}[\mathbb{T}]$, describes all possible gaits for an n-legged robot. Locomotive differences exist, however, amongst the various gaits. We introduce a second cellular decomposition of $\mathbb{T}^n$, the Stance Complex, $\mathsf{Stance}_n[\mathbb{T}]$, to describe the inherent discreteness of legged locomotion and note all possible leg support configurations. We utilize this complex to identify a priori which cells of $\mathsf{Gaits}_{n-1}[\mathbb{T}]$ produce viable statically stable locomotion.

Computation of static stability margin of locomotion can be determined by projecting the mass center onto a support polygon defined by a robot's surface contacts, a computation that can be quite expensive in the presence of complex surface interactions [1]. For a particular contact configuration of limbs, a single cell of $\mathsf{Stance}_n[\mathbb{T}]$, we argue, this stability margin is locally a continuous function of posture, but varies more dramatically (and discretely) when toe contacts are added or removed, other cells of $\mathsf{Stance}_n[\mathbb{T}]$. In the case of underactuated robots, where available postures are limited by low numbers of degrees of freedom, this is particularly true. Even for high degree of freedom system, the workspace of individual limb motions can be quite small when compared to body size, thus expounding the small variation in stability margin within a cell of $\mathsf{Stance}_n[\mathbb{T}]$ compared to dramatic changes when making or breaking contact.

3.1 The Stance Complex

The Gait Complex of Section 2 describes all possible gait timings, noting specific leg phase relationships. Of the immense number of possible gaits, 1082 for a 6-legged robot, not all are locomotively viable, as many may recirculate legs together which produce unstable configurations of the robot's body. $\mathsf{Stance}_n[\mathbb{T}]$ provides us with accurate constraints regarding this aspect of the locomotion task.

Each axis of the n-torus corresponds to the possible gait timings for an individual leg during locomotion, containing both stance and recirculation as discrete regions on the axis. The duty factor of a gait, $\delta \in (0, 2\pi)$, reflects the percentage of stride spent in stance versus recirculation, thus for an individual leg i, if $x_i < \delta$ the leg is considered to be in stance. This demarcation is defined for each axis of the torus, thus producing a complex $\mathsf{Stance}_n[\mathbb{T}]$ of 2^n total cubical cells, as well as intersecting faces and edges. As an example of a cubical member of $\mathsf{Stance}_n[\mathbb{T}]$, consider the cell where $\forall_i x_i < \delta$. This cell corresponds to all legs in stance, while $\forall_i x_i \geq \delta$ is all legs in recirculation.

3.2 *Examples:* $\mathsf{Stance}_2[\mathbb{T}]$ *and* $\mathsf{Stance}_3[\mathbb{T}]$

If we consider the 2-torus, the space of gait timings for a bipedal robot, there exist four cells shown. One cell on this torus has both legs in stance, two cells have a single leg recirculating, and one cell has both legs recirculating. Assuming a quasi-static locomotory system, it would be dangerous for the robot to use a gait that tries to recirculate both legs at once, thus this last cell should be avoided.

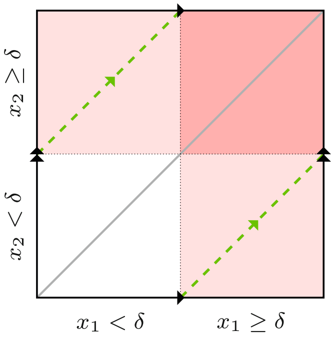


Fig. 1 $\mathsf{Stance}_2[\mathbb{T}]$: Demarcations between stance and recirculation produce 4 unique cells in the Stance Complex. With a stance duty factor of 50% ($\delta = \pi$), there exists only a single gait (dashed line) that does not pass through the cell corresponding to both legs recirculating together (upper right).

Considering a similar system on the 3-torus, depicted as a cube with faces identified in Fig. 2, we demarcate each axis with regions dedicated to stance and recirculation to produce a total of 8 cubical cells in the Stance Complex. Depending upon the exact mechanics of the robot, it may be undesirable to recirculate certain sets of legs together. In the figure we highlight potentially dangerous cells that recirculate 2 or 3 of the legs together at the same time. The cell where all legs are in stance, or only a single leg recirculates, would be considered safe cells.

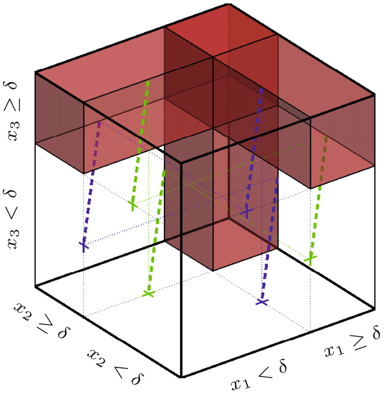
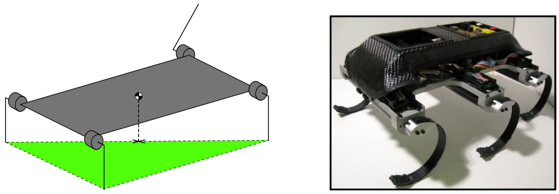


Fig. 2 $\mathsf{Stance}_3[\mathbb{T}]$: A total of 8 unique cubical cells exist in the Stance Complex for the 3-torus. In this figure are highlighted the 4 cells which recirculate 2 or more legs of a 3-legged robot simultaneously. A safe gait, in this scenario, would try to only recirculate a single leg at a time, two such gaits possible when $\delta = \frac{2}{3}2\pi$.

3.3 Static Stability Metric

We utilize $\mathsf{Stance}_n[\mathbb{T}]$ to define in general the global properties of static stability for an n-legged robot. This approach allows us to evaluate gait stability simply by studying the cells through which a given gait passes.



(a) Basic model of rotary joint quadruped

(b) Picture of a RHex-style hexapedal robot

Fig. 3 For basic analysis of static stability, we consider a robot with single actuators per hip, similar to the RHex-style robot shown. With such a model, the stability margin of gaits is computed.

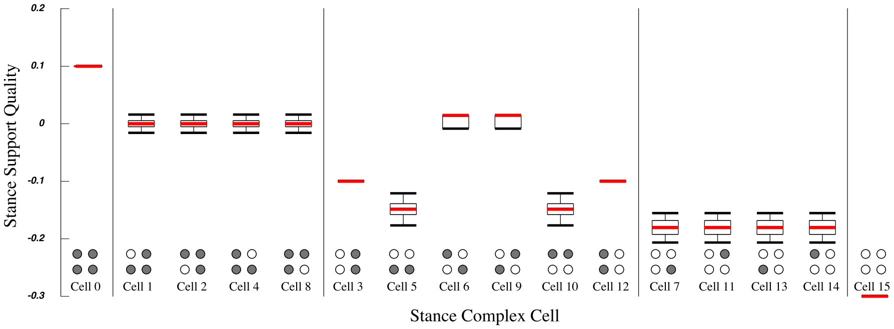


Fig. 4 $\mathsf{Stance}_4[\mathbb{T}]$: There exist 16 cubical cells within $\mathsf{Stance}_4[\mathbb{T}]$, with varying stability values between all. Cells with more legs in stance (represented as shaded circles on simple representations of robot at bottom) have greater stability margins.

Figure 4 shows analysis of the Stance Complex on $\mathbb{T}^4$. Using our simple model of a quadrupedal robot (Fig. 3), each cell is tested for static stability, consisting of a total of $160,000$ tested configurations for the 16 cubical cells, taking into full consideration the entire gait space of $\mathbb{T}^4$. Cells with more legs in stance offer greater static stability. Similarly, certain cells with two legs in stance perform better than others, for instance showing that the cells corresponding to a trot gait, cells 6 and 9, have greater stability than those for pace and bound gaits.

4 Planning and Control Approach

Utilizing both the Gait Complex and Stance Complex, we develop a mixed planning and control approach to automate safe switching between gaits. We intersect cells of $s^{-1}\left(\mathsf{Gaits}_{n-1}[\mathbb{T}]\right)$ with unsafe cells of $\mathsf{Stance}_n[\mathbb{T}]$ in order to prune gait cells that do not produce safe locomotion. A search algorithm is then used to plan routes amongst the remaining cells to generate a sequence of Young Tabloids that are used in a hybrid switching controller that transitions between gaits, while avoiding dangerous, unstable cells of the Stance Complex.

4.1 Transitions on Hasse Diagram of Gaits

The planning component of our hybrid controller relies heavily upon the partial order relation of adjacency by boundary (that we denote by $\succ$) in the Gait Complex. Topologically, it is impossible to pass from one cell to an adjacent neighbor of equal dimension without passing through a "neighbor" on the shared boundary. The very notion of cell adjacency is characterized by this partial order—conveniently captured by the formalism of the *Hasse Diagram* [16]. As different leg combinations incur very different locomotion behaviors (different passages through $\mathsf{Stance}_n[\mathbb{T}]$ in the present problem dealing with static stability) the choice of intermediate cells along the way from one to another gait—i.e., the particular path through the Hasse Diagram—requires a level of methodical scrutiny that we entrust in the present paper to a planner. The strong correspondence between the cellular structure of the Gait Complex, $\mathsf{Gaits}_n[\mathbb{T}]$, and the tabloids, $\mathcal{T}^n$ we use to index it affords our planner a very simple operation over the latter that faithfully represents the boundary operation over the former which we now outline.

Given a tabloid, $\mathsf{T} \in \mathcal{T}^{n+1}_{k+1}$, indexing a cell in $\mathsf{Gaits}_n[\mathbb{T}]^k$ there is a very simple operation,

$$\partial : \mathcal{T}^{n+1}_{k+1} \to 2^{\mathcal{T}^{n+1}_{k}}; \quad 0 \le k \le n$$

yielding the tabloids that index all its boundary cells in a manner we merely sketch here (but present rigorously along with a proof of correctness in [2]) as follows. For each pair of contiguous rows of T, collapse the entries into one row comprising the union of the entries of the pair. Compute such a collapsed tabloid for each successive continguous pair of rows, including, finally, the first and the last row, so as to achieve a set of $k+1$ tabloids in $\mathcal{T}^{n+1}_k$. Each of these indexes one and only one adjacent (boundary) cell in the $(k-1)$-skeleton, $\mathsf{Gaits}_n[\mathbb{T}]^{k-1}$. Hopefully, it is intuitively clear that the "tabular inverse" of this operation,

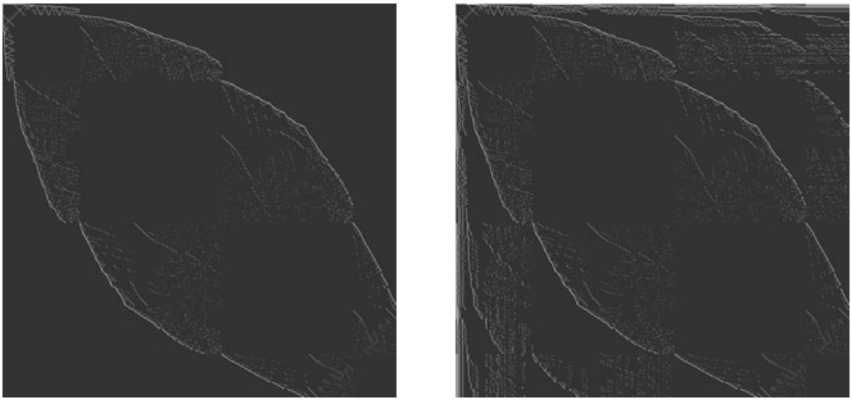
$$\partial^{-1} : \mathcal{T}^{n+1}_{k} \to 2^{\mathcal{T}^{n+1}_{k+1}}, \quad 0 \le k \le n$$

yields the one-row-longer tabloids that index the cells that share a boundary component indexed by the argument. For example the partial order adjacency relation at the quadrupedal "half-bound" gait (for example reported in [8]) is computed as

$$\left\{\begin{array}{|c|}\hline 1\\\hline 2\\\hline 3\\\hline 4\\\hline\end{array}, \begin{array}{|c|}\hline 1\\\hline 2\\\hline 4\\\hline 3\\\hline\end{array}\right\} = \partial^{-1}\left\{\begin{array}{|c|c|}\cline{1-1} 1&\multicolumn{1}{c}{}\\\cline{1-1} 2&\multicolumn{1}{c}{}\\\hline 3&4\\\hline\end{array}\right\} \succ \left\{\begin{array}{|c|c|}\cline{1-1} 1&\multicolumn{1}{c}{}\\\cline{1-1} 2&\multicolumn{1}{c}{}\\\hline 3&4\\\hline\end{array}\right\} \succ \partial\left\{\begin{array}{|c|c|}\cline{1-1} 1&\multicolumn{1}{c}{}\\\cline{1-1} 2&\multicolumn{1}{c}{}\\\hline 3&4\\\hline\end{array}\right\} = \left\{\begin{array}{|c|c|c|}\hline 1&3&4\\\hline 2&\multicolumn{2}{c}{}\\\cline{1-1}\end{array}, \begin{array}{|c|c|c|}\cline{1-1} 1&\multicolumn{2}{c}{}\\\hline 2&3&4\\\hline\end{array}, \begin{array}{|c|c|}\hline 1&2\\\hline 3&4\\\hline\end{array}\right\}$$

Considering legs in recirculation, this definition of adjacency makes intuitive sense: when multiple legs enter recirculation together, as legs within the same row of a tabloid would, it is possible for one leg to speed up while another slows, thus splitting the row apart as the current gait cell changes. Conversely, legs entering recirculation may wait indefinitely for other legs—so long as the robot remains statically stable—such that they synchronize, merging rows of a tabloid.

Using this definition of adjacency between tabloids, and starting from the initial "pronk" tabloid on $\mathbb{T}^6$, $\begin{array}{|c|c|c|c|c|c|}\hline 1&2&3&4&5&6\\\hline\end{array}$, we build an adjacency matrix amongst all 1082 cells of $\mathsf{Gaits}_5[\mathbb{T}]$, shown in Fig. 5a. This matrix is block adjacent, since a given tabloid may only be adjacent to cells with either one less or one more length in rows. If we extend our definition of adjacency to allow multiple *sequential* rows to be compressed together (or a single row split into more than rows)—a reflection that more than two groups of legs can split or join in a given operation (or, more topologically, that we will allow immediate passage to boundaries of boundaries of cells in the gait complex)—this expands the definition of the Hasse Diagram to include other adjacencies, shown in Fig. 5b.



(a) Hasse Diagram of Young Tabloids

(b) Hasse Diagram including multi-dimensional adjacency

Fig. 5 Hasse Diagrams over the set of Young Tabloids. A 1082 × 1082 matrix represents the graph of gait cells, white dots representing adjacency. Cells are in order of increasing dimensionality, block-wise groupings shown here, prior to filtering based upon the Stance Complex.

Lastly, before applying a discrete planner over this set of gait cells, we prune based upon static stability. In our basic implementation of the Stance Complex, we limit ourselves to cells that do not recirculate either ipsilateral nor contralateral legs (excluding the middle pair) together, thus only allowing 18 out of the 64 cells of $\mathsf{Stance}_n[\mathbb{T}]$. This conservative restriction on the Stance Complex, when intersected with the Gait Complex, reduces allowable gait cells to only 477 of the original 1082, however all 477 are statically stable gaits. Another result of the conservative estimate of gait stability is the existence of disconnected clusters of gaits in the Gait Complex. Of the 477 gaits, 301 exist in one large cluster (including the most commonly accessed legged gaits for $N = 6$) with another two symmetric clusters of 87 gaits, while each of the two circular crawls is likewise disconnected from all other gaits. By our estimation of static stability, it is impossible to reach one cluster from another, due to our constraints on ipsilateral and contralateral legs.

4.2 Planning Gait Complex Switching

We consider the problem of an underactuated robot, where freedom to control leg phasing only occurs during recirculation. To discretely plan over this set of operations, we utilize an A* planner that computes cost, in terms of total transition time, between arbitrary cells of the Gait Complex.

The cost of an individual transition between two cells depends upon the initial phase of the first gait. Given that legs must recirculate together to switch, we sum the wait time until legs begin recirculation with total time of recirculation to get actual cost. An admissible heuristic in this case is cost of 1.0, as adjacent cell transitions cannot take more than one stride.

4.3 Controller Activation

Our controllers take a given sequence of tabloids, as output by the A* planner, and construct individual reference field controllers, following from the examples in Section 2.2. Several additional controller modifications are as follows.

Foremost, active control of leg phase occurs only during recirculation. We assume legs in stance to be "rigidly" attached to the surface, particularly relevant when considering climbing robots where inappropriate torquing of individual feet may cause them to lose grasp. In this way, the gradient field simply zeros any action along axes for legs currently in stance.

The duty factor of the system is also modified when the robot approaches new gaits. A virtual biped gait, such as the alternating tripod, will use a gait with 50% duty factor when close in phase space. A simple controller that provides an algebraic relationship between duty factor and phase is detailed in [7].

Lastly, in our definition of the reference field controllers, we leave freedom in the choice of the exact structure of an individual tableaux. [1]

$$t\left(\begin{array}{|c|c|}\hline 1 & 2 \\ \hline 3 & 4 \\ \hline\end{array}\right) = \left\{\begin{array}{|c|c|}\hline 1 & 2 \\ \hline 3 & 4 \\ \hline\end{array}, \begin{array}{|c|c|}\hline 2 & 1 \\ \hline 3 & 4 \\ \hline\end{array}, \begin{array}{|c|c|}\hline 1 & 2 \\ \hline 4 & 3 \\ \hline\end{array}, \begin{array}{|c|c|}\hline 2 & 1 \\ \hline 4 & 3 \\ \hline\end{array}\right\}$$

[1] Following the terminology of [16], a *tabloid* is the equivalence class of all numerically filled-in diagrams whose rows are identical, disregarding the order of the integer entries of each row.

Each tableaux describes the same system, consisting of the same limit cycle gait, with different enumerations of legs within the rows of the tableux. The distinction between individual tableaux, however, affects the construction of the related reference field controllers, as the first element of each row is chosen as *leader* (formally specified by the choice of left inverse as exemplified by $p_{\boxed{1}\boxed{2}}$ in equation (5)). To effect a rational choice of "leader" in any new instantiation of a gait, we consider all identified tableaux of a specific tabloid T, and make use of the one whose potential function has the lowest value.

$$\operatorname*{argmin}_{T' \in t(T)} \nu_{T'}(r)$$

By selecting the minimum cost potential function from which to generate our reference field, we achieve an online adjustment of the transient behaviors of the system such that it follows near-minimum distance paths between gaits, without introducing local minima (as each possible x has the same stable critical point and our system always decreases in potential). For an alternating tripod gait, this operation includes a total of 36 function evaluations in order to choose an ordering.

5 Experimental Results

Using a hexapedal robot platform, we have implemented our gait transition method and shown its efficacy in producing near arbitrary transitions between safe gaits while preventing loss of static stability. We discuss examples of such transitions, compare with a naïve coupled oscillator approach, and project directions in which this research will enable new behaviors for both walking and climbing legged robots.

5.1 Gait Switching

Our new gait switching methods attempt to rectify deficiencies in our prior approaches. For the domain of climbing robots, we have constructed hand-designed transitions between gaits [10], but these transitions were not easily generated nor guaranteed. Further work produced control laws that converged to desired gaits [7], but was limited to a small number of gaits with no choice of which exact gait to converge. The methods described here attempt to automate the generation of transitions, allow transitions between arbitrary pairs of gaits, while preventing static instability.

Fig. 6 shows the transition from a crawl gait to the alternating tripod gait.The top of the figure shows the sequence of tabloids that the planner has determined to converge the fastest. The bottom plot shows roll, pitch, and yaw angles, relatively stable while undergoing a transition of gait.

Fig. 7 shows two different attempts to produce a non-trivial gait transition. The first uses the tabloid-derived control law to simply converge to the desired gait, however it has the undesireable results of poorly designed paths of convergence, following from a basic coupled oscillator approach [12]. The second uses a sequence of planned intermediary gaits to produce a transition that retains static stability.

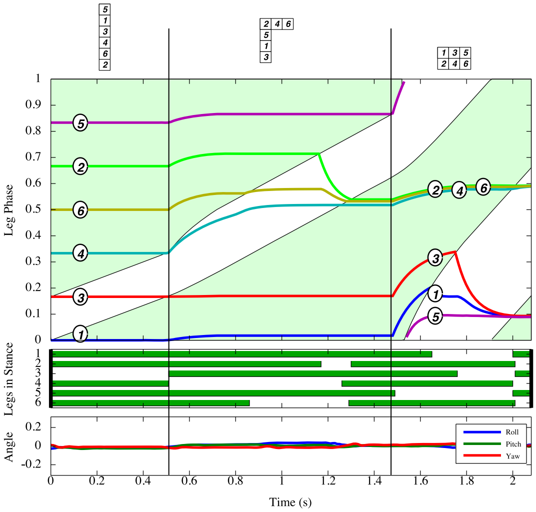
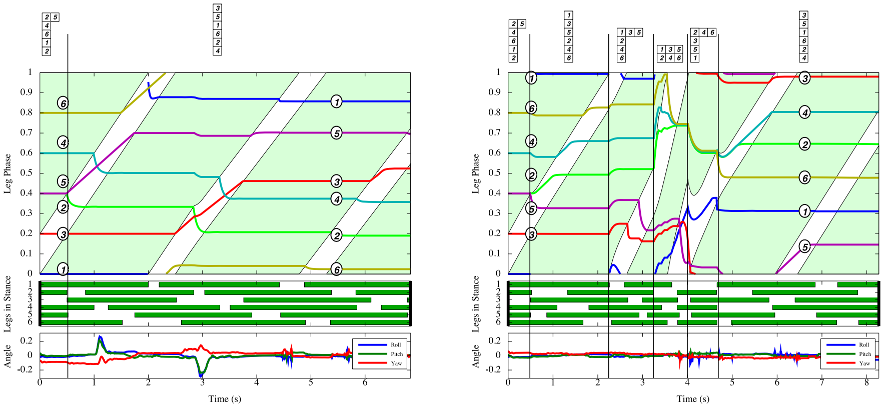


Fig. 6 A sequenced transition of Young Tabloids between gaits. At each switch time, the controller changes, converging to the next gait. End result is a transition behavior that avoids static instabilities. Each line is a "phase offset" of a given leg [**?**], with green regions indicating stance. Phase control occurs during recirculation while adjustment of duty factor (ratio of green to white) takes place as the system reaches desired gaits.



(a) Direct Controller Transition

(b) Planned Sequence of Controllers

Fig. 7 Two transitions between gaits. The planned sequence (right) prevents loss of static stability, while the direct approach (left) loses static instability, as measured by pitch-roll-yaw angles from a Vicon system.

As can be seen in the plots, the unplanned version recirculates too many legs together, loses static stability, and pitches, rolls, and yaws during the transition. The planned version remains relatively level throughout the entire transition. Inconvenient and disruptive during a level ground walk, such perturbations would be catastrophic in a climbing setting, or even, most likely, in a high performance dynamical level ground setting.

6 Conclusions and Future Work

We have introduced a combined planning and control method that uses both discrete and continuous representations to plan and execute transitions amongst gaits implemented on an underactuated legged robot. Introduction of both the Gait Complex, a structure that characterizes all the possible one-cycles achievable with an n-legged machine, as well as the Stance Complex, classifying the ground contact status of all legs, brings about a greater understanding of the space of gaits, and points the way to global approaches for gait control.

In these preliminary experimental results, it seems possible that naïve transitions using gaits that simply avoid bad cells of the Stance Complex may perform just as well as the planner's sophisticated use of cell adjacency. Furthermore, the situation is of course a good deal more complex than we allow in this first paper on these cellular decompositions. For example, the present gait reference fields (Sec. 2) yield steady state limit cycles whose paths maintain rigid phase relationship amongst legs. There is no reason not to consider more general gaits whose orbits may wind about the torus in different ways in order to avoid bad cells of the Stance Complex.[2]

We are currently studying methods of extending our level-ground transitions to domains in which the terrain varies, such as climbing over rubble-like obstacles or transitions between level-ground locomotion and vertical climbing, in which we expect different sets of viable gaits to be available to us. In both of these cases, studying how body geometry and contact mechanics affect the allowable cells of the Gait and Stance Complexes is part of our future work.

Acknowledgements. This work was supported in part by the ONR HUNT project, the DARPA SToMP project, and an Intelligence Community Postdoctoral Fellowship held by the first author.

References

1. Bretl, T., Lall, S.: Testing static equilibrium for legged robots. IEEE Transactions on Robotics 24(4), 794–807 (2008)
2. Cohen, F., Koditschek, D.E.: Hybrid control over the coordination complex (2009) (in preparation)
3. Cohen, F.R.: On Configuration Spaces (2008) (in preparation)
4. Farley, C., Taylor, C.: A mechanical trigger for the trot-gallop transition in horses. Science 253, 306–308 (1991)

[2] For an example of a more complicated family of reference gait generators capable of producing such "winding" limit cycles, consider the example in Fig. 3 of [20].

5. Griffin, T., Kram, R., Wickler, S., Hoyt, D.: Biomechanical and energetic determinants of the walk-trot transition in horses. Journal of Experimental Biology 207, 4215–4223 (2004)
6. Hatcher, A.: Algebraic topology. Cambridge University Press, Cambridge (2002)
7. Haynes, G.C.: Gait regulation control techniques for robust legged locomotion. Ph.D. thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA (2008)
8. Haynes, G.C., Khripin, A., Lynch, G., Amory, J., Saunders, A., Rizzi, A.A., Koditschek, D.E.: Rapid pole climbing with a quadrupedal robot. In: Proc. IEEE International Conference on Robotics and Automation (2009)
9. Haynes, G.C., Rizzi, A.A.: Gait regulation and feedback on a robotic climbing hexapod. In: Proceedings of Robotics: Science and Systems, Philadelphia, USA (2006)
10. Haynes, G.C., Rizzi, A.A.: Gaits and gait transitions for legged robots. In: ICRA, Orlando, FL, USA, pp. 1117–1122 (2006)
11. Hildebrand, M.: Symmetrical gaits of horses. Science 150, 701–708 (1965)
12. Klavins, E., Koditschek, D.E.: Phase regulation of decentralized cyclic robotic systems. International Journal of Robotics Research 21(3) (2002)
13. McMordie, D., Buehler: Towards pronking with a hexapod robot. In: Proc. 4th Intl. Conf. on Climbing and Walking Robots, Storming Media, Karlsruhe, Germany, pp. 659–666 (2001)
14. Moore, E.Z., Campbell, D., Grimminger, F., Buehler, M.: Reliable stair climbing in the simple hexapod 'rhex'. In: IEEE International Conference on Proceedings Robotics and Automation 2002 (ICRA 2002), vol. 3 (2002)
15. Muybridge, E.: Animals in Motion. Dover Publications, New York (1899)
16. Sagan, B.: The Symmetric Group: Representations, Combinatorial Algorithms and Symmetric Functions. Wadsworth and Brooks/Cole, Mathematics Series (1991)
17. Saranli, U., Buehler, M., Koditschek, D.E.: RHex: A Simple and Highly Mobile Hexapod Robot. The International Journal of Robotics Research 20(7), 616–631 (2001), doi:10.1177/02783640122067570
18. Saranli, U., Buehler, M., Koditschek, D.E.: Rhex: A simple and highly mobile hexapod robot. The International Journal of Robotics Research 20, 616 (2001)
19. Spenko, M.A., Haynes, G.C., Saunders, A., Rizzi, A.A., Cutkosky, M., Full, R.J., Koditschek, D.E.: Biologically inspired climbing with a hexapedal robot. Journal of Field Robotics 25(4-5) (2008)
20. Weingarten, J.D., Groff, R.E., Koditschek, D.E.: A framework for the coordination of legged robot gaits. In: IEEE Conference on Robotics, Automation and Mechatronics (2004)
21. Wickler, S., Hoyt, D., Cogger, E., Myers, G.: The energetics of the trot-gallop transition. Journal of Experimental Biology 206, 1557–1564 (2003)
22. Wilson, D.M.: Insect walking. Annual Review of Entomology 11, 103–122 (1966)

Appendix A: The Gait Complex and Its Defining Inclusions

The gait complex $\mathsf{Gaits}_n[\mathbb{T}]$ is a cellular decomposition of $\mathbb{T}^n$ built upon the image of $\mathbb{T}^{n+1}$ under the "shearing map" [3],

$$s^{n+1} : \mathbb{T}^{n+1} \to \mathbb{T}^n : (r_1, \ldots, r_n, r_{n+1}) \mapsto (r_1 r_{n+1}^{-1}, \ldots, r_n r_{n+1}^{-1}). \qquad (4)$$

The cells of $\mathsf{Gaits}_n[\mathbb{T}]$ arise by "shearing" down all of the "diagonal" subspaces of $\mathbb{T}^{n+1}$—that is, all of those orbits wherein some subset of entries maintain the identical phase—and thus represent via a single n-tuple, an "orbit" of $(n+1)$-tuples that circulate while maintaining the same relative phase. [3]

For example, the bipedal steady state gaits may be coarsely distinguished by whether or not the legs are held in the same phase ("pronking") during circulation through periodic stride. Following the account at the beginning of Section 2.2, we use the Young Tabloid $\mathcal{T} = \boxed{1}\boxed{2} \in \mathfrak{T}_1^2$ to index the bipedal "Pronk" by defining a parametrization of the appropriate diagonal,

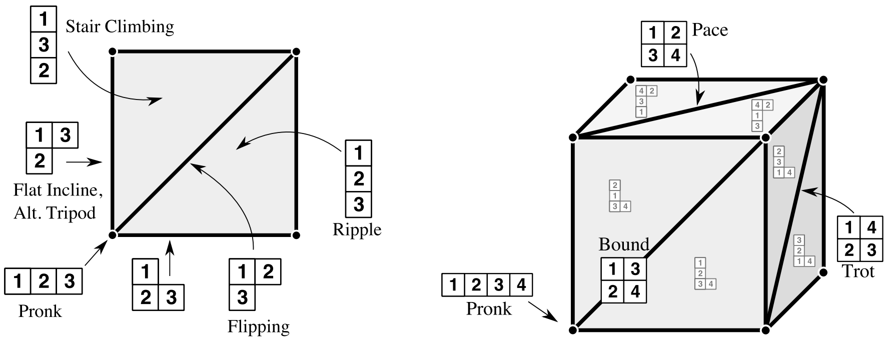
$$p_{\boxed{1}\boxed{2}} : \mathbb{T}^1 \to \mathbb{T}^2 : r \mapsto (r, r); P_{\boxed{1}\boxed{2}} := Dp_{\boxed{1}\boxed{2}} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}; \quad p^{\dagger}_{\boxed{1}\boxed{2}}(r_1, r_2) := r_1, \tag{5}$$

(that we display along with its Jacobian matrix, $P_{\boxed{1}\boxed{2}}$, and a choice of left inverse, $p^{\dagger}_{\boxed{1}\boxed{2}}$, made apparent in the discussion of Section 2.2) which is then "sheared down" to get the representative cell, $\mathsf{Gaits}_1[\mathbb{T}]^0 = \left\{s^2 \circ p_{\boxed{1}\boxed{2}}(\mathbb{T}^1)\right\}$—in this case, the single"vertex" $s^2 \circ p_{\boxed{1}\boxed{2}}(r) = e^{2\pi i 0}$. In contrast, if the legs are out of phase in steady state, then we locate the gait in the sheared image of the identity map of the two-torus to get $\mathsf{Gaits}_1[\mathbb{T}]^1 = \left\{s^2 \circ p_{\mathcal{T}}(\mathbb{T}^2)\right\}$ for $\mathcal{T} = \begin{array}{|c|}\hline 1 \\ \hline 2 \\ \hline \end{array} \in \mathfrak{T}_2^2$, which evaluates to the entire "circle" $s^2 \circ p_{\mathcal{T}}(r_1, r_2) = e(2\pi i(x_1 - x_2))$. The Delta Complex formalism [6] requires these inclusions be made from the domain of a simplex,

$$\begin{aligned} \Delta[m] := \{(x_1, x_1 + x_2, \ldots, x_1 + \ldots + x_m) \in [0,1]^m : \\ 0 \le x_1 + \ldots + x_j \le 1, j = 1, \ldots m\} \end{aligned}$$

and given a Young Tabloid, $\mathcal{T} \in \mathfrak{T}_{k+1}^{n+1}$ with toral inclusion, $p_{\mathcal{T}} : \mathbb{T}^{k+1} \to \mathbb{T}^{n+1}$, the associated "characteristic function" is $\tilde{p}_{\mathcal{T}} := p_{\mathcal{T}} \circ exp^{k+1}$ where $exp^m : \Delta[m] \to \mathbb{T}^m : (x_1, \ldots, x_m) \mapsto (e^{2\pi i x_1}, \ldots, e^{2\pi i x_m})$. Continuing along this specific example, for $m = n + 1 = 2$, the Delta Complex formalism now re-assembles these two sets—the zero-skeleton, $\mathsf{Gaits}_1[\mathbb{T}]^0$, and the one-skeleton $\mathsf{Gaits}_1[\mathbb{T}]^1$—into the cell complex, $\mathsf{Gaits}_1[\mathbb{T}] = \mathsf{Gaits}_1[\mathbb{T}]^0 \coprod \mathsf{Gaits}_1[\mathbb{T}]^1$ by identifying their shared point $e^{2\pi 0}$ via the characteristic maps as detailed in [2]. The corresponding generalization of this construction to the 3-legged gait complex is listed in Table 1 and depicted in Figure 8 which also provides a view of the 4-legged gait complex.

[3] We will shift back and forth as a matter of convenience between representing phase as an "angle" x, or as a point on the unit circle in the complex plane, $r = e^{2\pi i x}$, where e is the standard exponential map.



(a) $\mathsf{Gaits}_2[\mathbb{T}]$ from $s^3 \circ p_T : \mathbb{T}^3 \to \mathbb{T}^2$ (b) $\mathsf{Gaits}_3[\mathbb{T}]$ from $s^4 \circ p_T : \mathbb{T}^4 \to \mathbb{T}^3$

Fig. 8 An exhaustive view of the cells of the "three-legged" gait complex (left) and a typical view of the "four-legged" gait complex (right) annotated by their associated Young Tabloids, T, and examples of some of the gaits they represent.

Table 1 The Gait Complex $\mathsf{Gaits}_2[\mathbb{T}]$

Tabloid	InclusionMaps	NormalMap	Gait
$T \in \mathcal{T}^3_{k+1}$	$\Delta[k+1] \xrightarrow{\tilde{p}_T} \mathbb{T}^3 \xrightarrow{s^3} \mathbb{T}^2$	$n_T(r_1, r_2, r_3)$	Name
1 2 3	$(x) \mapsto (e^{2\pi i x}, e^{2\pi i x}) \mapsto (e^{2\pi i 0})$	$(r_1 r_3^{-1}, r_2 r_3^{-1})$	Pronk
1 2 3	$(x_1, x_1 + x_2) \mapsto (e^{2\pi i x_1}, e^{2\pi i x_1}, e^{2\pi i (x_1 + x_2)}) \mapsto (e^{-2\pi i x_2}, e^{-2\pi i x_2})$	$r_1 r_2^{-1}$	Reverse Flip
1 3 2	$(x_1, x_1 + x_2) \mapsto (e^{2\pi i x_1}, e^{2\pi i (x_1 + x_2)}, e^{2\pi i x_1}) \mapsto (e^{2\pi i 0}, e^{2\pi i x_2})$	$r_1 r_3^{-1}$	Incline
2 3 1	$(x_1, x_1 + x_2) \mapsto (e^{2\pi i (x_1 + x_2)}, e^{2\pi i x_1}, e^{2\pi i x_1}) \mapsto (e^{2\pi i x_2}, e^{2\pi i 0})$	$r_2 r_3^{-1}$	Flip
1 2 3	$(x_1, x_1 + x_2, x_1 + x_2 + x_3) \mapsto$ $(e^{2\pi i x_1}, e^{2\pi i (x_1 + x_2)}, e^{2\pi i (x_1 + x_2 + x_3)}) \mapsto (e^{-2\pi i (x_2 + x_3)}, e^{-2\pi i x_3})$	$\emptyset$	Ripple
1 3 2	$(x_1, x_1 + x_2, x_1 + x_2 + x_3) \mapsto$ $(e^{2\pi i x_1}, e^{2\pi i (x_1 + x_2 + x_3)}, e^{2\pi i (x_1 + x_2)}) \mapsto (e^{-2\pi i x_2}, e^{2\pi i x_3})$	$\emptyset$	StairClimbing

Table 2 Gait Fields over Limb Phase Coordinates.

Gait	Field	Formula	Change of Coordinates
Bipedal Pronk [18]	R^2_{PR}	$P_{\boxed{1}\boxed{2}} \cdot R^1_{BC} \circ p^{\dagger}{}_{\boxed{1}\boxed{2}} - \text{grad}\,\nu_{\boxed{1}\boxed{2}}$	$\emptyset$
Alternating Phase Biped [18]	R^2_{AP}	$Dh^2_{TR} \cdot R^2_{PR} \circ \left(h^2_{TR}\right)^{-1}$	$(x_1, x_2) \mapsto (x_1, x_2 + \pi)$
Three-Legged Pronk [13]	R^3_{PR}	$P_{\boxed{1}\boxed{2}\boxed{3}} \cdot R^1_{BC} \circ p^{\dagger}{}_{\boxed{1}\boxed{2}\boxed{3}} - \text{grad}\,\nu_{\boxed{1}\boxed{2}\boxed{3}}$	$\emptyset$
Three-Legged Crawl [14]	R^3_{CR}	$Dh^3_{TR+} \cdot R^3_{PR} \circ \left(h^3_{TR+}\right)^{-1}$	$(x_1, x_2, x_3) \mapsto (x_1, x_2 - \frac{4\pi}{3}, x_3 - \frac{2\pi}{3})$
Hexapedal Alternating Tripod [18]	R^6_{AP}	$P_{\begin{array}{ccc}\boxed{1}&\boxed{4}&\boxed{5}\\\boxed{2}&\boxed{3}&\boxed{6}\end{array}} \cdot R^2_{AP} \circ p^{\dagger}{}_{\begin{array}{ccc}\boxed{1}&\boxed{4}&\boxed{5}\\\boxed{2}&\boxed{3}&\boxed{6}\end{array}} - \text{grad}\,\nu_{\begin{array}{ccc}\boxed{1}&\boxed{4}&\boxed{5}\\\boxed{2}&\boxed{3}&\boxed{6}\end{array}}$	$\emptyset$
Hexapedal Stair Gait [13]	R^6_{Stair}	$P_{\begin{array}{cc}\boxed{1}&\boxed{2}\\\boxed{3}&\boxed{4}\\\boxed{5}&\boxed{6}\end{array}} \cdot R^3_{CR} \circ p^{\dagger}{}_{\begin{array}{cc}\boxed{1}&\boxed{2}\\\boxed{3}&\boxed{4}\\\boxed{5}&\boxed{6}\end{array}} - \text{grad}\,\nu_{\begin{array}{cc}\boxed{1}&\boxed{2}\\\boxed{3}&\boxed{4}\\\boxed{5}&\boxed{6}\end{array}}$	$\emptyset$

Stable Dynamic Walking over Rough Terrain

Theory and Experiment

Ian R. Manchester, Uwe Mettin, Fumiya Iida, and Russ Tedrake

Abstract. We propose a constructive control design for stabilization of non-periodic trajectories of underactuated mechanical systems. An important example of such a system is an underactuated "dynamic walking" biped robot walking over rough terrain. The proposed technique is to compute a transverse linearization about the desired motion: a linear impulsive system which locally represents dynamics about a target trajectory. This system is then exponentially stabilized using a modified receding-horizon control design. The proposed method is experimentally verified using a compass-gait walker: a two-degree-of-freedom biped with hip actuation but pointed stilt-like feet. The technique is, however, very general and can be applied to higher degree-of-freedom robots over arbitrary terrain and other impulsive mechanical systems.

1 Introduction

It has long been a goal of roboticists to build a realistic humanoid robot. Clearly, one of the most fundamental abilities such a robot must have is to walk around its environment in a stable, efficient, and naturalistic manner.

When one examines the current state of the art, it seems that one can have either *stability and versatility* or *efficiency and naturalism*, but not all four. This paper reports some recent efforts to bridge this gap.

I. R. Manchester · R. Tedrake
Computer Science and Artificial Intelligence Lab, Massachusetts Institute of Technology, Cambridge MA, 02139, USA
e-mail: irm@mit.edu

U. Mettin
Department of Applied Physics and Electronics, Umeå University, SE-901 87 Umeå, Sweden

F. Iida
Institute of Robotics and Intelligent Systems, Department of Mechanical and Process Engineering, ETH Zurich, CH-8092 Zurich, Switzerland

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 123–138.
springerlink.com

We propose a general method of exponentially stabilizing arbitrary motions of underactuated mechanical systems. In particular, we develop a provably-stable feedback control strategy for efficient "dynamic walking" bipeds over uneven terrain, and demonstrate experimentally that the method is feasible and effective.

1.1 Bipedal Walking Robots

The world of bipedal walking robots can be divided into two broad classes. The first, including well-known robots such as the Honda ASIMO and the HRP-2, are based on the "zero moment point" (ZMP) principle (see, e.g., [1] and references therein). The main principle of stability and control is that the center of pressure always remains within the polygon of the stance foot, and so the foot always remains firmly planted on the ground. Satisfaction of this principle ensures that all dynamical degrees of freedom remain fully actuated at all times, and thus control design can be performed systematically using standard tools in robotics. However, the motions which are achievable are highly conservative, inefficient, and unnatural looking.

The second broad class consists of passive-dynamic walkers and limit-cycle walkers. Inspired by the completely passive walkers of McGeer [2], these robots forgo full actuation and allow gravity and the natural dynamics to play a large part in the generation of motion. They may be completely passive, or partially actuated. Even with partial actuation, the motions generated can be life-like and highly efficient energetically [3]. However, there is presently a lack of tools for systematic control design and systems analysis.

Comparatively little work has been done as yet on walking over rough terrain, especially for underactuated dynamic walkers. The problem of footstep planning has been approached using computational optimal control [4] and experimental studies have shown that a minimalistic open-loop control can achieve stability for the compass-gait walker [5]. Recently, more complete planning and control systems have been developed for quadruped walkers: see, e.g., [6].

To give the present paper context, in Fig. 1 we depict a possible organizational structure for the perception and control of a dynamic walker on rough terrain. The main components are:

1. **Terrain Perception:** fusion of sensors such as vision, radar, and laser, perhaps combined with pre-defined maps, generating a model of the terrain ahead.
2. **Motion Planning:** uses the terrain map, current robot state, and a model of the robot's dynamics to plan a finite-horizon feasible sequence of footstep locations and joint trajectories. Slow time-scale: motion plan might be updated once per footstep.
3. **Motion Control:** feedback control to stabilize the planned motion in the face of inaccurate modelling, disturbances, time delays, etc. Fast time-scale: typically of the order of milliseconds.
4. **Robot State Sensing:** optical encoders, accelerometers, gyros, foot pressure sensors, and so on. Provides local information about the physical state of the robot to all other modules.

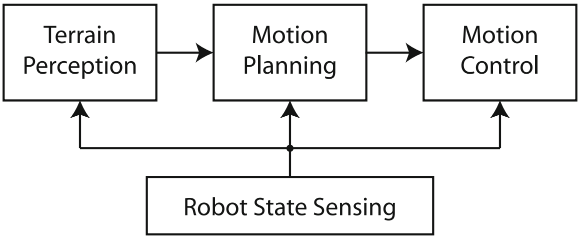


Fig. 1 Possible organization of perception and control of a walking robot.

A complete humanoid robot would have all these components, and many others. In this paper, we focus our attention on component 3: motion control. That is, we assume that the terrain has been sensed and a motion plan generated, and the task remaining is to compute a stabilizing feedback controller which achieves this motion.

1.2 *Motion Control for Walking Robots*

The problem of motion control of a compass-gait walker has been approached via energy-shaping and passivity-based control techniques (see, e.g., [7, 8, 9]). However, it is not clear how such methods can be extended to robots with more degrees of freedom, or to walking on uneven terrain.

Most tools for underactuated walking make use of Poincaré-map analysis (see, e.g., [10, 11, 12, 13, 14, 15] and many others). For a mechanical system of state dimension $2n$, one constructs a *Poincaré section*: a $(2n-1)$-dimensional surface transverse to the orbit under study (e.g. $S(0)$ in Fig. 2). By studying the behaviour of the system only at times at which it passes through this surface, one obtains a $(2n-1)$-dimensional discrete time system, the *Poincaré map*:

$$x_\perp(k+1) = \mathscr{P}[x_\perp(k)], \;\; x_\perp(\cdot) \in \mathbb{R}^{2n-1}$$

which has a fixed point at the periodic orbit: $\mathscr{P}[x_\perp^\star] = x_\perp^\star$. Stability or instability of this reduced system corresponds to orbital stability and orbital instability, respectively, of the periodic orbit. Exponential orbital stability corresponds to all the eigenvalues of the linearization of $\mathscr{P}$ being inside the unit circle.

One disadvantage of the Poincaré map is that it does not give a continuous representation of the dynamics of the system transverse to the target orbit, but focuses only at *one* point on the orbit. This means it has limited use for constructive control design.

However, the biggest problem for the present study is that the method of Poincaré sections is only defined for periodic orbits. It can be used to study biped walking on flat ground or constant slopes, but on uneven ground where we have no reasonable expectation of periodic orbits it is not applicable.

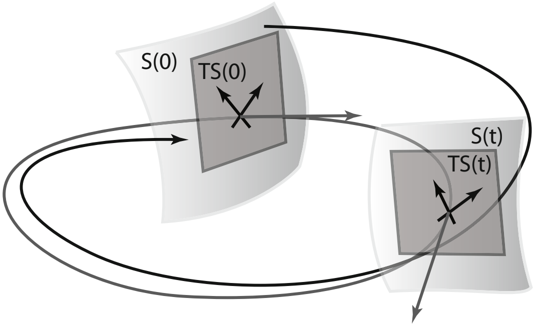


Fig. 2 A visualization of Poincaré surfaces and transverse linearization of a periodic orbit (grey) and a trajectory converging to it (black).

With this as motivation, in this work we use instead the *transverse linearization* of the target trajectory, which has previously been used for analysis and stabilization of *periodic* motions of nonlinear systems including walking robots [16, 17, 18, 19, 20].

This can be visualized via the related concept of a *moving Poincaré section*, introduced in [21]. This is a continuous family of $(2n-1)$-dimensional surfaces transverse to the desired trajectory, with one member of the family present at *every* point along the cycle ($S(t)$ for all t in Fig. 2).

In contrast to the classical Poincaré map, a transverse linearization (or moving Poincaré section) provides a continuous representation of the relationship between controls and transverse coordinates, and can be extended to the study of non-periodic motions.

Stabilizing only the transverse dynamics — as opposed to, e.g., a full-order linear time-varying (LTV) approximation [22] — is particularly useful for underactuated systems, which are often weakly controllable in the direction along the trajectory.

In this paper, we make use of this, and propose a computationally feasible feedback control strategy for the time-varying impulsive linear system that results. Successful experiments demonstrate the feasibility of our approach.

2 Impulsive Mechanical Systems

The mathematical model we consider is that of a nonlinear mechanical system subject to instantaneous impacts. Let q be a vector of generalized coordinates, and u be a vector of forces and torques which can be assigned, then the dynamics of the system can be written like so [23, 24, 14]:

$$\begin{aligned} &M(q)\,\ddot{q}+C(q,\dot{q})\,\dot{q}+G(q)=B(q)\,u \qquad \text{for} \quad q\notin\mathcal{Q} \\ &\left.\begin{aligned} q^{+}&=\Delta_q\, q^{-} \\ \dot{q}^{+}&=\Delta_{\dot{q}}(q^{-})\,\dot{q}^{-} \end{aligned}\right\} \qquad \text{whenever} \quad q^{-}\in\mathcal{Q}\,, \end{aligned} \tag{1}$$

where $M(q)$ is the inertia matrix, $C(q,\dot{q})$ is the matrix of Coriolis and centrifugal terms, $G(q)$ is the gradient of the potential energy field, and $B(q)$ describes the effects of actuators on the generalized coordinates. The set $\mathscr{Q}$ represents switching surfaces, e.g. for a walking robot, states at which the foot of the swing-leg hits the ground, and a new step begins.

2.1 Representation of a Planned Motion

Consider an n-degree-of-freedom impulsive mechanical system for which some desired and feasible trajectory has been specified:

$$q(t) = q^{\star}(t) \in \mathbb{R}^n,\ t \in [0,\infty).$$

Let $t_j,\ j = 1,2,\ldots$ be the time moments at which an impact occurs, and let $\mathscr{I}_j := [t_j,t_{j+1}),\ j = 1,2,\ldots$ be the time intervals of smooth behaviour in between impulses, and let $\mathscr{I}_0 := [0,t_1)$.

Assumption 1. *There exists $\tau_1 > \tau_2 > 0$ such that $\tau_1 \geq t_{j+1} - t_j \geq \tau_2$ for all j.*

That is, the footsteps do not get infinitely long or infinitely short.

Assumption 2. *For all t_j, the vector $[\dot{q}^{\star}(t)^T \ddot{q}^{\star}(t)^T]^T$ is linearly independent of the $2n-1$ vectors spanning the tangent plane of the switching surface at $q^{\star}(t)$.*

That is, all impacts are "real" impacts, not grazing touches of the switching surface.

For each interval $\mathscr{I}_j,\ j = 0,1,2,\ldots$, choose one generalized coordinate or some scalar function of the generalized coordinates $\theta := \Theta_j(q)$ which evolves monotonically along a desired trajectory.

Remark 1. In the case of the compass-gait walker, which we will consider in Sections 5 and 6 we will take θ to be the "ankle angle". It is a reasonable assumption that for any useful walking motion, this angle evolves monotonically over any given step. This representation is common in walking robot control [14]. □

Since it evolves monotonically θ can then be considered as a reparametrization of time, and hence the nominal trajectories of all other coordinates over each interval $\mathscr{I}_j$ can be given as well-defined functions of θ:

$$\begin{aligned} q_1^{\star}(t) &= \phi_1^j(\theta(t)), \\ &\ \vdots \\ q_n^{\star}(t) &= \phi_n^j(\theta(t)) \quad \forall\, t \in \mathscr{I}_j. \end{aligned}$$

Having thus defined the functions $\phi_1^j,\ldots,\phi_n^j$, one can define variables representing deviations from the nominal trajectory:

$$
\begin{aligned}
y_1(t) &:= q_1(t) - \phi_1^j(\theta(t)),\\
&\vdots\\
y_n(t) &:= q_n(t) - \phi_n^j(\theta(t)) \quad \forall\, t \in \mathscr{I}_j,
\end{aligned}
$$

where $y_m(t) = 0$ for all m implies the system is on the nominal trajectory.

Consider now the quantities $\theta, y_1, ..., y_n$. These $n+1$ quantities are excessive coordinates for the system, and hence one can be dropped. Without loss of generality, let us assume we drop y_n, and our new coordinates are $y = [y_1, ..., y_{n-1}]^T$ and θ.

Remark 2. When the conditions $y_m = 0$ for all m are enforced via feedback action, the functions $\phi_1^j, .., \phi_n^j$ are often referred to as *virtual holonomic constraints* [14, 11, 12]. Our control strategy does not require that these constraint be strictly enforced to guarantee stability, they are simply used as a set of coordinates. However, we retain the terminology "virtual constraints". □

3 Construction of the Transverse Linearization

A mechanical system's dynamics along a target motion can be decomposed into two components: a scalar variable θ representing the position *along* the target motion, and a vector $x_\perp$ of dimension $2n-1$ representing the dynamics *transverse* to the target motion.

A transverse linearization is a time-varying linear system representing of the dynamics of $x_\perp$ close to the target motion. The stabilization of the transverse linearization implies local exponential orbital stabilization of the original nonlinear system to the target motion [16]. The construction of a transverse linearization for an impulsive mechanical system such as a walking robot can be broken down into two parts: the continuous phases and the impacts maps.

3.1 Construction of the Continuous Part of the Transverse Linearization

A method for analytical construction of a transverse linearization for continuous mechanical systems was proposed in [25, 18]. We will use this construction for the continuous phases of the desired walking motion.

The representation of trajectories introduced in the previous section allows us to analytically construct, at any θ, a set of transversal coordinates without solving the nonlinear differential equations of the system. The first $2n-2$ coordinates are given by the coordinates y given in (2), and their derivatives:

$$
\dot{y}_i = \dot{q}_i - \frac{d\phi_i^j(\theta)}{d\theta}\dot{\theta}, \quad i = 1, ..., n-1
$$

defined in each continuous interval $\mathscr{I}_j$. For a full set of transverse coordinates, one more independent coordinate is required.

For the continuous phase of an underactuated mechanical system, if the relations $y = 0$ are maintained then the dynamics of the coordinate θ take the following form:

$$\alpha(\theta)\ddot{\theta} + \beta(\theta)\dot{\theta}^2 + \gamma(\theta) = 0 \tag{2}$$

where $\alpha(\cdot), \beta(\cdot), \gamma(\cdot)$ are straightforward to compute. An important fact is that a partial closed-form solution of the system (2) can be computed:

$$\dot{\theta}^2 = \psi_p(\theta, \theta_0)\dot{\theta}_0^2 + \Gamma(\theta, \theta_0)$$

where $(\theta_0, \dot{\theta}_0)$ is any point on the desired trajectory of the reduced system (2). That is, $\dot{\theta}^2$ can be computed as a function of θ, analytically.

The variable

$$I = \dot{\theta}^2 - \psi_p(\theta, \theta_0)\dot{\theta}_0^2 - \Gamma(\theta, \theta_0)$$

is then a clear candidate for the final transverse coordinate: it is independent of y and $\dot{y}$ and is zero when the system is on the target motion.

Our complete set of transverse coordinates are then: $x_\perp := \begin{bmatrix} I, & y^T, & \dot{y}^T \end{bmatrix}^T$.

Now, for systems of underactuation degree one, there exists a partial feedback-linearizing transformation of the form [26]:

$$u = N(y, \theta)^{-1}[v - W(y, \theta, \dot{y}, \dot{\theta})]$$

creating the dynamics $\ddot{y} = v$, $\dot{I} = f(\theta, \dot{\theta}, y, \dot{y}, v)$ where f can be calculated analytically.

From this we construct the continuous part of the transverse linearization, with z representing the state of the linearization of the dynamics of $x_\perp$:

$$\dot{z}(t) = A(t)z(t) + B(t)v(t) \tag{3}$$

where $A(t)$ and $B(t)$ are

$$A(t) = \begin{bmatrix} a_{11}(t) & a_{12}(t) & a_{13}(t) \\ 0_{(n-1)} & 0_{(n-1)^2} & I_{(n-1)} \\ 0_{(n-1)} & 0_{(n-1)^2} & 0_{(n-1)^2} \end{bmatrix} B = \begin{bmatrix} b_1(t) \\ 0_{(n-1)^2} \\ I_{(n-1)} \end{bmatrix} \tag{4}$$

where $I_{(n-1)}$ is the $(n-1)$-dimensional identity matrix, $0_{(n-1)}$ is an $(n-1)$-dimensional column of zeros, and $0_{(n-1)^2}$ is an $(n-1) \times (n-1)$ matrix of zeros. The functions $N(y, \theta), R(y, \theta, \dot{y}, \dot{\theta}), a_{11}(t), a_{12}(t), a_{13}(t)$, and $b_1(t)$ can be computed analytically, see [25, 18].

3.2 Transverse Linearization of Impacts

Certain care is required in linearizing the impact map. The transversal surfaces are orthogonal in phase space to the target motion, but the switching surfaces will not be in general. Therefore, we must also introduce two projection operators.

Suppose $d\Delta_j$ is the linearization of the impact map at time t_j about the nominal trajectory, then

$$\begin{aligned} z(t^+) &= F_j z(t) \ \text{ for } \ t = t_j, j = 1,2,\ldots \\ \text{where } \ F_j &= P_j^+ d\Delta_j P_j^- \end{aligned} \tag{5}$$

The construction of P_j^+ and P_j^- is given in [20].

The complete hybrid transverse linearization system is given by (3), (4), and (5).

Assumption 3. *The hybrid transverse linearization system is uniformly completely controllable.*

This assumption essentially states that there is sufficient dynamical coupling between the unactuated and actuated links of the system. It is always satisfied with reasonable walking robot designs.

4 Receding-Horizon Control Design

Exponential stabilization of time-varying systems, even linear systems, is a nontrivial problem. For time-invariant or periodic linear systems one can compute constant or periodic gain matrices, respectively, which exponentially stabilize the system. This is not true in general for time-varying systems. A common technique which is computationally feasible is receding-horizon control, also known as model predictive control (see, e.g., [27, 22] and many others). In this section we describe a slightly modified version of receding-horizon control suitable for impulsive linear systems.

The basic strategy is to repeatedly solve a constrained-final-time linear quadratic optimal control problem for the system (3), (4), (5). That is, minimize the following cost function:

$$J(x,u) = \int_{t_i}^{t_f} [z(t)^T Q(t) z(t) + v(t)^T R(t) v(t)]dt + \sum_{j=1}^{N_j} z(t_j)^T Q_j z(t_j)$$

subject to the constraint $z(t_f) = 0$.

Assumption 4. *There exists $\alpha_i > 0$, $i = 0,..,4$, such that $\alpha_0 I \leq Q(t) \leq \alpha_1 I, \alpha_2 I \leq R(t) \leq \alpha_3 I$, and $0 \leq Q_j \leq \alpha_4 I$ for all t and j.*

The traditional approach is to set a constant time horizon, but in this work we choose to look a fixed *number of footsteps* ahead[1]. Thus, every time the robot takes a footstep, a new optimization is computed.

We propose the following receding-horizon strategy, looking h footsteps ahead, beginning with $i = 0$:

[1] Or, more generally, a fixed number of impulses ahead.

1. Consider the union of intervals $\mathscr{I}_{i,h} := \mathscr{I}_i \cup \mathscr{I}_{i+1} \cup \ldots \cup \mathscr{I}_{i+h}$. Let t_i and t_f denote the beginning and end times of $\mathscr{I}_{i,h}$.
2. Compute this footstep's optimal control by solving following jump-Riccati equation backwards in time from t_f to t_i with a final condition $Z(t_f) = 0_{(n-1)^2}$

$$\begin{aligned} -\dot{Z} &= -ZA^T - AZ + BR^{-1}B^T - ZQZ \\ Z(t_j) &= \left\{F_j^T Z(t_j^+)^{-1} F_j + Q_j\right\}^{-1} \text{ for } t \in \mathscr{T}_j \end{aligned} \tag{6}$$

3. Over the interval $\mathscr{I}_i$, apply the following state-feedback controller:

$$\begin{aligned} u(y,\theta,\dot{y},\dot{\theta}) &= N(y,\theta)^{-1}[K(\theta)x_\perp(y,\theta,\dot{y},\dot{\theta}) - W(y,\theta,\dot{y},\dot{\theta})], \\ K(\theta) &= -R^{-1}(s)B(s)^T Z(s)^{-1}, s = \Theta^{-1}(\theta) \end{aligned} \tag{7}$$

   where $\Theta^{-1} : [\theta_+, \theta_-] \rightarrow [t_i, t_{i+1})$ is a projection operator, which is straightforward to construct since θ is monotonic over each step.
4. for the next footstep, set $i = i+1$ and return to stage 1.

Theorem 1. *If Assumptions 1, 2, 3, and 4 are satisfied, then the controller (6), (7) locally exponentially orbitally stabilizes the planned motion of the original nonlinear system.*

The proof is given in Appendix A.

5 Experimental Setup

We have constructed a two-degree-of-freedom planar biped robot with point feet, a photograph and schematic of which are shown in Fig. 3. The robot is mounted on a boom arm with a counterweight, and thus walks in a circular path. The dynamical effect of the boom is approximated by having different values of hip mass for inertial (m_H) and gravitational (m_{Hg}) terms in the model. The robot is fitted with retractable

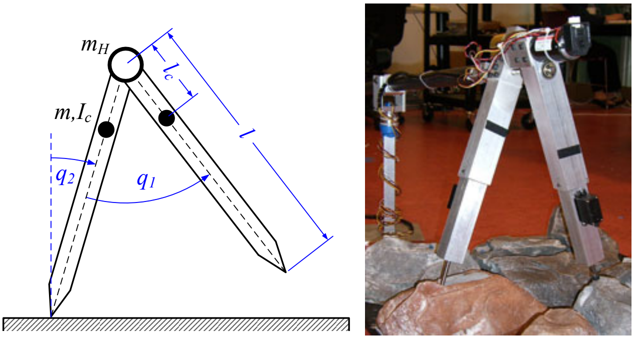


Fig. 3 Schematic and photograph of the experimental setup

feet to avoid toe-scuffing, however we do not consider the feet as additional degrees of freedom since their masses are negligible.

The robot is modelled in the form of an impulsive mechanical system (1), the parameters of which were estimated via nonlinear system identification. The full equations for the model are given in Appendix B. Good fitting required the addition of a friction model for the hip joint consisting of Coulomb and viscous parts:

$$\tau_F = F_C \,\mathrm{sign}(\dot{q}_1) + F_V \dot{q}_1.$$

The parameters of the model are given in Table 1.

The robot is fitted with optical encoders measuring the angle between the legs and the absolute angle of the inner leg. From these measurements q_1 and q_2 can be calculated. The control law relies on velocities as well, and these are estimated with an observer. The observer structure we choose is one which has previously been used successfully in walking robot control, consisting of a copy of the nonlinear dynamics and a linear correction term [28]. Let $\hat{q}$ and $\hat{\dot{q}}$ be the estimates of the configuration states and velocities, then the observer is given by:

$$\frac{d}{dt}\begin{bmatrix}\hat{q}\\ \hat{\dot{q}}\end{bmatrix} = \begin{bmatrix}\hat{\dot{q}}\\ M(\hat{q})^{-1}(-C(\hat{q},\hat{\dot{q}})\hat{\dot{q}} - G(\hat{q}) + B(\hat{q})u)\end{bmatrix} + L(y-\hat{q}),$$
$$\hat{q}^+ = \Delta_q \hat{q}, \quad \hat{\dot{q}}^+ = \Delta_{\dot{q}}(\hat{q})\hat{\dot{q}},$$

where y is the measurement of q. The gain L can be chosen as $L = [1/\varepsilon \;\; 2/\varepsilon^2]$ placing the eigenvalues of the linearized error system at $-1/\varepsilon$. In our experiments we found that $\varepsilon = 0.02$ gave a reasonable compromise between speed of convergence and noise rejection.

Table 1 Parameters of the compass-gait biped.

Parameters	Values
Masses [kg]	$m = 1.3$, $m_H = 2.2$, $m_{Hg} = -1.2$
Inertia [kg m^2]	$I_c = 0.0168$
Lengths [m]	$l = 0.32$, $l_c = l - 0.0596$
Gravitational constant [m/s^2]	$g = 9.81$
Ratio current/input [A]	$k_I = 1.1$
Motor torque constant [Nm/A]	$k_\tau = 0.0671$
Coulomb friction [Nm]	$F_C = 0.02$
Viscous friction [Nm s]	$F_V = 0.01$

5.1 Polynomial Representation of Desired Motion

For the compass biped we take $\theta = q_2$, the “ankle” angle of the stance leg relative to horizontal. Then to specify the path through configuration space for each step j, we need to specify only the inter-leg angle q_1 as a function of the ankle angle: $q_1^\star = \phi^j(\theta)$. We chose to construct the ϕ^j functions as fourth-order Bézier polynomials, which can represent a wide range of useful motions with quite a low number of parameters, and furthermore admit simple representations of the constraints in the previous section. For details, see [14, Ch. 6], in which Bézier polynomials were used to design periodic trajectories. It is straightforward to extended this method to non-periodic trajectories and because of space restrictions we omit the details.

6 Experimental Results

To test the controller experimentally, a relatively simple task was chosen: the robot should walk flat for two steps, then down two "stairs", and then continue along the flat. A video of a successful experiment has been placed online [29].

The control design was implemented as in Section 4 with constant weighting matrices $Q(t) = Q_j = I_3$ and $R(t) = 1$ for all t and j. The look-ahead horizon was chosen as three footsteps ahead.

For each step, the solution of the jump-Riccati equation took approximately half a second to compute using the `ode45` solver in MATLAB running on a Pentium III desktop computer. This is roughly the time it takes for the robot to complete a step, and it is reasonable to expect that highly optimised C code could perform this task much more quickly. Hence, one can say that the control law could be feasibly computed in real-time, as a part of a dynamic motion-planning and control system.

Figure 4 depicts the results of one experiment. Figure 4(A) is a cartoon of the biped's motion generated from real data, showing the state every 0.3 seconds, with the current stance leg always indicated in red.

In Fig. 4(B) the evolution of the "ankle angle" q_2 is plotted vs time for one experiment. During the continuous phases, q_2 serves as our reparametrization of time θ. We note here that, particularly on the second and fourth steps, there is some jitter in the curve.

In Fig. 4(C) the inter-leg angle q_1 is plotted vs time in blue, along with the "nominal" value of q_1 plotted in red. Note that, since the nominal value of q_1 is not a function of time but a function of q_2, defined by the virtual constraint, the jitters in the q_2 measurement lead to jitters in the nominal value of q_1. Nevertheless, tracking is quite good, and sufficient for the robot to maintain a stable walking trajectory.

Figures 4(D) and (E) depict the joint velocities $\dot{q}_1$ and $\dot{q}_2$, obtained from the same observer used in the control system, along with their nominal values as functions of the current value of q_2. Again, the jitter in q_2 leads to large noise in the velocity estimates. Despite this, good tracking is maintained through all the planned steps. Repeated experiments were performed with similar results each time, indicating good robustness of the control strategy.

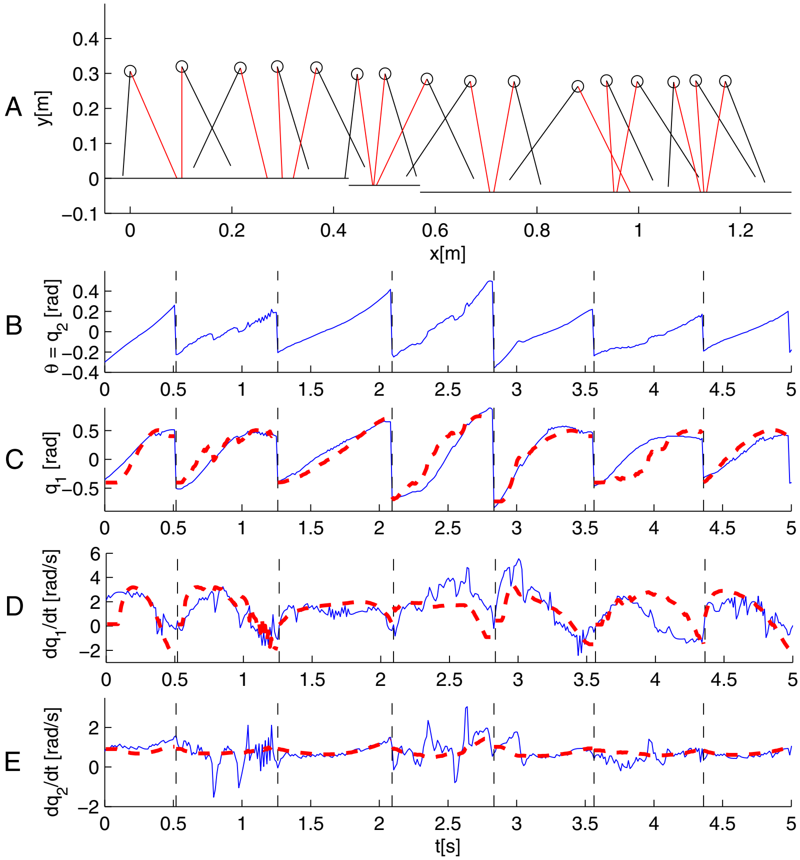


Fig. 4 Results from a successful experiment walking on uneven terrain. See Section 6 for discussion.

7 Conclusions

In this paper we have described a novel method for stabilization of trajectories of impulsive mechanical systems. The method guarantees local exponential stability to a target orbit under reasonable assumptions. The method is quite general, but a clear target application is motions of underactuated walking robots on rough and uneven terrain. To the authors' knowledge, this is the first systematic control method which can provably stabilize such motions.

The proposed technique was experimentally verified using a compass-gait biped walker. It was seen that, despite measurement errors and inevitable uncertainties in modelling, the controller reliably stabilized the target motions. The method of transverse linearization can be applied to any "dynamic walking" robot to design stabilizing controllers, or to give certificates of stability and assist choice of gains for existing control laws.

Future work will include application to robots with more degrees of freedom on more challenging terrain, and computation of basins of attraction. Furthermore, the theoretical tools developed in this work also have application to problems of motion planning, which will be explored.

References

1. Vukobratovic, M., Borovac, B.: Zero-moment point – thirty five years of its life. International Journal of Humanoid Robotics 1(1), 157–173 (2004)
2. McGeer, T.: Passive dynamic walking. International Journal of Robotics Research 9(2), 62–82 (1990)
3. Collins, S., Ruina, A., Tedrake, R., Wisse, M.: Efficient bipedal robots based on passive-dynamic walkers. Science 307(5712), 1082–1085 (2005)
4. Byl, K., Tedrake, R.: Approximate optimal control of the compass gait on rough terrain. In: Proc. of the IEEE International Conference on Robotics and Automation, Pasadena, CA (2008)
5. Iida, F., Tedrake, R.: Minimalistic control of a compass gait robot in rough terrain. In: Proc. of the IEEE International Conference on Robotics and Automation, Kobe, Japan (2009)
6. Byl, K., Shkolnik, A., Roy, N., Tedrake, R.: Reliable dynamic motions for a stiff quadruped. In: Proc. of the 11th International Symposium on Experimental Robotics (ISER), thens, Greece (2008)
7. Goswami, A., Espiau, B., Keramane, A.: Limit cycles in a passive compass gait biped and passivity-mimicking control laws. Autonomous Robots 4(3), 273–286 (1997)
8. Asano, F., Yamakita, M., Kamamichi, N., Luo, Z.W.: A novel gait generation for biped walking robots based on mechanical energy constraint. IEEE Transactions on Robotics and Automation 20(3), 565–573 (2004)
9. Spong, M., Holm, J., Lee, D.: Passivity-based control of bipedal locomotion. IEEE Robotics and Automation Magazine 14(2), 30–40 (2007)
10. Hurmuzlu, Y., Moskowitz, G.: The role of impact in the stability of bipedal locomotion. Dynamics and Stability of Systems 1, 217–234 (1986)
11. Grizzle, J., Abba, G., Plestan, F.: Asymptotically stable walking for biped robots: Analysis via systems with impulse effects. IEEE Transactions on Automatic Control 46(1), 51–64 (2001)
12. Westervelt, E., Grizzle, J., Koditschek, D.: Hybrid zero dynamics of planar biped walkers. IEEE Transactions on Automatic Control 48(1), 42–56 (2003)
13. Wisse, M., Schwab, A., van der Linde, R., van der Helm, F.: How to keep from falling forward: elementary swing leg action for passive dynamic walkers. IEEE Transactions on Robotics 21(3), 393–401 (2005)
14. Westervelt, E., Grizzle, J., Chevallereau, C., Choi, J., Morris, B.: Feedback Control of Dynamic Bipedal Robot Locomotion. CRC Press, Boca Raton (2007)
15. Hobbelen, D., Wisse, M.: A disturbance rejection measure for limit cycle walkers: The gait sensitivity norm. IEEE Transactions on Robotics 23(6), 1213–1224 (2007)
16. Hauser, J., Chung, C.: Converse Lyapunov function for exponential stable periodic orbits. Systems and Control Letters 23, 27–34 (1994)
17. Banaszuk, A., Hauser, J.: Feedback linearization of transverse dynamics for periodic orbits. Systems and Control Letters 26, 95–105 (1995)

18. Shiriaev, A.S., Freidovich, L.B., Manchester, I.R.: Can we make a robot ballerina perform a pirouette? Orbital stabilization of periodic motions of underactuated mechanical systems. Annual Reviews in Control 32(2), 200–211 (2008)
19. Freidovich, L.B., Shiriaev, A.S., Manchester, I.R.: Stability analysis and control design for an underactuated walking robot via computation of a transverse linearization. In: Proceedings of the 17th IFAC World Congress, Seoul, Korea, July 6-11 (2008)
20. Shiriaev, A.S., Freidovich, L.B., Manchester, I.R.: Periodic motion planning and analytical computation of transverse linearizations for hybrid mechanical systems. In: Proceedings of the 47th IEEE Conference on Decision and Control, Cancun, Mexico (2008)
21. Leonov, G.: Generalization of the Andronov-Vitt theorem. Regular and chaotic dynamics 11(2), 281–289 (2006)
22. Mayne, D.Q., Rawlings, J.B., Rao, C.V., Scokaert, P.O.M.: Constrained model predictive control: Stability and optimality. Automatica 36(6), 789–814 (2000)
23. Spong, M., Hutchinson, S., Vidyasagar, M.: Robot Modeling and Control. John Wiley and Sons, New Jersey (2006)
24. Hurmuzlu, Y., Marghitu, D.: Rigid body collisions of planar kinematic chains with multiple contact points. The International Journal of Robotics Research 13(1), 82–92 (1994)
25. Shiriaev, A., Perram, J., Canudas-de-Wit, C.: Constructive tool for orbital stabilization of underactuated nonlinear systems: virtual constraints approach. IEEE Transactions on Automatic Control 50(8), 1164–1176 (2005)
26. Spong, M.: Partial feedback linearization of underactuated mechanical systems. In: Proc. of International Conference on Intelligent Robots and Systems, Munich, Germany (2004)
27. Kwon, W., Bruckstein, A., Kailath, T.: Stabilizing state-feedback design via the moving horizon method. International Journal of Control 37(3), 631–643 (1983)
28. Grizzle, J., Choi, J., Hammouri, H., Morris, B.: On observer-based feedback stabilization of periodic orbits in bipedal locomotion. In: Proc. Methods and Models in Automation and Robotics (2007)
29. http://groups.csail.mit.edu/locomotion/movies/cgexperiment20081222_4.mov.
30. Kalman, R.: Contributions to the theory of optimal control. Bol. Soc. Mat. Mexicana 5, 102–119 (1960)
31. Bainov, D., Simeonov, P.: Systems with Impulse Effects: Stability, Theory and Applications. Ellis Horwood, Chichester (1989)

Appendix A: Proof of Local Stability

Some details are omitted to save space. We consider the Lyapunov function candidate

$$\mathscr{V}(x(t),t,\mathscr{K}) = x(t)^T P(t) x(t)$$

where $P(t) = X(t)^{-1}$, the solution of the finite-time jump-Riccati equation. i.e. the total "cost-to-go" from a state $x(t)$ with a feedback strategy $\mathscr{K}$ defining $u(t) = -K(t)x(t)$

We state the following three facts about this Lyapunov function candidate:

1. It follows from Assumptions 1, 3, and 4 and standard arguments from optimal control [30] that there exists $\beta_1 > \beta_0 > 0$ such that

$$\beta_0 I \leq P(t) \leq \beta_1 I. \tag{8}$$

2. Throughout the continuous phase from t_i to t_{i+1},

$$\frac{d}{dt}\mathscr{V}(x(t),t,\mathscr{K}_i) = -x(t)^T[Q(t) + K(t)^T B(t)^T R(t) B(t) K(t)]x(t) < 0.$$

Therefore, it follows from the bounds on $Q(t)$ in Assumption 4 that

$$\frac{d}{dt}\mathscr{V}(x(t),t,\mathscr{K}_i) \leq \alpha_0 \|x(t)\| \tag{9}$$

for all t.

3. Let $\mathscr{K}_i$ refer to the strategy of using finite-time controller calculated at the beginning of step i. Under this strategy, $x(t_{i+h}) = 0$ and remains zero for all $t > t_{i+h}$. After step i, the state is $x(t_{i+1})$. A feasible strategy from here would be to continue with control strategy $\mathscr{K}_i$. However, a new optimization is performed at step $i+1$ over a new horizon $i+1+h$. Since continuing with $\mathscr{K}_i$ is a feasible strategy, the new optimal strategy $\mathscr{K}_{i+1}$ must have a cost to go

$$\mathscr{V}(x(t_{i+1}),t_{i+1},\mathscr{K}_{i+1}) \leq \mathscr{V}(x(t_{i+1}),t_{i+1},\mathscr{K}_i). \tag{10}$$

i.e. the Lyapunov function is non-increasing when an impulse occurs.

From the facts (8), (9), and (10) it follows that the time-varying impulsive linear comparison system (3), (4), (5) is exponentially stable, using a generalization of Lyapunov's second method [31, Ch. 13].

Using Assumptions 1, 2, and 3 arguments similar to those of [25, 18, 20] prove that exponential stability of the transverse linearization implies local orbital exponential stability of the original nonlinear system to the target trajectory.

Appendix B: Compass Biped Model

The model of the experimental setup is given by (1) where

$$
\begin{aligned}
M(q) &= \begin{bmatrix} p_1 & -p_1+\cos(q_1)p_2 \\ -p_1+\cos(q_1)p_2 & p_3+2p_1-2\cos(q_1)p_2 \end{bmatrix}, \\
C(q,\dot q) &= \begin{bmatrix} 0 & -\dot q_2\sin(q_1)p_2 \\ -\sin(q_1)(\dot q_1-\dot q_2)p_2 & \sin(q_1)\dot q_1 p_2 \end{bmatrix}, \quad B = \begin{bmatrix} k_I k_\tau \\ 0 \end{bmatrix}, \\
G(q) &= \begin{bmatrix} \sin(-q_2+q_1)p_4, & -\sin(-q_2+q_1)p_4-\sin(q_2)p_5-\sin(q_2)p_4 \end{bmatrix}^T .
\end{aligned}
$$

The coefficients are defined by the physical parameters of the robot like so:

$$
\begin{aligned}
&p_1 = (l-l_c)^2 m + I_c\,, \qquad p_2 = ml(l-l_c)\,, \qquad p_3 = m_H l^2 + 2mll_c \\
&p_4 = mg(l-l_c)\,, \qquad p_5 = g(m_{Hg} l + 2ml_c)\,.
\end{aligned}
$$

The impact model in (1) is derived under the assumption of having an instantaneous and inelastic collision of the swing leg with the ground and no occurrence of slip or rebound [24]:

$$
\Delta_q = \begin{bmatrix} -1 & 0 \\ -1 & 1 \end{bmatrix}, \quad \Delta_{\dot q}(q^-) = \Delta_q \left[H^+(q^-)\right]^{-1} H^-(q^-),
$$

where

$$
\begin{aligned}
H^+_{1,1}(q^-) &= p_1 - 2p_6 l + l^2 p_7, \\
H^+_{1,2}(q^-) &= H^+_{2,1}(q^-) \;=\; (l^2 p_7 + (-p_8-2p_6)l + p_2)\cos(q_1) - l^2 p_7 + 2p_6 l - p_1, \\
H^+_{2,2}(q^-) &= (-2l^2 p_7 + (2p_8+4p_6)l - 2p_2)\cos(q_1) + 2l^2 p_7 + (-4p_6-2p_8)l + p_3 \\
&\quad +2p_1, \\
H^-_{1,1}(q^-) &= p_1 - p_6 l, \qquad H^-_{1,2}(q^-) = ((-p_6-p_8)l + p_2)\cos(q_1) - p_1 + p_6 l, \\
H^-_{2,1}(q^-) &= (p_2 - p_6 l)\cos(q_1) - p_1 + p_6 l, \\
H^-_{2,2}(q^-) &= ((2p_6+p_8)l - 2p_2)\cos(q_1) + (-p_8-2p_6)l + p_3 + 2p_1 \\
p_6 &= m(l-l_c) \qquad p_7 = m_H + 2m, \qquad p_8 = l_c p_7 + m_H(l-l_c).
\end{aligned}
$$

Design and Analysis of Hybrid Systems, with Applications to Robotic Aerial Vehicles

Jeremy H. Gillula, Haomiao Huang, Michael P. Vitus, and Claire J. Tomlin

Abstract. Decomposing complex, highly nonlinear systems into aggregates of simpler hybrid modes has proven to be a very successful way of designing and controlling autonomous vehicles. Examples include the use of motion primitives for robotic motion planning and equivalently the use of discrete maneuvers for aggressive aircraft trajectory planning. In all of these approaches, it is extremely important to verify that transitions between modes are safe. In this paper, we present the use of a Hamilton-Jacobi differential game formulation for finding continuous reachable sets as a method of generating provably safe transitions through a sequence of modes for a quadrotor performing a backflip maneuver.

1 Introduction

As robotic and automated systems become more complex, it has become increasingly difficult to design and analyze these systems, and it is especially hard to provide provable guarantees on safety and performance. One successful approach is to break down complex nonlinear systems into a hybrid collection of discrete modes, with different continuous dynamics for each mode. This decomposition can greatly

Jeremy H. Gillula
Stanford University Computer Science Department, Stanford, California 94305
e-mail: jgillula@stanford.edu

Haomiao Huang · Michael P. Vitus
Stanford University Aeronautics and Astronautics Department, Stanford, California 94305
e-mail: haomiao@stanford.edu, vitus@stanford.edu

Claire J. Tomlin
UC Berkeley Electrical Engineering and Computer Science Department,
Berkeley, California 94720
e-mail: tomlin@eecs.berkeley.edu

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 139–149.
springerlink.com

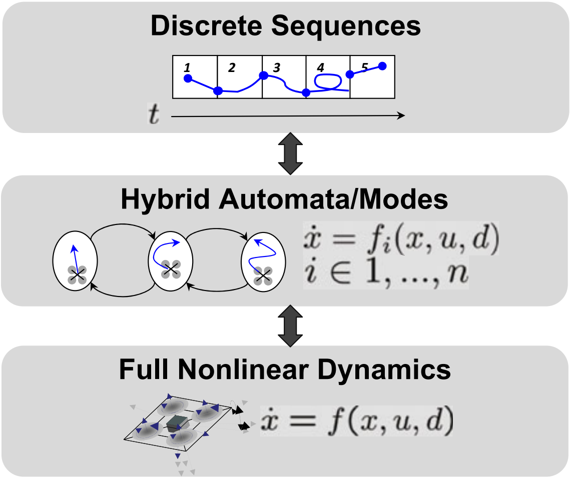


Fig. 1 Hybrid hierarchical representation of maneuvers for a quadrotor helicopter

simplify the analysis of the behavior of the overall system, and planning and control is also simplified by the ability to generate plans at the level of the discrete modes (see Figure 1). This hybrid, hierarchical approach to the design and control of autonomous systems has proven to be very powerful. Successful examples of this approach include aerobatic maneuver design (Frazzoli et al, 2005), linear-temporal logic specifications for generating robot behaviors (Kress-Gazit et al, 2008), and the use of motion primitives for robotic manipulator motion planning in complex dynamical tasks (Burridge et al, 1999).

A key consideration in the use of hybrid modes is the question of verifying that in transitioning between modes safety or performance criteria are met. For example, if constructing a sequence of maneuvers for an aircraft, it is necessary to know if one maneuver can be safely followed by another maneuver. In other words, an allowable grammar over the discrete modes must be constructed. Previous work has addressed this problem in a variety of ways. In the maneuver sequencing for helicopters, steady state ”trim states” were designated, with all maneuvers starting and ending in a trim state (e.g. level flight). Thus a maneuver that had a given end state could be followed by another maneuver with the same trim state as its initial condition (Frazzoli et al, 2005). The work on motion primitives proceeded similarly, with Lyapunov functions designated for each mode that guaranteed that the output of one action would be within the stable capture region of the subsequent action (Burridge et al, 1999).

In much of the existing literature, the particular methods to ensure continuity between modes have been specific to the application at hand. Moreover, in many cases, while ensuring feasibility of transitions with respect to the dynamics, the methodologies may require separate external mechanisms to meet safety criteria, such as avoiding obstacles. Ding, et. al. demonstrated the use of reachable sets in UAV refueling as a method for ensuring both safety relative to another aircraft and guaranteed arrival at a target state (Ding et al, 2008). In this work, we propose the use of reachable sets as a mechanism for combining both dynamic feasibility of switching and simultaneously the imposition of verifiable safety constraints on system trajectories for a quadrotor helicopter performing an aerobatic maneuver. We demonstrate the use of the Hamilton-Jacobi differential game formulation of reachable sets (Mitchell et al, 2005) to construct maneuvers that safely transition through a sequence of modes for a backflip maneuver, arriving at a target state while avoiding unsafe states en route.

The organization of this paper is as follows. Section 2 provides background on the theory of the Hamilton-Jacobi game formulation for generating reachable sets. Section 3 describes the dynamics of the quadrotor helicopter considered, and the details of the analysis of the backflip maneuver are described in Section 4. Finally, simulation results for the flip maneuver are presented in Section 5, with conclusions and future work in Section 6.

2 Backwards Reachable Sets

The backwards reachable sets in this work are generated according to the Hamilton-Jacobi game formulation as described in Mitchell et al (Mitchell et al, 2005). Two types of reachable sets are used: avoid sets, which are reachable sets generated to avoid undesired states, and capture sets, which are defined in order to reach certain desired states. The formulation will be summarized in the discussion of avoid sets, with the description of capture sets highlighting the differences.

2.1 Avoid Sets

The backwards reachable set $G(t)$ is defined to be the set of all states x such that, for any inputs u, the disturbance d can drive the system into a set G_0 in time t. The system dynamics are defined by

$$\dot{x} = f(x,u,d) \tag{1}$$

where x is the system state, u is the control input, and d is a disturbance input, where u and d are assumed to be constrained in some sets U and D, respectively. As detailed in (Mitchell et al, 2005), the boundary of the reachable set is defined by the solution to the modified Hamilton-Jacobi-Isaacs equation

$$-\frac{\partial J(x,t)}{\partial t} = \min\{0, \max_u \min_d \frac{\partial J(x,t)}{\partial x} f(x,u,d)\} \tag{2}$$

where the level set $J(x,0) = 0$ defines the initial undesired set G_0.

2.2 Capture Sets

The same principle behind the avoid set can also be used to reverse the role of the control and disturbance to generate capture sets. Given a desired target state region, the backwards reachable set can be calculated with the control input attempting to drive the state into the desired state region and the disturbance attempting to keep the state out. The capture set so generated is the set of all states such that for any possible action that the disturbance might take, the input will drive the state to the desired region in some time t. The formulation for the capture set is identical to that for the avoid set, except for reversing the roles of the input and disturbance. The conditions so derived are then

$$-\frac{\partial J(x,t)}{\partial t} = \min\{0, \min_u \max_d \frac{\partial J(x,t)}{\partial x} f(x,u,d)\} \tag{3}$$

It should be noted that this problem can be simplified even further if one has a desired control law $u(x)$; in this case the capture set is given by

$$-\frac{\partial J(x,t)}{\partial t} = \min\{0, \max_d \frac{\partial J(x,t)}{\partial x} f(x,u(x),d)\} \tag{4}$$

and is simply the set of all states such that for any possible disturbance, the given control law will drive the state to the desired region in some time t.

2.3 Maneuver Sequencing with Reachable Sets

Capture and avoid sets can be used to construct safe sequences of maneuvers. Starting with the final (target) set, the dynamics for the final (n^{th}) maneuver can be run backwards to generate a capture set for that maneuver. Then a target region can be selected within the capture set of the final maneuver as the target set for the previous ($n-1^{st}$) maneuver. Thus an initial condition within the capture set of the $n-1^{st}$ maneuver is guaranteed to arrive within the capture set of the n^{th} maneuver, allowing a safe switch into the n^{th} maneuver and eventual safe arrival at the final target set (see Figure 2). This process can be repeated for any number of desired maneuvers to identify a start region for the entire sequence. Safety considerations such as avoiding a particular unsafe set can be encoded either by choosing capture sets that avoid the unsafe regions of particular reach sets, or by generating reach-avoid sets that reach target sets while avoiding the unsafe sets.

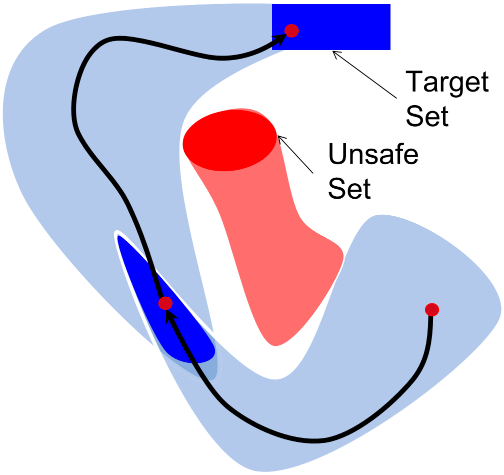


Fig. 2 Capture and avoid sets for sequencing two modes/maneuvers

3 Planar Quadrotor Dynamics

To simplify the problem the quadrotor's dynamics were modeled in a plane (as opposed to $\mathbb{R}^3$). It is assumed that the vehicle's out of plane dynamics can be stabilized, which we believe is a valid assumption. The resulting dynamics (based on the original dynamics in (Hoffmann et al, 2007)) are:

$$\frac{\partial}{\partial t}\begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \\ \theta \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ -\frac{1}{m}C_D^v\dot{x} \\ \dot{y} \\ -\frac{1}{m}\left(mg + C_D^v\dot{y}\right) \\ \dot{\theta} \\ -\frac{1}{Iyy}C_D^\theta\dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ D_x \\ 0 \\ D_y \\ 0 \\ D_\theta \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ -\frac{1}{m}\sin\theta & -\frac{1}{m}\sin\theta \\ 0 & 0 \\ \frac{1}{m}\cos\theta & \frac{1}{m}\cos\theta \\ 0 & 0 \\ -\frac{l}{Iyy} & \frac{l}{Iyy} \end{bmatrix}\begin{bmatrix} T_1 \\ T_2 \end{bmatrix} \tag{5}$$

where m is the vehicle's mass, g is gravity, C_D^v is the linear drag constant, C_D^θ is the rotational drag constant, D_x, D_y and D_θ are disturbances, I_{yy} is the moment of inertia, and all other variables are as depicted in figure 3. Several assumptions went into this formulation of the dynamics, including that the vehicle undergoes linear

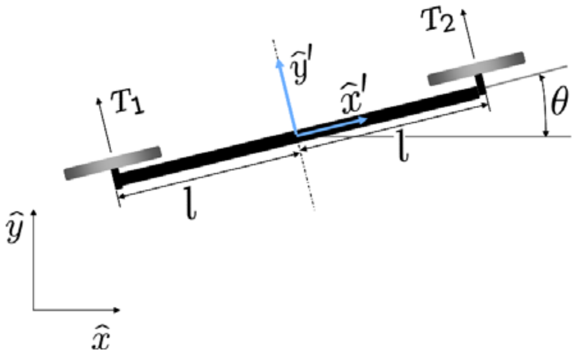


Fig. 3 The quadrotor's two-dimensional dynamics.

drag (as opposed to drag proportional to velocity), and that the thrust from each motor saturates at some value T_{max}.

4 Backflip

A diagram of the backflip maneuver is pictured in figure 4. The maneuver was broken down into three modes: impulse, drift, and recovery. (This break down was due to the fact that for at least part of the flip, the vehicle's motors must be turned off in order to prevent the vehicle from propelling itself into the ground.) In the impulse mode, a moment is applied to the vehicle that initiates the backflip. In the drift mode, the vehicle's motors are turned off and the vehicle completes the flip. Finally, in the recovery mode, the motors are turned back on and the vehicle is returned to a stable state (from which other maneuvers could potentially be initiated).

Of course, sequencing the maneuver in a manner that is guaranteed to be safe is a difficult problem: doing so requires hitting some target sets (e.g. a target set that ensures the vehicle is upside down) while avoiding some unsafe sets (e.g. the unsafe set consisting of states below $y = 0$). This problem is compounded by the fact that the quadrotor system dynamics are six-dimensional, and existing computational methods for computing reachable sets are only tractable for systems with

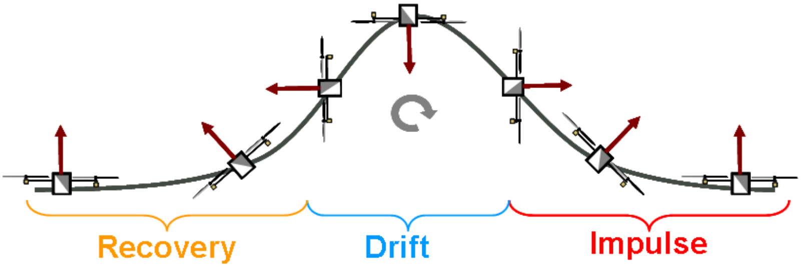


Fig. 4 The backflip maneuver, broken down into three modes.

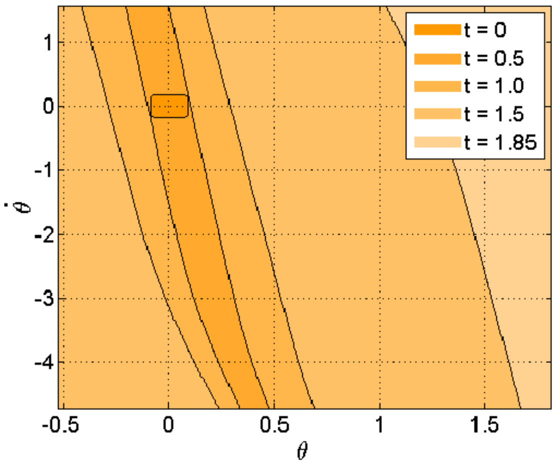


Fig. 5 The recovery mode target set and capture basin.

dimensions of four or less. To get past this problem, the system's states were broken apart apart into three sets and analyzed separately. The rotational dynamics (θ and $\dot{\theta}$) were analyzed to ensure that the vehicle achieved a flip; the vertical dynamics (y and $\dot{y}$) were analyzed to ensure the vehicle remained above some minimum altitude; and the horizontal dynamics (x and $\dot{x}$) were ignored to keep the problem as simple as possible.

4.1 Attainability

The general method for calculating the maneuver was as described in section 2. In particular, the target for the final state of the recovery mode was chosen to be $\theta = 0 \pm 5°$, $\dot{\theta} = 0 \pm 20°$, so as to have the vehicle end in a fairly level configuration with very little rotational velocity. Additionally, as mentioned in section 2, a fixed control law was chosen to drive the vehicle to this target set; in this case, a standard PD controller of the form $u = k_p\theta + k_d\dot{\theta}$ was used.[1] This target set was then propagated backwards using the reachable set toolbox in MATLAB, taking into account the worst-case disturbances (due to motor noise and wind). The resulting level sets represented the capture set for this maneuver, as pictured in figure 5.

For the drift mode, a similar procedure was followed. The target set was chosen as $\theta = 90 \pm 5°$, $\dot{\theta} = -138 \pm 20°$, and was propagated back (this time with no control input, and thus reduced worst-case disturbances due to the lack of motor noise) to produce the capture set for the drift mode (figure 6).

[1] It should be noted that the actual commanded thrust was of the form $T_1 = T_{nom} - u$, $T_2 = T_{nom} + u$, where T_{nom} was the nominal total thrust necessary to counteract gravity, and u was as given above.

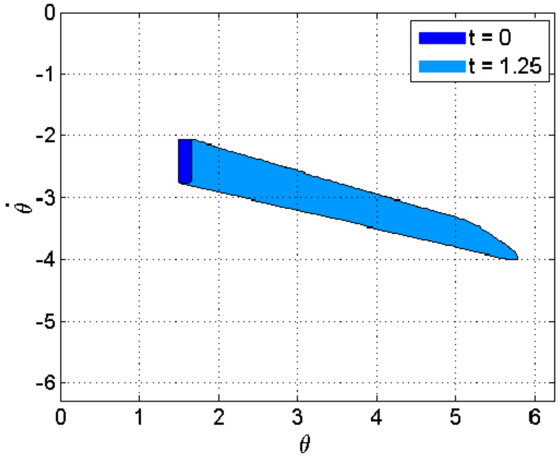


Fig. 6 The drift mode target set and capture basin.

Finally, for the impulse mode, the target set was $\theta = 300 \pm 5°$, $\dot{\theta} = -210 \pm 10°$. Once again, a fixed controller (of the form $u = k_p\theta + k_d\dot{\theta} + k_c$) was used, and the worst-case disturbances were chosen so as to account for motor noise and wind. The resulting capture set is pictured in figure 7.

4.2 Safety

To ensure safety, an initial unsafe set of $y < 0$ was chosen. Because the vehicle's vertical dynamics are coupled to its rotational dynamics when thrust is applied (see

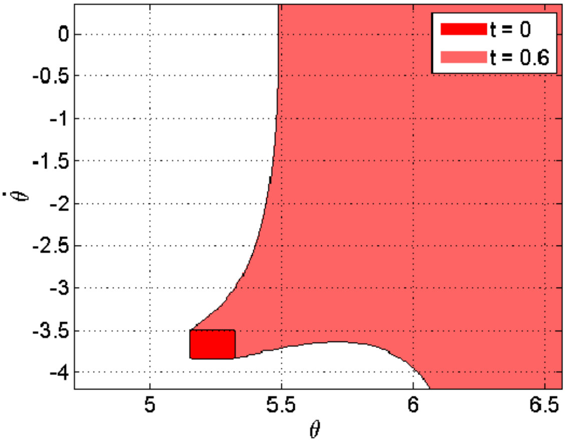


Fig. 7 The impulse mode target set and capture basin.

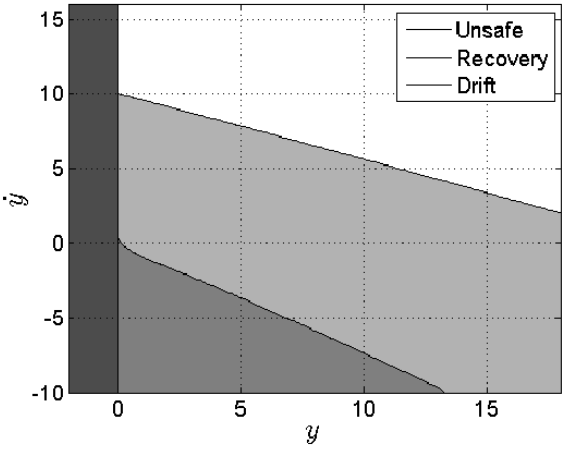


Fig. 8 The unsafe reachable sets in the vertical dynamics.

equation 5), the interaction between the two systems could not be ignored when trying to calculate the unsafe reachable set for this mode. However, because the recovery mode was designed with a fixed control law, a nominal trajectory in the θ, $\dot{\theta}$ space could be created as a function of y and $\dot{y}$. The unsafe set could then be calculated by plugging this nominal trajectory into the system dynamics, and proceeding as usual. The set was propagated backward for a fixed time T, based on the maximum time that the rotational part of the recovery mode could take.

In the drift mode, things are less complicated; the vehicle's dynamics decompose into three separate two-dimensional systems, given below.

$$\frac{\partial}{\partial t}\begin{bmatrix} x \\ \dot{x} \\ y \\ \dot{y} \\ \theta \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ -\frac{1}{m}C_D^v\dot{x} \\ \dot{y} \\ -\frac{1}{m}\left(mg + C_D^v\dot{y}\right) \\ \dot{\theta} \\ -\frac{1}{I_{yy}}C_D^\theta\dot{\theta} \end{bmatrix} + \begin{bmatrix} 0 \\ D_x \\ 0 \\ D_y \\ 0 \\ D_\theta \end{bmatrix} \tag{6}$$

Thus, it was easy to simply propagate the unsafe set from the recovery mode backwards using the y, $\dot{y}$ dynamics. Again, this was done for a fixed time based on the maximum length of the maneuver as calculated from the rotational dynamics. Finally, it was assumed that there would be no loss in altitude during the impulse mode because of the way the modes and their switching criteria were designed (i.e. the vehicle would never have a negative vertical velocity during the impulse mode). The resulting unsafe sets are pictured in figure 8; as long as the vehicle began each mode outside the unsafe set for that mode, the overall safety of the system was guaranteed.

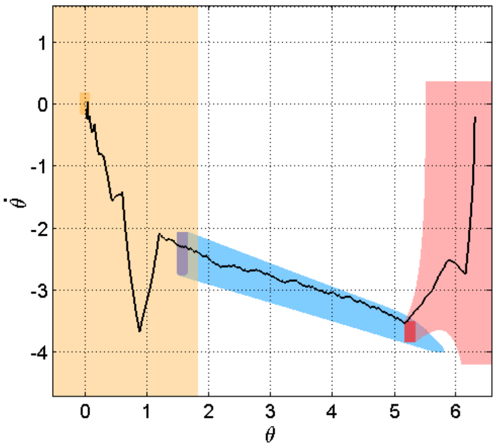


Fig. 9 A trajectory of the vehicle in the rotational state space. The yellow region is the capture basin and target for the recovery mode, blue is for the drift mode, and red is for the impulse mode.

5 Results

The combined results of the reachable set computations for the rotational state space are pictured in figure 9. Additionally, a sample trajectory in the rotational state space is overlaid, indicating that the vehicle does indeed remain inside each of the given capture basins as it completes the maneuver.

Figure 10 shows a time-lapse image of a resulting simulated trajectory. In this simulation, the transition between the different modes was triggered whenever both of the rotational states satisfied the conditions of the target set for the given mode.

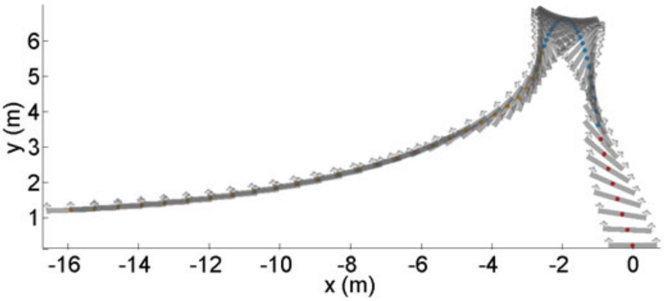


Fig. 10 A time-lapse image of a simulated trajectory.

6 Conclusions and Future Work

While the method we have proposed for using reachable sets to generate provably safe transitions between different modes shows great promise, several open questions remain. First, in future work we hope to explore how to parametrically describe reachable sets. In particular, it is apparent that in the current framework the resulting reachable set (whether it be for safety or attainability) depends a great deal on the parameters used when generating it. For example, the reachable set for a flip that ends with a slow rotational velocity would look very different from one that ends with a high rotational velocity. As a result, the reachable sets for each version of the same maneuver must be calculated offline using predefined parameters. Instead, it would be preferable if a reach set could be calculated and represented in such a way that the effect of a simple change in parameters could be quickly computed, resulting in the ability to choose between different versions of the same maneuver in an online manner.

Of course, the most immediate goal which we intend to accomplish is the implementation of the work described in this paper on an actual quadrotor vehicle. While this goal will likely entail a sizable amount of engineering work, we believe that due to the robustness of the theory we have developed, doing so is well within the realm of possibility.

References

Burridge, R., Rizzi, A., Koditschek, D.: Sequential composition of dynamically dexterous robot behaviors. International Journal of Robotics Research 18(6), 534–555 (1999)

Ding, J., Sprinkle, J., Sastry, S.S., Tomlin, C.J.: Reachability calculations for automated aerial refueling. In: Proceedings of the 47th IEEE Conference on Decision and Control, Cancun, Mexico (2008)

Frazzoli, E., Dahleh, M.A., Feron, E.: Maneuver-based motion planning for nonlinear systems with symmetries. IEEE Transactions on Robotics 21(6), 1077–1091 (2005)

Hoffmann, G.M., Huang, H., Waslander, S.L., Tomlin, C.J.: Quadrotor helicopter flight dynamics and control: Theory and experiment. In: Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit, Hilton Head, SC (2007)

Kress-Gazit, H., Fainekos, G.E., Pappas, G.J.: Translating structured english to robot controllers. In: Advanced Robotics Special Issue on Selected Papers from IROS 2007, vol. 22(12), pp. 1343–1359 (2008)

Mitchell, I.M., Bayen, A.M., Tomlin, C.J.: A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games. IEEE Transactions on Automatic Control 50(7), 947–957 (2005)

Motion Planning under Uncertainty for Robotic Tasks with Long Time Horizons

Hanna Kurniawati*, Yanzhu Du, David Hsu, and Wee Sun Lee

Abstract. Partially observable Markov decision processes (POMDPs) are a principled mathematical framework for planning under uncertainty, a crucial capability for reliable operation of autonomous robots. By using probabilistic sampling, point-based POMDP solvers have drastically improved the speed of POMDP planning, enabling POMDPs to handle moderately complex robotic tasks. However, robot motion planning tasks with long time horizons remain a severe obstacle for even the fastest point-based POMDP solvers today. This paper proposes *Milestone Guided Sampling* (MiGS), a new point-based POMDP solver, which exploits state space information to reduce the effective planning horizon. MiGS samples a set of points, called *milestones*, from a robot's state space, uses them to construct a simplified representation of the state space, and then uses this representation of the state space to guide sampling in the belief space. This strategy reduces the effective planning horizon, while still capturing the essential features of the belief space with a small number of sampled points. Preliminary results are very promising. We tested MiGS in simulation on several difficult POMDPs modeling distinct robotic tasks with long time horizons; they are impossible with the fastest point-based POMDP solvers today. MiGS solved them in a few minutes.

1 Introduction

Efficient motion planning with imperfect state information is an essential capability for autonomous robots to operate reliably in uncertain and dynamic environments.

Hanna Kurniawati
Singapore–MIT Alliance for Research Technology
e-mail: hannakur@smart.mit.edu

Yanzhu Du · David Hsu · Wee Sun Lee
Department of Computer Science, National University of Singapore
e-mail: {duyanzhu,dyhsu,leews}@comp.nus.edu.sg

* Most of the work was done while the author was with the Department of Computer Science, National University of Singapore.

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 151–168.
springerlink.com

With imperfect state information, a robot cannot decide the best actions on the basis of a single known state; instead, the best actions depend on the set of all possible states consistent with the available information, resulting in much higher computational complexity for planning the best actions. Partially observable Markov decision processes (POMDPs) [8, 18] provide a general and principled mathematical framework for such planning tasks. In a POMDP, we represent a set of possible states as a *belief*, which is a probability distribution over a robot's state space. We systematically reason over the belief space $\mathcal{B}$, the space of all beliefs, by taking into account uncertainty in robot control, sensor measurements, and environment changes, in order to choose the best robot actions and achieve robust performance. By incorporating uncertainty into planning, the POMDP approach has led to improved performance in a number of robotic tasks, including localization, coastal navigation, grasping, and target tracking [4, 7, 13, 16].

Despite its solid mathematical foundation, POMDP planning faces two major computational challenges. The first one is the "curse of dimensionality": a complex robotic task typically generates a high-dimensional belief space. If a robotic task is modeled with a discrete state space, its belief space has dimensionality equal to the number of states. Thus a task with 1,000 states has a 1,000-dimensional belief space! In recent years, point-based POMDP solvers [13] have made dramatic progress in overcoming this challenge by sampling the belief space and computing approximate solutions. Today, the fastest point-based POMDP solvers, such as HSVI2 [20] and SARSOP [10], can handle moderately complex robotic tasks modeled as POMDPs with up to 100,000 states in reasonable time. The success of point-based solvers can be largely attributed to *probabilistic sampling*, which allows us to use a small number of sampled points as an approximate representation of a high-dimensional belief space. The approximate representation substantially reduces computational complexity. The same reason underlies the success of probabilistic sampling in other related problems and approaches, e.g., probabilistic roadmap (PRM) algorithms [2] for geometric motion planning (without uncertainty).

The second major challenge is the "curse of history". In a motion planning task, a robot often needs to take many actions to reach the goal, resulting in a long time horizon for planning. Unfortunately the complexity of planning grows exponentially with the time horizon. Together, a long time horizon and a high-dimensional belief space compound the difficulty of planning under uncertainty. For this reason, even the best point-based POMDP algorithms today have significant difficulty with robotic tasks requiring long planning horizons (see Section 6 for examples).

To overcome this second challenge and scale up POMDP solvers for realistic robot motion planning tasks, we have developed a new point-based POMDP solver called *Milestone Guided Sampling* (MiGS). It is known from earlier work on related problems that the most important component of a planning algorithm based on probabilistic sampling is the sampling strategy [5]. MiGS reduces the planning horizon by constructing a more effective sampling strategy. It samples a set of points, called *milestones*, from a robot's state space S, uses the milestones to construct a simplified representation of S, and then uses this representation of S to guide sampling in the belief space $\mathcal{B}$. The intuition is that many paths in $\mathcal{B}$ are similar. Using the simplified representation of the state space, MiGS avoids exploring many of the similar belief space paths, which enables us to capture the essential features of $\mathcal{B}$ with a small number of sampled points from $\mathcal{B}$.

We tested MiGS in simulation on several difficult POMDPs modeling distinct robotic tasks with long time horizons, including navigation in 2D and 3D environments, and target finding. These tasks are impossible with the fastest point-based POMDP solvers today. MiGS solved them in a few minutes.

2 Background

2.1 Motion Planning under Uncertainty

Despite its importance and more than almost three decades of active research [11, 22], motion planning under uncertainty remains a challenge in robotics. Several recent successful algorithms are based on the probabilistic sampling approach. Stochastic Motion Roadmap [1] combines PRM with the Markov decision process (MDP) framework to handle uncertainty in robot control, but it does not take into account uncertainty in sensing. Another method, Belief Roadmap [15], handles uncertainty in both robot control and sensing, but one major limitation is the assumption that the uncertainty can be modeled as Gaussian distributions. Unimodal distributions such as the Gaussian distribution are inadequate when robots operate in complex geometric environments.

POMDPs are a general framework that can overcome the above limitations. By tackling the difficulty of long planning horizons, MiGS brings POMDPs a step closer to being practical for complex robotics tasks.

2.2 POMDPs

A POMDP models an agent taking a sequence of actions under uncertainty to maximize its reward. Formally, it is specified as a tuple $(S, A, O, T, Z, R, \gamma)$, where S is a set of states describing the agent and the environment, A is the set of actions that the agent may take, and O is the set of observations that the agent may receive.

At each time step, the agent lies in some state $s \in S$, takes some action $a \in A$, and moves from a start state s to an end state s'. Due to the uncertainty in action, the end state s' is modeled as a conditional probability function $T(s, a, s') = p(s'|s, a)$, which gives the probability that the agent lies in s', after taking action a in state s. The agent then receives an observation that provides information on its current state. Due to the uncertainty in observation, the observation result $o \in O$ is again modeled as a conditional probability function $Z(s, a, o) = p(o|s, a)$.

In each step, the agent receives a real-valued reward $R(s, a)$, if it takes action a in state s. The goal of the agent is to maximize its expected total reward by choosing a suitable sequence of actions. When the sequence of actions has infinite length, we typically specify a discount factor $\gamma \in (0, 1)$ so that the total reward is finite and the problem is well defined. In this case, the expected total reward is given by $\mathrm{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$, where s_t and a_t denote the agent's state and action at time t.

The solution to a POMDP is an *optimal policy* that maximizes the expected total reward. In a POMDP, the state is partially observable and not known exactly. So we rely on the concept of beliefs. A POMDP policy $\pi\colon \mathcal{B} \to A$ maps a belief $b \in \mathcal{B}$ to the prescribed action $a \in A$.

A policy π induces a value function $V_\pi(b) = \mathrm{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) | b, \pi\right]$ that specifies the expected total reward of executing policy π starting from b. It is known that V^*, the value function associated with the optimal policy π^*, can be approximated arbitrarily closely by a convex, piecewise-linear function,

$$V(b) = \max_{\alpha \in \Gamma} (\alpha \cdot b) \tag{1}$$

where Γ is a finite set of vectors called α-vectors and b is the discrete vector representation of a belief. Each α-vector is associated with an action. The policy can be executed by selecting the action corresponding to the best α-vector at the current belief. So a policy can be represented as a set of α-vectors.

Given a policy, represented as a set Γ of α-vectors, the control of the agent's actions, also called policy execution, is performed online in real time. It consists of two steps executed repeatedly. The first step is action selection. If the agent's current belief is b, it finds the action a that maximizes $V(b)$ by evaluating (1). The second step is belief update. After the agent takes an action a and receives an observation o, its new belief b' is given by

$$b'(s') = \tau(b, a, o) = \eta Z(s', a, o) \sum_{s} T(s, a, s') b(s) \tag{2}$$

where η is a normalization constant. The process then repeats.

2.3 Point-Based POMDP Solvers

The adoption of POMDPs as a planning framework in robotics has been hindered by the high dimensional belief space and the long time horizon typical of many robotics tasks. Many approaches have been proposed to alleviate these difficulties [22]. Point-based solvers [10, 13, 20, 21] are currently the most successful approach. Using the idea of probabilistic sampling, they have made impressive progress in computing approximate solutions for POMDPs with large number of states and have been successfully applied to a variety of non-trivial robotic tasks, including coastal navigation, grasping, target tracking, and exploration [4, 12, 13, 14, 19]. Despite this impressive progress, even the best point-based POMDP solvers today have significant difficulty with robotic tasks that require long planning horizons.

MiGS follows the approach of point-based POMDP solvers, but aims at overcoming the difficulty of long horizons. Learning from the successful PRM approach for geometric motion planning [2], MiGS tries to construct a more effective strategy for sampling the belief space.

3 Milestone Guided Sampling

A key idea of point-based POMDP solvers is to sample a set of points from $\mathcal{B}$ and use it as an approximate representation of $\mathcal{B}$. Let $\mathcal{R} \subseteq \mathcal{B}$ be the set of points reachable from a given initial belief point $b_0 \in \mathcal{B}$ under arbitrary sequences of actions and observations. Most of the recent point-based POMDP algorithms sample from $\mathcal{R}$ instead of $\mathcal{B}$ for computational efficiency. The sampled points form a belief tree $\mathcal{T}$ (Fig 1). Each node of $\mathcal{T}$ represents a sampled point $b \in \mathcal{B}$. The root of $\mathcal{T}$ is the initial belief point b_0. To sample a new point b', we pick a node b from $\mathcal{T}$ as well as an action $a \in A$ and an observation $o \in O$ according to suitable probability distributions or heuristics. We then compute $b' = \tau(b, a, o)$ using (2) and insert b' into $\mathcal{T}$ as a child of b. If a POMDP requires an effective planning horizon of h actions and observations, $\mathcal{T}$ may contain $\Theta((|A||O|)^h)$ nodes in the worst case, where $|A|$ is the number of actions and $|O|$ is the number of observations for the POMDP. Thus any point-based solvers trying to construct $\mathcal{T}$ exhaustively must have running time exponential in h and suffer from the "curse of history".

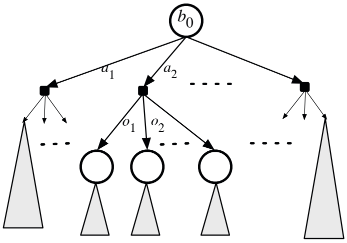


Fig. 1 The belief tree rooted at b_0.

To overcome this difficulty, let us consider the space from which we must sample. If the effective planning horizon is h, we must sample from a subset of $\mathcal{R}$ that contains $\mathcal{R}_h$, the set of belief points reachable from b_0 with at most h actions and observations. Our difficulty is that the size of $\mathcal{R}_h$ grows exponentially with h. The basic idea of MiGS is to sample $\mathcal{R}_h$ hierarchically at multiple resolutions and avoid exhaustively sampling $\mathcal{R}_h$ unless necessary.

To do so, MiGS builds a *roadmap* graph G in a robot's *state space* S. The nodes of G are states sampled from S and are called *milestones*. An edge e between two milestones s and s' of G is annotated with a sequence of actions $(a_1, a_2, \ldots, a_\ell)$ that can bring the robot from s to s'. The edge e is also annotated with a sequence of states $(s_0, s_1, \ldots, s_\ell)$ that the robot traverses under the actions $(a_1, a_2, \ldots, a_\ell)$, with $s_0 = s$ and $s_\ell = s'$. If we think of G as a collection of edges, each representing a sequence of actions, we can then use such sequences of actions to construct the belief tree $\mathcal{T}$. At a node b, we apply a *sequence of actions* associated with a selected edge of G, instead of a single action, to derive a child node b'. Suppose, for example, that G has maximum degree d and the minimum length of an action sequence contained in each edge of G is ℓ. Then, for a POMDP with time horizon h, $\mathcal{T}$ contains at most $\mathcal{O}((d\,|O|^\ell)^{h/\ell}) = \mathcal{O}(d^{h/\ell}|O|^h)$ nodes. This indicates that the action sequences encoded in G help in reducing the effect of long planning horizons due to actions, but not necessarily observations. Since the size of $\mathcal{T}$ grows exponentially with h, the reduction is nevertheless significant.

To sample at multiple resolutions, we sample S coarsely and connect the milestones with long sequences of actions, generating a large ℓ. We then refine the sampling of S, which gradually reduces the ℓ value.

Now it should be clear that MiGS is indeed faster, as the belief tree $\mathcal{T}$ is smaller. However, a more fundamental question remains: since the roadmap G contains only a subset of sampled states and not all states in the state space S, do the belief points sampled with the help of G cover the entire reachable belief space well and likely lead to a good approximation to the optimal value function and the optimal policy? The answer is yes, if we sample S adequately in a sense which we now explain.

Denote by Γ^* a set of α-vectors representing an optimal value function. Given a constant $\epsilon > 0$, we partition the state space S into a collection of disjoint subsets so that for any $\alpha \in \Gamma^*$ and any two states s and s' in the same subset, $|\alpha(s) - \alpha(s')| \leq \epsilon$. Intuitively, the partitioning condition means that any two states in the same subset are similar in terms of their significance in the optimal value function. The constant ϵ controls the resolution of partitioning. We call such a partitioning of S an ϵ-partitioning and denote it by $\mathcal{K}$. The partitioning $\mathcal{K}$ induces a distance metric on the belief space $\mathcal{B}$:

Definition 1. Let $\mathcal{K}$ be an ϵ-partitioning of the state space S. The distance between any two beliefs b and b' in $\mathcal{B}$ with respect to $\mathcal{K}$ is

$$d_{\mathcal{K}}(b, b') = \sum_{K \in \mathcal{K}} \left| \sum_{s \in K} b(s) - \sum_{s \in K} b'(s) \right|. \tag{3}$$

This new metric is more lenient than the usual L_1 metric and is upper-bounded by the L_1 metric. It measures the difference in probability mass for subsets of states rather than individual states. This is desirable, because the states within a subset $K \in \mathcal{K}$ are similar under our assumption and there is no need to distinguish them. Using $d_{\mathcal{K}}$, we can derive a Lipschitz condition on the optimal value function $V^*(b)$:

Theorem 1. *Let $\mathcal{K}$ be an ϵ-partitioning of the state space S. For any b and b' in the corresponding belief space $\mathcal{B}$, if $d_{\mathcal{K}}(b, b') \leq \delta$, then $|V^*(b) - V^*(b')| \leq \frac{R_{\max}}{1-\gamma}\delta + 2\epsilon$, where $R_{\max} = \max_{s \in \mathcal{S}, a \in \mathcal{A}} |R(s, a)|$.*

The proof is given in the appendix. Theorem 1 provides a sampling criterion for approximating V^* well. Suppose that B is a set of sampled beliefs that *covers* the belief space $\mathcal{B}$: for any $b \in \mathcal{B}$, there is a point b' in B with $d_{\mathcal{K}}(b, b') \leq \delta$, where δ is some positive constant. Theorem 1 implies that the values of V^* at the points in B serve as a good (sampled) approximation to V^*. Furthermore, to estimate these values, we do not need to consider all the states in S, because $d_{\mathcal{K}}$ does not distinguish states within the same subset $K \in \mathcal{K}$; it is sufficient to have one representative state from each subset. This justifies MiGS' sampling of the state space during the roadmap construction.

Of course, MiGS does not know the partitioning $\mathcal{K}$ in advance. To sample S adequately, one way is to use uniform random sampling. If each subset $K \in \mathcal{K}$ is sufficiently large, then we can guarantee that uniform sampling generates at least one sampled state from each $K \in \mathcal{K}$ with high probability. To improve efficiency, our implementation of MiGS uses a heuristic to sample S. See Section 4 for details.

We now give a sketch of the overall algorithm. MiGS iterates over two stages. In the first state, we sample a set of new milestones from S, and then use it to

Algorithm 1. Perform α-vector backup at a belief b.

```
BACKUP($b$, $\Gamma$)
For all $a \in A$, $o \in O$, $\alpha_{a,o} \leftarrow \text{argmax}_{\alpha \in \Gamma}(\alpha \cdot \tau(b,a,o))$.
For all $a \in A$, $s \in S$, $\alpha_a(s) \leftarrow R(s,a) + \gamma \sum_{o,s'} T(s,a,s')Z(s',a,o)\alpha_{a,o}(s')$.
$\alpha' \leftarrow \text{argmax}_{a \in \mathcal{A}}(\alpha_a \cdot b)$
Insert $\alpha'$ into $\Gamma$.
```

construct or refine a roadmap G. In the second stage, we follow the approach of point-based POMDP solvers and perform *value iteration* [17] on a set Γ of α-vectors, which represents a piecewise-linear lower-bound approximation to the optimal value function V^*. Exploiting the fact that V^* must satisfy the Bellman equation, value iteration starts with an initial approximation to V^* and performs backup operations on the approximation by iterating on the Bellman equation until the iteration converges. What is different in value iteration for point-based POMDP solvers is that backup operations are performed only at a set of sampled points from $\mathcal{B}$ rather than the entire $\mathcal{B}$. In MiGS, we sample incrementally a set of points from $\mathcal{B}$ by constructing a belief tree $\mathcal{T}$ rooted at an initial belief point b_0. To add a new node to $\mathcal{T}$, we first choose an existing node b in $\mathcal{T}$ in the least densely sampled region of $\mathcal{B}$, as this likely leads to sampled beliefs that cover $\mathcal{B}$ well. We then choose a suitable edge e from the roadmap G and use the associated action sequence $(a_1, a_2, \ldots, a_\ell)$ and state sequence $(s_0, s_1, s_2, \ldots, s_\ell)$, where ℓ is the length of the action associated with e, to generate a new node. Specifically, we first generate an observation sequence $(o_1, o_2, \ldots, o_\ell)$ so that each o_i is consistent with s_i and a_i, to be precise, $Z(s_i, a_i, o_i) = p(o_i|s_i, a_i) > 0$, for $1 \leq i \leq \ell$. We then start at b and apply the action-observation sequence $(a_1, o_1, a_2, o_2, \ldots, a_\ell, o_\ell)$ to generate a sequence of new beliefs $(b_1, b_2, \ldots, b_\ell)$, where $b_1 = \tau(b, a_1, o_1)$ and $b_i = \tau(b_{i-1}, a_{i-1}, o_{i-1})$ for $2 \leq i \leq \ell$. Finally, b_ℓ is inserted to $\mathcal{T}$ as a child node of b, while $(b_1, b_2, \ldots, b_{\ell-1})$ is associated with the edge from b to b_ℓ for backup operations. After creating the new node b_ℓ, we perform backup operations for every belief associated with the nodes and edges of $\mathcal{T}$ along the path from b_ℓ to the root b_0. A backup operation at b improves the approximation of $V^*(b)$ by looking ahead one step. We perform the standard α-vector backup (Algorithm 1). Each backup operation creates a new α-vector, which is added to Γ to improve the approximation of V^*. We repeat the sampling and backup processes until a sufficiently large number of new sampled beliefs are obtained. If necessary, we repeat the two stages to refine the roadmap G and sample additional new beliefs. The details for these two stages are reported in Sections 4 and 5, respectively.

4 Roadmap Construction

To construct the roadmap, MiGS assumes that positive reward is given only when the robot reaches the goal state. This assumption is suitable for general motion planning problem where the robot's goal is to reach one of the possible goal state(s).

4.1 Sampling the Milestones

Ideally, given ϵ, we would like to sample a set of milestones such that each milestone is a representative state of a partition of an ϵ-partitioning of the state space S. However since we do not know the optimal value function nor any ϵ-partitioning of S, MiGS uses a heuristic that biases sampling towards states where either high reward can be gained or informative observation can be perceived.

A particular distribution $P(s)$ MiGS uses to sample S is,

$$P(s) \propto K^{I(s)} \tag{4}$$

where K is a constant and $I(s)$ indicates the importance of visiting s in reducing uncertainty and gaining reward. The importance function we use, $I : S \rightarrow \mathbb{R}$, is a weighted sum of an expected reward function and a localization function, $avgReward(s) + \lambda \times locAbility(s)$, where the weight λ determines the importance of localization relative to the reward. Since localization depends on both the action performed and the observation perceived, we compute $locAbility(s)$ as an expected value over all possible actions and observations, assuming an action is selected uniformly at random and the observation is perceived according to the observation distribution function. More precisely, $locAbility(s) = \frac{1}{|A|}\sum_{a\in A}\sum_{o\in O} P(o|s,a) \cdot usefulness(s,o,a)$. To compute how useful an observation is towards localizing a state, we compute the posterior probability of being in the state after the observation is received, assuming a uniform prior on the states, $usefulness(s,a,o) = P(s|o,a) = P(o|s,a)/\sum_{s\in S} P(o|s,a)$. The reward component is $avgReward(s) = \frac{1}{|A|}\sum_{a\in A} R(s,a)$.

4.2 Partitioning the State Space

Once a set of milestones has been sampled, MiGS partitions the state space based on a notion of "distance" to the milestones. To help establish the notion of distance, MiGS constructs the state graph $\mathcal{S}$, a weighted multi-digraph where the vertices are states in S. We will refer to the vertices of $\mathcal{S}$ and their corresponding states interchangeably, as there is no confusion. An edge from $s \in S$ to $s' \in S$, labeled with action $a \in A$, exists in $\mathcal{S}$ whenever $T(s,a,s') > 0$. The weight of the edges act as the distance for partitioning.

We define the weight $w((\overline{ss'},a))$ of an edge $(\overline{ss'},a)$ in $\mathcal{S}$ as the sum between the cost for performing a from s and the expected total regret of continuing from s' even when the robot may be at a state other than s' after performing a from s. More precisely,

$$w((\overline{ss'},a)) = c(s,a) + \sum_{s''\in S} T(s,a,s'') \cdot r(s',s'') \tag{5}$$

where $c(s,a)$ is the cost of performing action a from s. For computational efficiency, we would like the weight to always be positive. Therefore, we set $c(s,a) = -R(s,a)$

if $R(s,a) < 0$ and $c(s,a) = 0$ otherwise. The second component indicates how different the future expected total reward can be if we assume the robot is in s' while it is actually in s''. The function $r(s', s'')$ can be defined in many ways. For simplicity, MiGS sets $r(s', s'')$ as a constant positive value whenever $s' \neq s''$ and 0 otherwise.

Once the distance between states is defined, Voronoi decomposition can be used to partition S. MiGS uses inward Voronoi partition [3] on the state graph $\mathcal{S}$ with M as the Voronoi sites. It partitions the vertices of $\mathcal{S}$ based on the distance *to* the Voronoi sites.

4.3 Roadmap Edges

To generate sequences of actions and states for guiding belief space sampling, MiGS uses the adjacency relation between the Voronoi sets in $\mathcal{S}$. An edge $\overline{mm'}$ from milestone m to milestone m' is inserted to the roadmap whenever there is a path from m to m' in the graph induced by $Vor(m) \cup Vor(m')$, where $Vor(m)$ is the Voronoi set of m. The edge $\overline{mm'}$ is annotated based on the shortest path $\Pi(m, m')$ from m to m' in the graph induced by $Vor(m) \cup Vor(m')$. The sequence of actions that annotates $\overline{mm'}$ is the action labels in the sequence of edges in $\Pi(m, m')$, while the sequence of states is the sequence of vertices in $\Pi(m, m')$. The weight of $\overline{mm'}$ is then the total weight of $\Pi(m, m')$.

4.4 Roadmap Refinement

The roadmap G can be refined by adding milestones or by adding edges to G. To decide when to refine G by adding milestones and when to refine by adding edges, we first describe the effect of the two refinement strategies in terms of reducing approximation error (Section 3). Adding milestones may reduce the two components, i.e., "resolution" of the simplified state space and density of belief space sampling, that contribute to the approximation error. Adding milestones refines the current state space partitioning, which may reduce the approximation error due to state space simplification. Furthermore, adding milestones increases the size of the beliefs that can be sampled, which may eventually increase the density of belief space sampling. On the other hand, adding edges only increases the size of the beliefs that can be sampled without refining the state space partitioning. Therefore, as a heuristic to quickly reduce the approximation error, MiGS refines the roadmap by adding milestones whenever possible, and only switch to adding edges when no more milestones can be added to the roadmap.

MiGS refines the roadmap whenever $V(b_0)$ does not improve after a consecutive pre-specified number of beliefs have been expanded and the corresponding backups have been performed, as it indicates that the roadmap may not be sufficient to generate a better approximation to the optimal policy.

5 Belief Space Sampling

So far we have discussed the construction of roadmap G, a simplified representation of the state space. Now, the question is how to use G to guide belief space sampling.

The goal of belief space sampling is to quickly sample a set of beliefs that is sufficient to approximates V^* well. Although such set of beliefs are not known a priori, it is known that point-based methods work well when the size of such set of beliefs is small [6]. Furthermore, in POMDP with long planning horizon, the sufficient set of beliefs would form a deep belief tree. Therefore, we assume that many sets of beliefs sufficient for approximating V^* well, forms belief trees with small branching factor. Furthermore, we assume that there is an abundance of such sets of beliefs where different sets correspond to different policies. Utilizing these assumptions, MiGS tries to quickly explore different sets of beliefs that correspond to different policies, by expanding b_0 using instantiations of different policies. An instantiation of a policy is an entire sequence of actions taken and states visited by the robot, during a single run using the policy.

Paths in G, from a support of b_0 to a goal state, are instantiations of different policies. Ideally, we would like to use paths that are instantiations of the best policy embedded in G, to expand b_0. However, we do not know which paths are instantiation of the best policy. Furthermore, the cost of expanding b_0 using a long path is expensive. If a path generates a sequence of beliefs near to the beliefs that have been sampled before, we would have wasted a lot of computational resources without improving the value function by much. To alleviate wasting a lot of computational resources, MiGS iteratively expand $\mathcal{T}$ using shorter partial paths. It expands a node of $\mathcal{T}$ using *edges* of G, but maintains a memory of which path is being used for expanding a particular branch of $\mathcal{T}$. For instance, if b' is a node of $\mathcal{T}$ generated by expanding b_0 using edge $\overline{mm'}$ of G, then b' is annotated with m'. The next time b' is selected for expansion, MiGS chooses an out-edge from m' in G to expand b'.

To decide which node b of $\mathcal{T}$ to expand, MiGS uses a simple heuristic based on the density of sampled beliefs around b. Since nearby beliefs have similar optimal value [6], when two or more nearby nodes of $\mathcal{T}$ lie close together, expanding one of the nodes would have similar effect as if all of the nodes are expanded. Therefore, MiGS sets the weight $w(b)$ of a node b in $\mathcal{T}$, to be the number of nodes in $\mathcal{T}$ that lie within a small pre-specified distance from b. And selects a node b for expansion based on the probability $P(b) \sim \frac{1}{w(b)}$.

6 Experimental Setup and Results

The purpose of our experiment is two folds. One (Section 6.1) is to compare the performance of MiGS with the best point-based POMDP solvers and other alternatives to motion planning with uncertainty. The other (Section 6.2) is to test the robustness of the generated policy.

6.1 Comparison with Other Planners

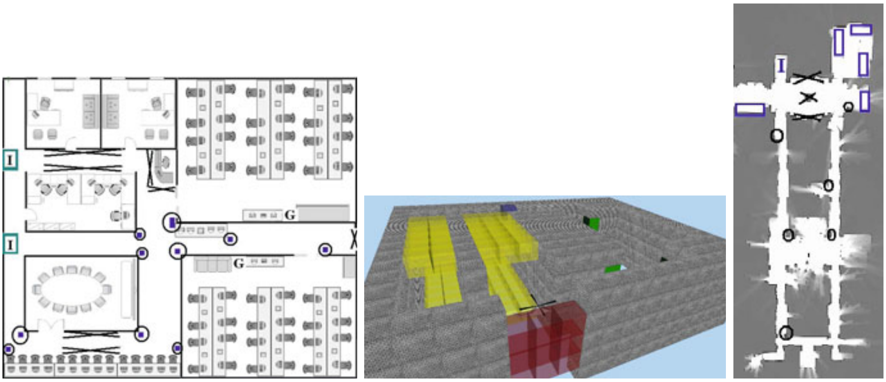
We tested MiGS in several complex realistic robotics scenarios that require long planning horizon. The scenarios are presented in Section 6.1.1. The experimental setup and results are presented in Section 6.1.2 and Section 6.1.3, respectively.

6.1.1 Scenarios

In our experiment, we use the three scenarios below.

(a) ***2D-Navigation.*** In this problem, a 2-DOFs mobile robot navigates in a research lab (Fig 2(a)). The robot's position is represented as a uniform grid of size 60×70. The robot needs to navigate from the entrance of the lab (marked with "I") to one of the goal states (marked with "G"), while avoiding obstacles. The robot never knows its exact position, but it can localize well at some parts of the environment, marked with circles. At each step, the robot performs an action to move to one of its eight adjacent cells. Due to control error, the robot reaches its intended destination 90% of the time. For the rest of the time, the robot either remains in its cell or drift to the left or the right of its intended destination. Moreover, there are danger zones (marked with crosses) that the robot must avoid. Despite the imperfect information about its position and its control uncertainty, the robot needs to move to reach the goal as fast as possible while avoiding both obstacles and danger zones.

(b) ***3D-Navigation.*** In this problem, a 5-DOFs unmanned aerial vehicle (UAV) navigates in a tunnel where GPS signal is not available. The robot's configuration is represented as $(x, y, z, \theta_p, \theta_y)$, where the first three DOFs are the robot's position in a 3D environment, θ_p is the pitch angle, and θ_y is the yaw angle. The configuration space is discretized into grids. The position dimensions are represented as a 3D uniform grid of 5 levels and 18×14 positions at each level (Fig 2(b)). The robot



(a) *(b)* *(c)*

Fig. 2 Experimental scenarios. (a) *2D-Navigation.* (b) *3D-Navigation.* (c) *Target Finding.*

needs to navigate from the entrance of the tunnel (colored red) to a goal position (colored blue). The robot never knows its exact configuration. It can localize by observing the landmarks (colored green) in the environment. However, due to limited sensing ability, it can only observe the landmarks when the robot is at a cell adjacent to the landmark and is heading towards the landmark. At each time step, the robot can either rotate in place or move forward to one of its adjacent cells according to its heading. However, when the robot moves forward, 10% of the time, the robot fails to reach its destination state, instead it drifts to its left or right or remains at the same cell. Moreover, the environment contains danger zones (colored yellow) that the robot must avoid. The main problem in this task is similar to that of the 2D-Navigation scenario, but the state space is much larger. Despite the imperfect information about its position and its control uncertainty, the robot needs to decide which way to go such that it can reach the goal as fast as possible while avoiding both obstacles and danger zones.

(c) ***Target Finding.*** In this problem, a 2-DOFs mobile robot needs to find a moving target in an environment (Fig 2(c), courtesy of the Radish data set). The state space is represented as (r, t), where r is the robot's position and t is the target's position. These positions are represented as a uniform grid of size 49×29. The target may be in one of the places marked with rectangles. It can move within the rectangle, but it can not move to different rectangles. The target motion behavior is entirely unknown. The robot starts from an initial position marked with ”I” and needs to find the target by exploring the possible places of the target. The robot never knows its exact position, but it can localize itself at places marked with circles. Furthermore, due to sensing limitation, the robot will only know the position of its target, and hence accomplish the task, when both the robot and target are in the same grid cell. At each step, the robot performs an action to move to one of its eight adjacent cells. However, due to control error, 15% of the time the robot fails to reach its intended destination, instead it may remain in its cell or drift to the left or to the right of its intended destination. Furthermore, some parts (marked with crosses) of the environment are too dangerous for the robot to pass. Therefore, despite the imperfect information about its position, the target position, and its control uncertainty, the robot needs to decide an exploration strategy that finds the target as fast as possible while avoiding both obstacles and dangerous places in the environment.

6.1.2 Experimental Setup

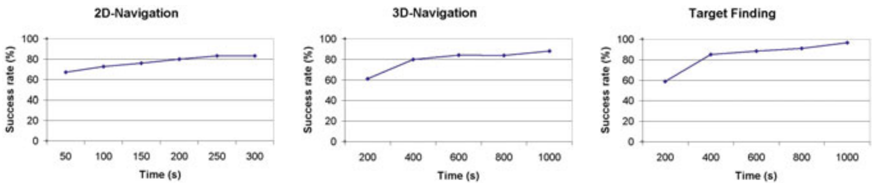
We implemented MiGS in C++ on top of the software package APPL v0.2 [10]. We tested MiGS on the three tasks above and compared the results with PRM [9], a successful motion planner that does not take uncertainty into account, with QMDP [22] which is an approximate POMDP solver well-known in robotics, and with the fastest point-based POMDP solver today, HSVI2 [20]. PRM is implemented in C++. QMDP is implemented on top of the software package APPL v0.2. For HSVI2, we used the newest software released by their original authors, ZMDP v1.1.5. All the experiments were performed on a 2.66GHz Intel processor PC and 2GB memory.

For each task and each method, we performed preliminary runs to determine the suitable parameters, and used the best parameters for generating the results. For MiGS and HSVI2, we used the best parameters to run each method on each scenario for at most 2 hours.

For each task, we compared the success rate, i.e., the percentage that the robot accomplishes the given task successfully within a pre-specified time limit. For MiGS and PRM, we generate 30 different POMDP policies and roadmaps for each task and average the results, as these methods use randomization. For each task, each method, and each policy/roadmap, we ran 100 simulation trial runs to test how well the robot that uses a particular policy/roadmap performs in solving the given task.

6.1.3 Results

The results show that MiGS significantly out-performs other methods for planning with uncertainty, as well as the fastest POMDP solvers today. It is interesting to notice that in 3D-Navigation, PRM that does not take uncertainty into consideration performs better than QMDP. The reason is that the successful runs of PRM are due to luck. On lucky runs, the shortest path from a possible initial state reaches the goal state. However, due to uncertainty, most of the time, the shortest path heuristic moves the robot to a danger zone and prevents it from reaching the goal. QMDP is able to realize that the shortest path strategy has a very high risk, and therefore tries to avoid it. However, QMDP's one lookahead is not sufficient to generate an alternative policy that reaches the goal. By performing farther lookahead, HSVI2 performs better than QMDP. In fact in general, HSVI2 performs much better than QMDP or other POMDP solvers today. However, due to the long planning horizon



MiGS' results.

2D-Navigation	% success
PRM	0.0
QMDP	0.0
HSVI2	0.0
MiGS	**83.3**

3D-Navigation	% success
PRM	14.4
QMDP	0.1
HSVI2	24.0
MiGS	**88.2**

Target finding	% success
PRM	–
QMDP	14.5
HSVI2	17
MiGS	**96.7**

Fig. 3 Experimental results. The time for MiGS includes all initialization and pre-processing needed to construct the roadmap. The results in the table for MiGS is after 300 seconds of runs for 2D-Navigation, and after 1,000 seconds of runs for 3D-Navigation and Target Finding. The results for HSVI2 are the results after 2 hours of runs. We do not apply PRM to the target finding task, as it is unreasonable. The goal (i.e., target) in this problem is moving dynamically, while PRM executes a pre-computed path blindly.

required to generate a good policy in the above problems, HSVI2 is still unable to perform well. The main reason for the poor performance of HSVI2 is that they over-commit to a single heuristic, i.e., the MDP heuristic, to guide its belief-space sampling. This strategy significantly alleviates the difficulty of planning in a high dimensional belief space, but at the cost of significant reduction in belief space exploration ability, which is important for performing well when the heuristic is misleading. MiGS alleviates this problem by using a simplified representation of the state space to guide belief-space sampling. This strategy enables MiGS to explore significantly different parts of the belief space fast.

A video demo of the policy generated for the above scenarios can be seen in http://bigbird.comp.nus.edu.sg/∼hannakur/migs.html. We have also tested MiGS on 2D navigation with uncertain map. The results are similar to the above results and we do not reported it here due to space limitation.

6.2 Robustness of the Generated Plan

In many robotics scenarios, the uncertainty model that one has for planning may not be accurate. Therefore, a desired property of motion planning with uncertainty is to generate motion strategies that are robust enough against slight distortion in the uncertainty model used for planning.

In this set of experiments, we use a simple scenario to test the robustness of the policy generated by MiGS. The scenario involves a 2-DOFs mobile robot navigating in a simple environment (Fig 4), represented as a uniform grid of size 42×20. In this scenario, the robot needs to navigate from the initial position (marked with ”I”) to the goal position (marked with ”G”), while avoiding obstacles and dangerous regions (marked with crosses). Due to limited sensing, the robot can only localize at places marked with circles. There are two landmarks in this environment, and we use the sensing model to encode how well the robot is able to differentiate these two landmarks. At each step, the robot performs an action to move to one of its eight adjacent cells. However, due to control error, the robot will not always reach its intended destination. To model the uncertainty of the system, we need to decide the motion accuracy of the robot and how well its sensing is in differentiating the two landmarks.

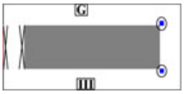
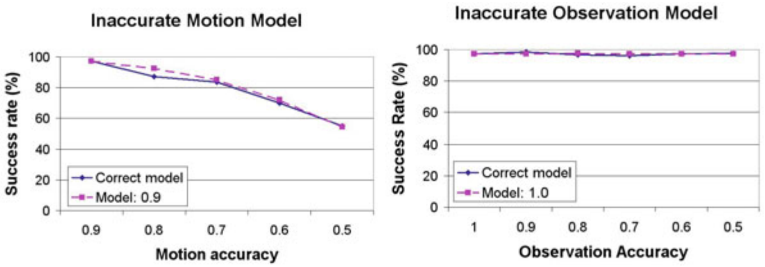


Fig. 4 Simple navigation scenario.

To test the robustness of the policy generated by MiGS, we first generate the policies for the above scenario with a particular motion and sensing model of the robot, and then use the generated policies for robots with different motion and sensing uncertainty to accomplish the same task. Since MiGS uses randomization, we generate 30 policies for a particular robot motion and sensing uncertainty. These policies are generated for robots with 0.9 motion accuracy and perfect sensing ability to differentiate which landmark it sees. We then use each of the 30 policies on robots with



(a) (b)

Fig. 5 The results of generating the policy in 3s. Similar trend holds for the policies generated at different time. The success rate of the "Correct model" is the benchmark. The policy used by the robot is generated based on the correct motion and observation models. (a) "Model: 0.9" means that the policy used by the robot is generated based on motion accuracy 0.9. (b) "Model: 1.0" means that the policy used by the robot is generated based on perfect observation accuracy.

motion accuracy ranging from 0.8 to 0.5, and sensing accuracy ranging from 0.9 to 0.5. The average success rates are shown in Fig 5.

The results show that the policy generated by MiGS based on inaccurate motion or observation model can still be used by the robot to accomplish the given task well. The reason is that knowing a rough idea of a good motion strategy is often sufficient to accomplish the task. For instance, in this environment, as long as the robot knows to stay away from the passages containing dangerous regions, regardless of the exact motion it performs, the robot would reach the goal with high probability. This result corroborates the intuition that many paths in the belief space generate policies with similar quality.

7 Conclusion and Future Work

We have proposed *Milestone Guided Sampling* (*MiGS*), a new point-based POMDP solver that uses a set of sampled states, called milestones, to guide belief-space sampling. MiGS uses the milestones to construct a simplified representation of the state space, and then uses this simplified representation of the state space to guide belief-space sampling. Reasoning using a simplfieid representation of the state space significantly reduces the planning horizons while still capturing most of the useful beliefs. Preliminary experimental results are very promising. They indicate that long planning horizons problems that are impossible to solve using the fastest POMDP solvers today, can be solved by MiGS in just a few minutes.

Two main challenges for enabling POMDP to be practical for realistic robotics problem are the large number of states and the long planning horizon typical of robotics problems. The recently introduced point-based algorithms have shown impressive progress in solving POMDP problems with very large number of states. However, the performance of point-based POMDP degrades significantly when the required planning horizon is long. By alleviating the difficulty of solving problems that require long planning horizon, we hope our work would bring POMDP a step closer to becoming a practical tool for robot motion planning in uncertain and dynamic environment.

Acknowledgements. We thank Sylvie Ong and Shao Wei Png for reading the first draft of this paper and helping with scripting a POMDP model. This work is supported in part by AcRF grant R-252-000-327-112 from the Ministry of Education of Singapore.

References

1. Alterovitz, R., Simeon, T., Goldberg, K.: The stochastic motion roadmap: A sampling framework for planning with markov motion uncertainty. In: Proc. Robotics: Science and Systems (2007)
2. Choset, H., Lynch, K.M., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L.E., Thrun, S.: Principles of Robot Motion: Theory, Algorithms, and Implementations. The MIT Press, Cambridge (2005)
3. Erwig, M.: The graph voronoi diagram with applications. Networks 36(3), 156–163 (2000)
4. Hsiao, K., Kaelbling, L.P., Lozano-Perez, T.: Grasping POMDPs. In: Proc. IEEE International Conference on Robotics & Automation, pp. 4685–4692 (2007)
5. Hsu, D., Latombe, J.C., Kurniawati, H.: On the probabilistic foundations of probabilistic roadmap planning. International Journal of Robotics Research 25(7), 627–643 (2006)
6. Hsu, D., Lee, W.S., Rong, N.: What makes some POMDP problems easy to approximate? In: Proc. Neural Information Processing Systems (2007)
7. Hsu, D., Lee, W.S., Rong, N.: A point-based POMDP planner for target tracking. In: Proc. IEEE International Conference on Robotics & Automation, pp. 2644–2650 (2008)
8. Kaelbling, L., Littman, M., Cassandra, A.: Planning and acting in partially observable stochastic domains. Artificial Intelligence 101, 99–134 (1998)
9. Kavraki, L.E., Švestka, P., Latombe, J.-C., Overmars, M.H.: Probabilistic roadmaps for path planning in high-dimensional configuration space. IEEE Transactions on Robotics & Automation 12(4), 566–580 (1996)
10. Kurniawati, H., Hsu, D., Lee, W.S.: SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In: Proc. Robotics: Science and Systems (2008)
11. Latombe, J.-C.: Robot Motion Planning. Kluwer Academic Publishers, Dordrecht (1991)
12. Pineau, J., Gordon, G.: POMDP planning for Robust Robot Control. In: Proc. International Symposium on Robotics Research (2005)
13. Pineau, J., Gordon, G., Thrun, S.: Point-based value iteration: An anytime algorithm for POMDPs. In: International Joint Conferences on Artificial Intelligence, pp. 1025–1032 (August 2003)
14. Pineau, J., Montemerlo, M., Pollack, M., Roy, N., Thrun, S.: Towards robotic assistants in nursing homes: Challenges and result. Robotics and Autonomous Systems 42(3–4), 271–281 (2003)

15. Prentice, S., Roy, N.: The Belief Roadmap: Efficient Planning in Linear POMDPs by Factoring the Covariance. In: Proc. International Symposium on Robotics Research (2007)
16. Roy, N., Gordon, G., Thrun, S.: Finding approximate POMDP solutions through belief compression. Journal of Artificial Intelligence Research 23, 1–40 (2005)
17. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach, 2nd edn. Prentice-Hall, Englewood Cliffs (2002)
18. Smallwood, R.D., Sondik, E.J.: The optimal control of partially observable Markov processes over a finite horizon. Operations Research 21, 1071–1088 (1973)
19. Smith, T., Simmons, R.: Heuristic search value iteration for POMDPs. In: Proc. Uncertainty in Artificial Intelligence (2004)
20. Smith, T., Simmons, R.: Point-based POMDP algorithms: Improved analysis and implementation. In: Proc. Uncertainty in Artificial Intelligence (July 2005)
21. Spaan, M.T.J., Vlassis, N.: Perseus: Randomized point-based value iteration for POMDPs. Journal of Artificial Intelligence Research 24, 195–220 (2005)
22. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. MIT Press, Cambridge (2005)

Appendix 1. Proof of Theorem 1

Proof. The optimal value function V^* can be approximated arbitrarily closely by a piecewise-linear convex function and represented as $V^*(b) = \max_{\alpha\in\Gamma}(\alpha \cdot b)$ for a suitable set Γ of α-vectors. Let α and α' be the maximizing α-vectors at b and b', respectively. Without loss of generality, assume $V^*(b) \geq V^*(b')$. Thus $V^*(b) - V^*(b') \geq 0$. Since α' is a maximizer at b', we have $\alpha' \cdot b' \geq \alpha \cdot b'$ and $V^*(b) - V^*(b') = \alpha \cdot b - \alpha' \cdot b' \leq \alpha \cdot b - \alpha \cdot b' \leq \alpha \cdot (b - b')$. It then follows that

$$|V^*(b) - V^*(b')| \leq |\alpha \cdot (b - b')|.$$

Next, we calculate the inner product over the partitioned state space:

$$|V^*(b) - V^*(b')| \leq \Big|\sum_{s\in S} \alpha(s)(b(s) - b'(s))\Big| \leq \Big|\sum_{K\in\mathcal{K}}\sum_{s\in K} \alpha(s)(b(s) - b'(s))\Big|$$

Let s_K denote any state in the subset $K \in \mathcal{K}$. We have

$$\begin{aligned} &|V^*(b) - V^*(b')| \\ \leq &\Big|\sum_{K\in\mathcal{K}}\sum_{s\in K}(\alpha(s) - \alpha(s_K) + \alpha(s_K))(b(s) - b'(s))\Big| \\ \leq &\Big|\sum_{K\in\mathcal{K}}\sum_{s\in K}(\alpha(s) - \alpha(s_K))(b(s) - b'(s))\Big| + \Big|\sum_{K\in\mathcal{K}}\sum_{s\in K}\alpha(s_K)(b(s) - b'(s))\Big|. \end{aligned} \qquad (6)$$

Let e_1 and e_2 denote the two terms in (6), respectively. We now bound e_1 and e_2 separately. By the definition of ϵ-partitioning, $|\alpha(s) - \alpha(s_K)| \leq \epsilon$ for all s and s_K in $K \in \mathcal{K}$. Thus,

$$e_1 \leq \sum_{K\in\mathcal{K}}\sum_{s\in K} \epsilon|b(s) - b'(s)| = \epsilon\sum_{s\in S}|b(s) - b'(s)| \leq 2\epsilon, \qquad (7)$$

where the last inequality holds because the L_1 distance between any two beliefs is no greater than 2. Let us now consider e_2. Since the absolute values of α-vector coefficients are no more than $R_{\max}/(1-\gamma)$, it follows that

$$e_2 \leq \sum_{K \in \mathcal{K}} \Big|\alpha(s_K)\Big| \Big|\sum_{s \in K} (b(s) - b'(s))\Big| \leq \sum_{K \in \mathcal{K}} \frac{R_{\max}}{1-\gamma} \Big|\sum_{s \in K} b(s) - b'(s)\Big|.$$

Using the condition $d_{\mathcal{K}}(b, b') \leq \delta$, we get $e_2 \leq \frac{R_{\max}}{1-\gamma}\delta$. Combining this with (6) and (7) gives the desired result.

Scansorial Landing and Perching

Alexis Lussier Desbiens, Alan T. Asbeck, and Mark R. Cutkosky

Abstract. We describe an approach whereby small unmanned aircraft can land and perch on outdoor walls. Our prototype uses an ultrasonic sensor to initiate a pitch-up maneuver as it flies toward a wall. As it begins to stall, it contacts the wall with compliant "feet" equipped with rows of miniature spines that engage asperities on the surface. A nonlinear hierarchical suspension absorbs the kinetic energy and controls contact forces in the normal and tangential directions to keep spines engaged during the landing process. Future work will include powered take-offs and maneuvering in contact with the wall.

1 Introduction

The work described in this paper is a first step toward small flying robots that can land, maneuver and take off from arbitrary surfaces. Such robots combine the best attributes of climbing robots and unmanned air vehicles (UAVs). They can reach remote sites rapidly, flying directly to them. After landing, they can remain fixed or crawl slowly, while consuming little power. The applications include surveillance, environmental monitoring, and inspection of hard-to reach surfaces on buildings, dams, bridges and other large structures. If the robots cling tenaciously, they can also ride out weather that is too rough for flight. When conditions improve, or when their mission is completed, they can jump off and become airborne again.

We are particularly interested in hybrid *scansorial* robots that are adapted for landing and ultimately maneuvering on arbitrary surfaces such as the walls of buildings. As fig. 1 (right) reveals, in the aftermath of an earthquake or other disaster, horizontal surfaces may be too littered with debris and too dangerous for landing, while vertical surfaces are comparatively unobstructed. Moreover, if the flying robots can

Alexis Lussier Desbiens · Alan T. Asbeck · Mark R. Cutkosky
Stanford University, Biomimetic and Dextrous Manipulation Laboratory,
Center for Design Research, 424 Panama Mall, Bldg. 560, Stanford, CA 94305-2232
e-mail: {alexisld,aasbeck,cutkosky}@stanford.edu

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 169–184.
springerlink.com

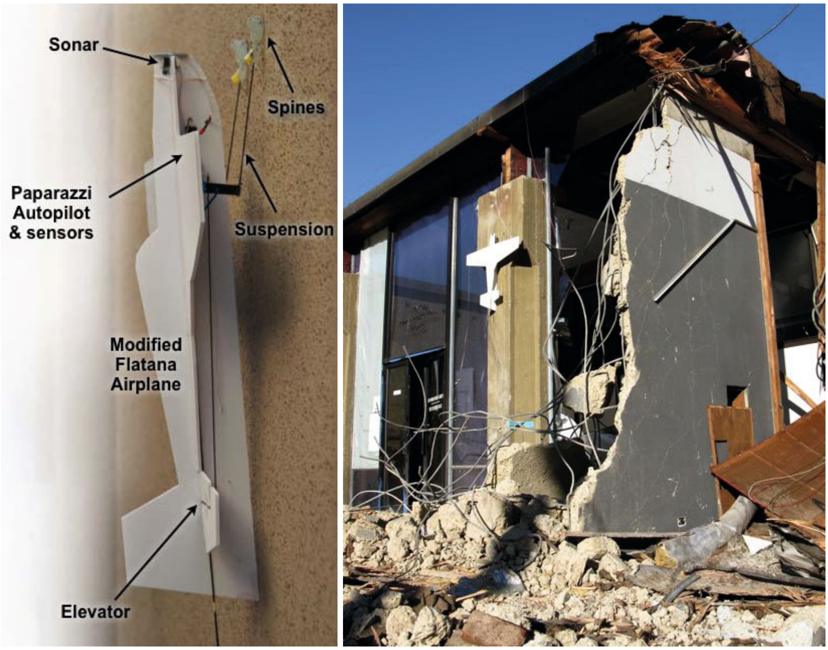


Fig. 1 Left: Our scansorial UAV is based on a modified Flatana (Great Planes, 2009) aerobatic plane. To enable landing on vertical surfaces, several components were added: a controller based on the Paparazzi (Paparazzi, 2008) open-source autopilot, an ultrasonic range sensor, a nonlinear leg suspension and compliant ankles and toes equipped with miniature spines. The fully loaded airplane has a mass of 400g, the landing system accounting for only 28g. Right: Sometimes the best place to land is a wall.

take shelter under the eaves of a building (like swallows or bats) they may perch safely and unobtrusively for long periods of time.

The development of scansorial flying robots draws directly upon two areas of work: unmanned air vehicles capable of acrobatic maneuvers, and vertical climbing robots. In the following section we briefly review the relevant prior work in each of these areas. We then present our approach, which utilizes a small fixed-wing airplane based on the popular Flatana (Great Planes, 2009) platform, as shown in fig. 1 (left). The plane uses feet equipped with miniature spines to grip asperities on the surface. In Section 3 we show that the characteristics of the spines impose constraints on the normal and tangential forces between the plane and the wall. We describe a nonlinear leg suspension in Section 4 that absorbs the kinetic energy of the plane and redirects it toward the feet to satisfy the spine/wall interaction constraints.

The problem of increasing the reliability of scansorial perching has two parts: the first is to design a suspension that will achieve spine engagement and loading for as wide a range of initial contact orientations and velocities as possible; the second is to control the plane so that it reliably approaches the wall within a given envelope of velocities and orientations, despite gusts of wind, etc. In this paper we focus on the former problem using a combination of modeling and experiments with an unpowered glider and a wall with a force plate. We conclude with a discussion of ongoing work to increase the reliability of the approach in outdoor conditions and future work to permit maneuvering on the wall and powered take-off to resume flight.

1.1 Previous Work

Prior related work includes unmanned air vehicles that can perform aerobatic maneuvers such as hovering and alighting on structures. In one approach, researchers (Frank et al, 2007) have used motion capture cameras to control an RC plane for maneuvers such as flying in a room and hovering to land on a docking station. A similar system was used in (Cory and Tedrake, 2008) to create an accurate high-dimensional model of a glider during high angle-of-attack (AOA) maneuvers required during perching. Using this model, they show (Roberts et al, 2009) that the glider becomes less controllable as its airspeed drops just before perching, even if controllability is improved with a fixed propeller or thrust vectoring. Newer work has focused on simplifying the high-dimensional model in a form suitable for the design of feedback controllers (Hoburg and Tedrake, 2009) and on sensors for the detection of electrical wires (Moore and Tedrake, 2009). In other work, autonomous hovering has been demonstrated with fixed-wing aircraft (Green and Oh, 2008). Still other work has focused on performing perching maneuvers using a morphing airplane (Wickenheiser and Garcia, 2008) to maintain controllability.

Recent work at the Air Force Research Laboratory has investigated the aerodynamics and power requirements for a mechanized wing concept (Reich et al, 2009; Lukens et al, 2008) with application to low-speed maneuvers. Another group has recently investigated several creative solutions for perching UAVs (Anderson et al, 2009). Their most successful approach was to fly directly into a wall with a sticky pad located on the nose. After impact, the aircraft is released and hangs from a tether, which can be cut to take off again.

In much of the previous work, an off-board camera system provides accurate trajectory information along with off-board computation for the controller. For our approach, we are interested in landing repeatedly on outdoor surfaces, without access to external vision data or off-board computing. Fortunately, the accuracy requirements for landing on a wall are less severe than those for perching on a wire or pole. In (Lussier Desbiens and Cutkosky, 2009) we describe a simple on-board controller and ultrasonic sensor used with our plane. In this paper we focus on the design of the spines and suspension system to accommodate a relatively wide range of initial conditions at first contact.

The second general area of related work is climbing robots. In particular, the work presented here draws directly upon previous results for robots that climb vertical surfaces such as brick, stucco and concrete using arrays of small spines (Asbeck et al, 2006; Spenko et al, 2008). In the future, it may also be possible to adapt directional dry adhesion (e.g. as used by Kim et al (2008)) for perching aircraft.

Extensive biological research has also been devoted to flying and to ground locomotion. However, much less has focused on the physics of transitions that occur during perching. It has been suggested that flying evolved from the advantages of having even a small amount of lift to control and reduce landing forces (Caple et al, 1983). An example of this phenomenon can be found in the flying squirrel, with its low aspect ratio wing providing aerodynamic stability and lift at angles of attack up to 40 degrees. Furthermore, squirrels deliberately stall themselves prior to landing, allowing them to reduce by 60% their horizontal velocity, while spreading the impact over all four limbs (Byrnes et al, 2008; Paskins et al, 2007). Animals such as geckos also exploit aerodynamics for gliding and landing on vertical surfaces (Jusufi et al, 2008).

2 Dynamic Perching Approach

The basic sequence of landing on a wall is shown in fig. 2. The plane (here an unpowered glider) approaches the wall head-on. When the filtered signal from an onboard ultrasonic sensor indicates that the wall is approximately 5 m away, the plane initiates a pitch-up maneuver to shed speed and align with the wall. Once the plane is nose-up, it starts to fall. The motion of the fixed wing plane after tilt-up is essentially ballistic, with little opportunity for aerodynamic control. As it contacts

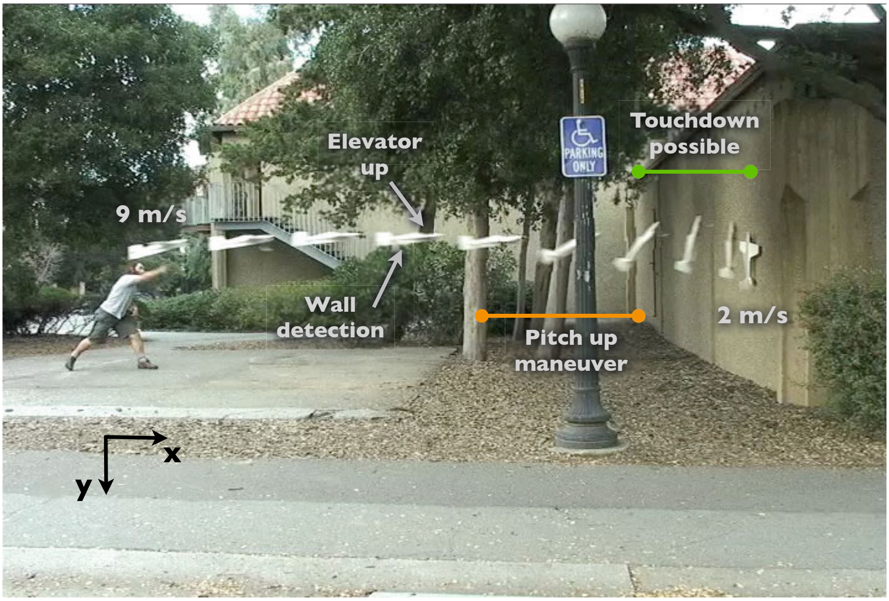


Fig. 2 Landing sequence for dynamic scansorial perching: ultrasonic sensor initiates tilt up at ≈5 m from wall; subsequent motion is largely ballistic. Plane contacts wall at 1-3 m/s. Leg and foot suspensions keep spines engaged while dissipating energy.

the wall, the plane is moving with a total velocity of 1-3 m/s. The plane's "feet" contact the wall and drag briefly until the spines engage. Meanwhile, the leg suspension absorbs the kinetic energy of the plane, preventing it from bouncing off the wall or pitching uncontrollably. As described in the next section, for the spines to engage asperities and remain attached, it is important to maintain the normal/tangential force ratio within a certain range throughout the landing. The complete maneuver, from detection to steady perch on the wall, lasts less than one second.

3 Spines

Spines and spine/surface interactions are described in (Asbeck et al, 2006) with application to climbing robots. The spines used here are similar to those used on the RiSE robot (Spenko et al, 2008), but many fewer are needed because the plane is lighter (400g vs. 3.8kg) and keeps both feet in contact throughout the landing. Each foot is equipped with 5 spines consisting of a small hardened steel hooks ($\approx 10-25\mu$m tip radius) embedded in a molded structure of hard and soft urethane created using Shape Deposition Manufacturing (Cham et al, 2002). The system can be modeled as a damped, elastic linkage as shown in fig. 3.

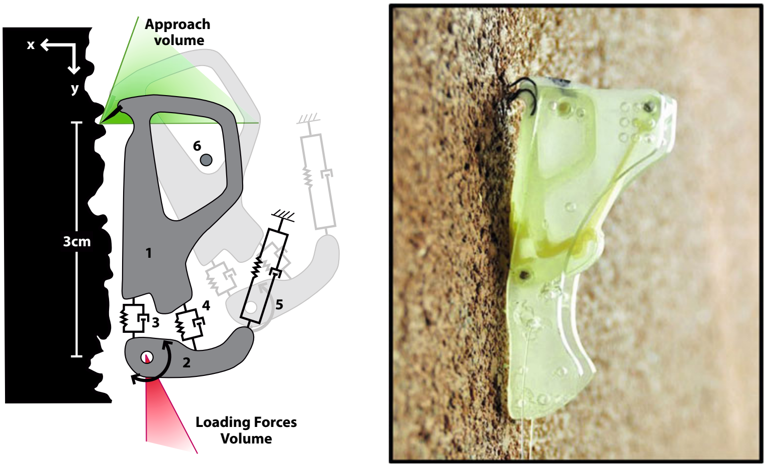


Fig. 3 Left: Model of spine linkage. The spring elements 3 & 4 contribute to the tangential compliance (k=184 N/m for 3 & 4 in parallel, ζ=0.03), while element 5 provides compliance normal to the wall (k=16 N/m). Element 6, a pin going through the entire foot, acts as an overload protection mechanism in conjunction with the hole in element 1. The approach volume is mostly a function of the shape and orientation of asperities; the loading forces volume also depends on the coefficient of friction. Right: Picture of the spines on the aircraft ankle.

To determine the forces required to engage the spines and keep them attached to the surface, an array of 10 spines was tested on a mechanized stage and force plate previously used for directional adhesion tests (Santos et al, 2007). The stage has a positioning accuracy of ±20μm and force measurement accuracy of 25mN. The spines were tested using a sample of roofing tarpaper, chosen because it has a similar roughness to stucco and is easy to cut to size.

The spines require only small forces to engage, up to a maximum of 0.2 N at the maximum preload deflection of 6.5 mm. The maximum preload deflection is determined by an overload mechanism included in the spine design. As shown in fig. 3, a pin passes through the trapezoidal hole in element 1, limiting its travel and preventing over-stretching of the urethane flexures.

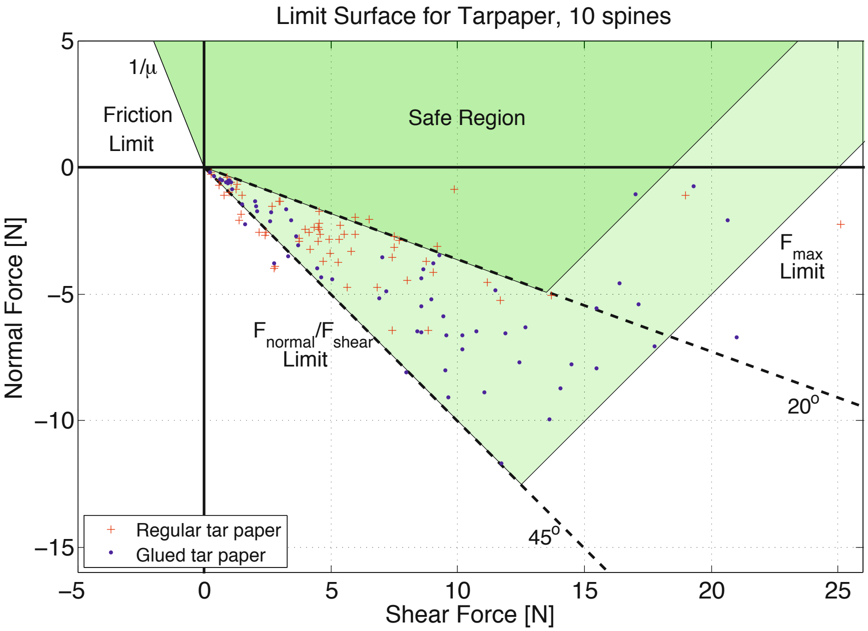


Fig. 4 Limit surface as spines are dragged over unreinforced (red +) and reinforced (blue dots) roofing paper covered with small rock particles. Dark region shows combinations of normal and tangential force that sustain contact without failure; light shaded region shows combinations with occasional failures. When normal force is positive and shear force is negative, Coulomb friction applies.

As the spines drag across the test surface, they can disengage for any of the following reasons:

- they slide off asperities because the loading angle exceeds the F_{normal}/F_{shear} limit for a given spine/asperity contact (Asbeck et al, 2006);
- the asperities dislodge from the surface;
- the spines hit their overload protection pins and detach.

The resulting maximum forces are plotted in fig. 4 as red plus symbols. The normal forces, F_n, are negative, indicating that we are pulling away from the surface ($-x$ direction) and the shear forces, F_s, are positive, indicating that we are pulling in the positive y direction. If we push the spines in the opposite direction, Coulomb friction applies. We also conducted a second set of tests in which the tarpaper surface was strengthened with a thin layer of cyanoacrylate glue to bond the rock particles more firmly in place. The results of these tests, shown as blue dots, reveal the same trend but with higher average forces.

The general picture that emerges from fig. 4 is that there is a region of normal and shear forces for which the spines attach to the surface without failures. The dark green region, bounded by a 20° line, shows regions where very few failures occurred, whereas occasional failures occurred in the lighter green region bounded by a 45° line corresponding to $-F_n = F_s$. The uncertainty in when a failure will occur is due to the probabilistic nature of the spine-surface interaction: asperities are distributed randomly over the surface, and the maximum loading angle that each asperity can sustain is also a random variable, depending on its shape and local friction conditions. With a population of ten spines, the overall forces are typically due to several spine/asperity contacts in slightly different locations. In theory one could observe trials in which all ten spines did not sustain attachment. However, because the spines are dragged a relatively long distance over the surface (16 mm), this is unlikely. Other constraints on the spine forces arise from the overload mechanism, which limits the overall combination of $-F_n$ and F_s (plotted as F_{max} in fig. 4).

Different surfaces will have different safe regions depending on which failure mechanisms dominate. However, the behavior seen in fig. 4 remains. For surfaces such as rough concrete, the limit corresponding to the spine overload-protection mechanism, which permits a maximum force of $\approx$ 2 N per spine with a corresponding elastic deflection of $\approx$ 11 mm, may be the limiting factor. In all cases, the maximum adhesion that can be sustained is a function of the shear force. This relationship has consequences for resisting disturbances such a large gusts of wind and is revisited in section 5. Note also that exceeding the force constraints in fig. 4 is not necessarily catastrophic. If the friction limit for an individual spine/asperity contact is exceeded, or if a particular asperity comes loose, that spine will slip and may reattach, provided that the foot remains in intimate proximity to the wall.

In addition to providing adhesion to the wall, the spines also act as part of the airplane's suspension in series with the leg structure described in section 4. The parallel combination of elements 3 and 4 in fig. 3 has a spring constant of 184 N/m per spine and a damping ratio of ζ=0.03. This low damping ratio causes the plane to rebound upward slightly after the spines are initially loaded. Higher damping in the spine mechanism would be desirable to suppress this effect.

4 Suspension

In contrast to climbing robots, which have control over the force/motion trajectories of their legs, a fixed-wing airplane is essentially ballistic after it has stalled. It is the job of the leg suspension to absorb the kinetic energy of the plane and direct forces to the feet so as to satisfy the constraints depicted in fig. 4. In addition, there are geometric constraints. For example, if the plane and its legs are modeled as a mechanism, only the distal element (i.e., the feet) should contact the wall to avoid unpredictable forces during landing. Ideally, we wish for these force and kinematic constraints to be satisfied for a wide range of touchdown velocities and pitch angles, so that the requirements on the plane's wall sensor and controller can be reduced and the overall system made simpler and more robust.

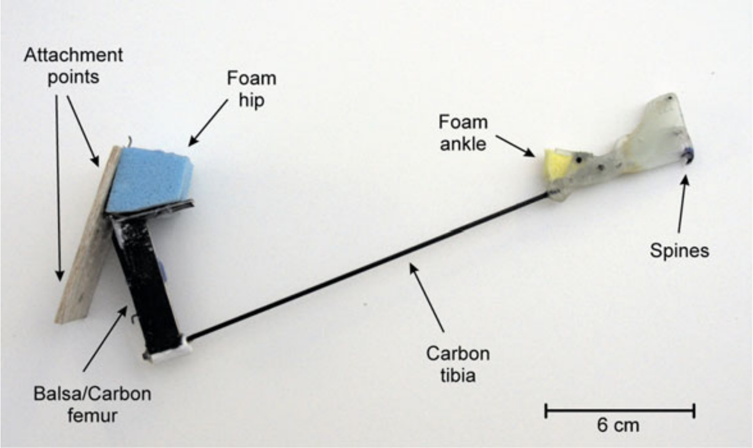


Fig. 5 Compliant leg suspension with stiffness and damping at the hip, knee and ankle.

4.1 Suspension Construction

Before going into the details of the modeling, it is useful to describe the suspension. As seen in fig. 5, the suspension has three articulations: the hip, the knee and the ankle. The hip and the ankle are surrounded by urethane foam, providing stiffness and a high level of damping to these joints. Although the tibia (between the knee and ankle) is a carbon fiber strut, it bends enough that we must consider its stiffness when modeling the system.

The hip joint is designed to have a limited range of motion, achieved by placing the foam in compression, in order to protect the propeller of a powered version of the airplane. This construction results in a nonlinear joint stiffness, as shown in fig. 6, which can be approximated by:

$$k_h = 0.0041 + \frac{0.05}{q_h\,\text{deg} - 100}\ \text{Nm/deg} \tag{1}$$

Damping at the hip is 0.022 Nms/deg for $\theta \geq 130$ deg. and is assumed to scale as $\sqrt{k_h}$ for smaller angles to keep the damping ratio constant.

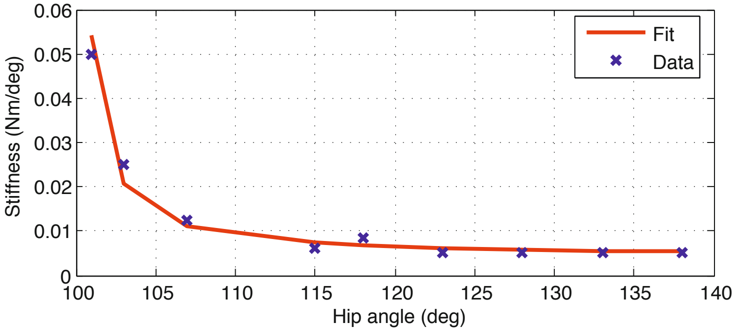


Fig. 6 Hip joint stiffness as a function of the hip angle. The non-linearity prevents excessive compression for high landing forces.

4.2 Planar Landing Model

In order to predict and tune the forces during landing, a simple planar model of the airplane and suspension was created as shown in fig. 7. In this model we ignore roll and yaw motions and lump the two legs together as a single mechanism. The plane is modeled as a rigid body subject to gravity. We ignore aerodynamic forces as we have determined that they do not contribute significantly to the motion of our plane after contact.

We introduce four right-handed reference frames: The wall frame W is defined with the unit vector $\mathbf{w}_x$ oriented toward the wall and $\mathbf{w}_y$ upward along the surface; the airplane frame A is rotated by θ from W around $\mathbf{w}_z$, with its origin at the airplane center of mass; the femur frame F is rotated by -q_H from A with its origin at the hip joint; and the tibia frame T, is rotated by q_K from F with its origin at the knee joint.

Intermittent contact forces, $\mathbf{N}$, with the wall are modeled at the knee and the tail by the use of a spring and damper:

$$\mathbf{N} = \begin{cases} \mathbf{0} & \text{if } x_c < 0 \\ k_g x_c \mathbf{w}_x & \text{if } x_c > 0 \text{ and } \dot{x}_c < 0 \\ (k_g x_c + b_g \dot{x}_c)\mathbf{w}_x & \text{if } x_c > 0 \text{ and } \dot{x}_c > 0 \end{cases} \tag{2}$$

where k_g and b_g are the properties of the ground and $x_c = x_{tail} - x_{wall}$ for the tail point.

Friction at the contact points is modeled using the continuous model from (Mitiguy and Banerjee, 1999):

$$\mathbf{F}_f = -\mu_k\,|\mathbf{N}|\,\frac{\mathbf{v}}{|\mathbf{v}| + \varepsilon_v} \tag{3}$$

where μ_k is the coefficient of kinetic friction, $|\mathbf{N}|$ is the magnitude of the normal force, $\mathbf{v}$ is the velocity of the point in contact and ε_v is a small positive number.

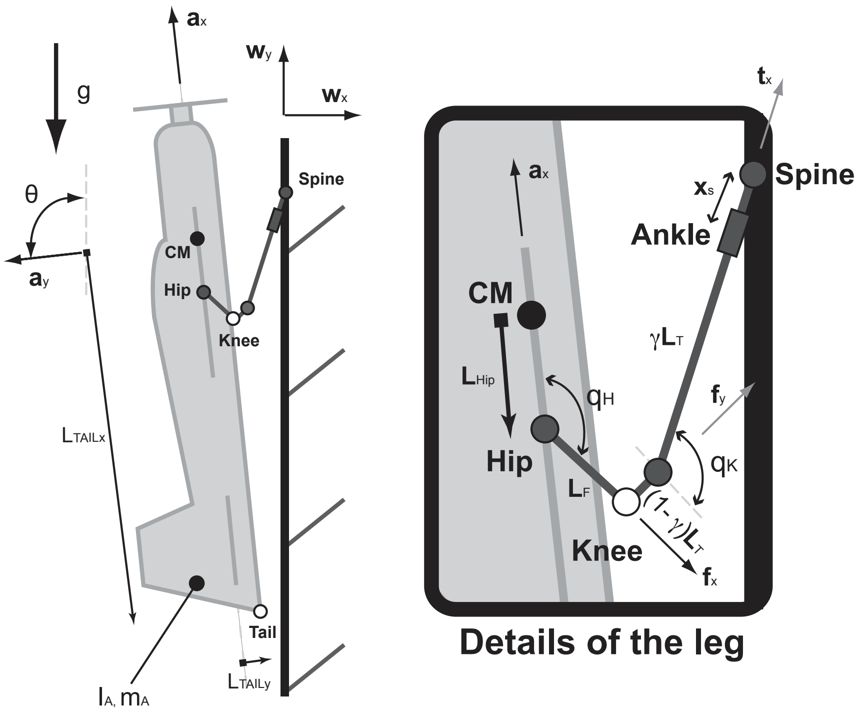


Fig. 7 Planar model: The airplane is a rigid body with possible contacts at the foot, knee and tail. The suspension is a massless linkage with a spring and damper at the hip, knee and spine joints. The knee uses a pseudo-rigid-body approximation (Howell, 2001) to account for large elastic deflection in the tibia.

Because of its light weight compared to the airplane, the suspension is modeled as three massless links, ignoring the ankle joint because of its small motion in comparison to the femur, tibia and spines. The hip joint is modeled with a non-linear spring and damper as defined in section 4.1, while the spine suspensions are modeled as a prismatic joint with a linear spring as described in section 3.

The large deflection of the tibia can be approximated with a pseudo-rigid-body model of a cantilever beam with a force at the free end (Howell, 2001). This approximation places a pseudo joint at a distance γL from the free end, where L is the length of the tibia and $\gamma \approx 0.85$. The stiffness of the virtual joint can then be expressed as $k_k \approx \pi\gamma^2 EI/L = 0.062$ Nm/deg.

Upon contact of the foot with the wall, the linkage of the suspension defines a constraint on the motion of the center of mass of the airplane given by:

$$x\,\mathbf{w}_x + y\,\mathbf{w}_y = L_{hip}\,\mathbf{a}_x - L_f\,\mathbf{f}_x - (1-\gamma)L_t\,\mathbf{f}_y - (\gamma L_t + x_s)\,\mathbf{t}_x + x_{foot}\,\mathbf{w}_x + y_{foot}\,\mathbf{w}_y \quad (4)$$

where $\mathbf{w}_x$, $\mathbf{w}_y$, $\mathbf{a}_x$, $\mathbf{f}_x$, $\mathbf{f}_y$ and $\mathbf{t}_y$ are unit vectors as defined in fig. 7.

Taking the dot product of this equation with $\mathbf{w}_x$ and $\mathbf{w}_y$, we obtain two scalar constraint equations that can be differentiated to establish two velocity constraint relations among the joint angle velocities, $\dot{q}_h$, $\dot{q}_k$ and $\dot{x}_x$ and the airplane center of mass velocity and pitch rate:

$$\begin{bmatrix} \dot{x}_{foot} \\ \dot{y}_{foot} \end{bmatrix} = J \begin{bmatrix} \dot{q}_h \\ \dot{q}_k \\ \dot{x}_s \end{bmatrix} + f(\dot{x}, \dot{y}, \dot{\theta}) \tag{5}$$

where the $\dot{x}_{foot}$ and $\dot{y}_{foot}$ are zero because we assumed a non-sliding contact with the wall. As the linkage is redundant, the preceding equation is not sufficient to solve for the joint velocities given the airplane velocities. However, static equilibrium must be maintained, as the linkage is massless, and this condition can be formulated as:

$$(I - J^T J^{T^{\#}})\tau = 0 \tag{6}$$

where $J^{T^{\#}}$ is the pseudo-inverse of J^T, and τ is a matrix of joint torques expressed as $-K(q - q_0) - C\dot{q}$. In the case of the linkage presented here, equation 6 gives us three redundant equations, from which one of them can be used in conjunction with equation 5 to solve for the joint velocities.

From the joints positions and velocities, it is possible to compute the torques at each joint. Then, as the joint motions have been computed while being subject to the constraints of the Jacobian matrix, it is possible to discard one column of the Jacobian matrix to obtain a reduced version (J_r^T) that can be inverted to find the tip forces at the end of the foot:

$$\begin{bmatrix} F_{foot_x} \\ F_{foot_y} \end{bmatrix} = J_r^{T^{-1}} \begin{bmatrix} \tau_{hip} \\ \tau_{knee} \end{bmatrix} \tag{7}$$

With gravity, tail contact and foot forces established, the equations of motion are generated with Autolev$^{\text{TM}}$, an analytical motion analysis software program, which also generates the Matlab$^{\text{TM}}$ code necessary to solve them.

4.3 Selection of Joint Parameters

With the model established in the previous section, and for a given suspension geometry and landing state, it is possible to determine the joint parameters that will let the plane land successfully. The success criteria are taken from section 3 for tarpaper, requiring that $-F_n/F_s < 1$, with a maximum total force of 25 N and a landing in which the suspension does not bottom out, so that only the foot contacts the wall.

The construction of the knee provides only a few discrete options for adjusting the overall stiffness and damping. Therefore, most of the tuning is done at the hip. Fixing the knee stiffness at 0.062 Nm/deg and ignoring the knee damping, which has only a small effect in comparison to the hip, a series of landings were simulated for a typical set of initial conditions: v_x = 1.2 m/s, v_y = -0.5 m/s and θ = 97 deg. As

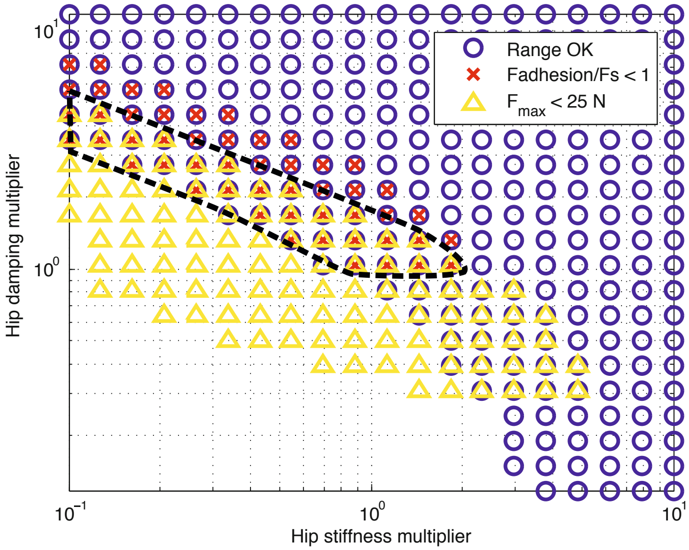


Fig. 8 Successful landings for variations in hip stiffness and damping with respect to the nominal values given by eq. 1. Only a narrow band of combinations satisfies simultaneous constraints on the $-F_n/F_s$ ratio, the maximum force, F_{max}, and acceptable ranges of joint motion for typical landing velocities, v_x = 1.2 m/s and v_y = -0.5 m/s.

seen in fig. 8 there is a relatively narrow band of possible hip stiffness and damping values around the nominal values given by equation 1 and scaling the damping as described in section 4.1.

4.4 Experimental Validation

To validate our model, the plane was landed on an instrumented wall. The instrumented wall consists of an ATI-Gamma SI-32-2.5 force sensor installed behind a 30x30cm piece of wall material. The force plate has a natural frequency of 100 Hz in the shear direction and 130 Hz in the normal direction. Forces were filtered using a 75Hz Butterworth filter for both the data and the simulation. The touchdown velocity and pitch angle were obtained by analyzing a 30 fps video of the landing.

Figures 9 and 10 compare typical results between the data collected on the force plate and the one generated with our simulator. To match the measured data, the joint stiffness and damping in the simulation were tuned using a genetic algorithm and resulted in a best-fit joint stiffness approximately 60% higher than what was measured during static tests. The values for the ground parameters in the simulation were roughly guessed, as they have a limited influence on the system response which is dominated by the soft suspension.

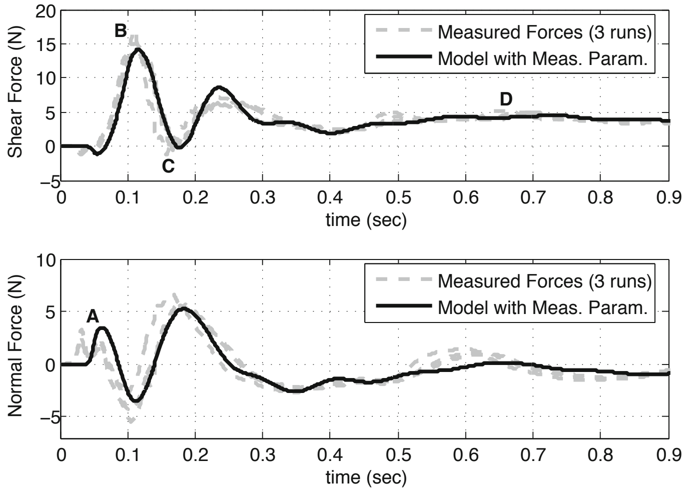


Fig. 9 Comparison between measured and simulated forces. Point **A** represents the touchdown (one impact for each foot in the real data). The shear force then peaks at point **B** when the tail touches the wall, and reaches equilibrium at point **D**. The dip in shear forces at point **C** represents the springback force due to the spine suspension.

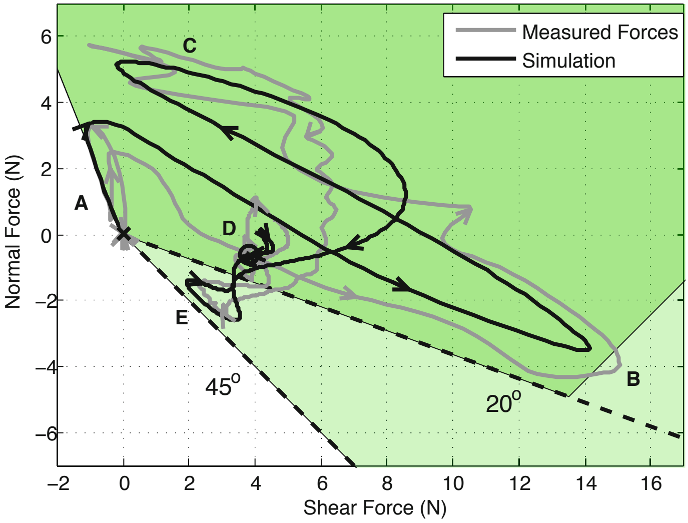


Fig. 10 Comparison of typical measured and simulated force trajectories plotted with respect to the safe regions given by fig. 4. The forces initially pass through point **A** as the tail of the plane touches the wall. The shear force then increases to **B**. The spines are then unloaded by their suspensions at **C** and reach a steady-state value at **D**. The critical point is at **E**, the point of maximum rebound.

Figure 10 shows more specifically that the forces remain, during the landing, in the safe 45° region for reinforced tarpaper as defined in fig. 4. The critical state is at point **E**, as the plane rebounds immediately after the initial impact. Increasing damping would reduce the rebound and increase the safety margin, progressing from **C** to point **D**. The main difference between the simulator and the measured forces, as shown in fig. 9 at point **A**, is that we can observe on the real data the impact of each foot while our model lumps both feet together.

The resulting spine and suspension system allows the plane to perch on a rough vertical wall for a relatively large envelope of touchdown velocity and pitch angle. This envelope, computed with the simulator ($-F_n/F_s < 1$, $F_{max} < 25N$ and the knee joint not touching the wall), is illustrated in figure 11. As one can see on this figure, landing is possible not only for a zero-velocity touchdown, but for forward velocities of up to $v_x = 2.7$ m/s, pitch angles between 65 and 110 degrees and downward velocities of up to 1 m/s.

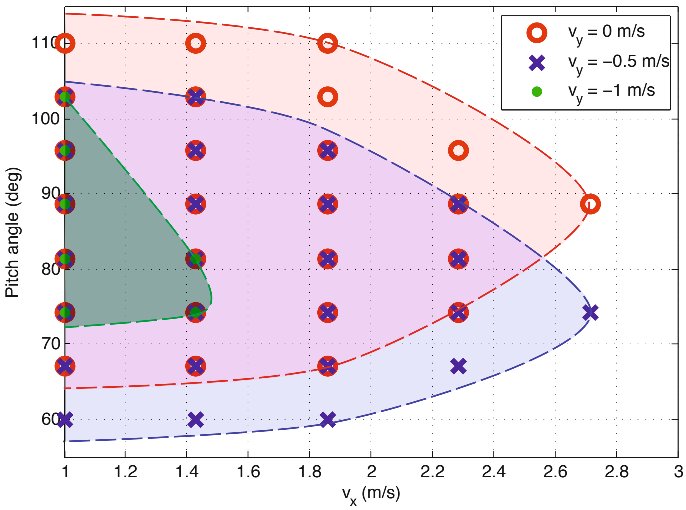


Fig. 11 Successful landings simulated for various touchdown states.

About 30 successful landings have been performed so far, out of 40 attempts, and touchdown states all over and around the limits of the computed envelope have been observed. This large envelope has numerous benefits for a small UAV system: it reduces the requirements on the control (less need for accurate control during a short maneuver), it reduces the need for sensing (fewer, cheaper and smaller sensors), it increases the robustness of the system and it allows smooth landing at a moderate forward speed.

5 Conclusion

We have demonstrated a system in which a combination of miniature spines and a nonlinear suspension allow a small unmanned airplane to approach and land on a vertical wall. The focus of the work reported here is on the suspension design, needed to dissipate the kinetic energy of the plane while maintaining a desirable ratio of normal and tangential contact forces for spine engagement. Simulations and experiments reveal that the system accommodates a range of typical initial conditions. In practice, we have observed 30/40 successful trials in calm outdoor conditions. The models also reveal room for improvement. In particular, the damping at the proximal and especially the distal elements can be increased for a smoother progression from initial contact to steady state.

A number of immediate extensions are also warranted. The model should be enhanced to account for the dynamic effects resulting from the (stochastic) spine/wall interactions. A fully three dimensional model, with two separate legs, would also help us to explore the sensitivity to roll and yaw variations. From a practical standpoint, the short term goal is to achieve powered takeoff and resumption of flight. The ideal way to accomplish this takeoff would be to exploit stored elastic energy in the legs so as to reach controllable airspeed rapidly. In parallel, the control of the plane should be improved, with more reliable wall sensing and better control of the velocity and orientation prior to landing, particularly in the presence of wind.

Looking a bit further ahead, we recognize that we will need to deploy opposed sets of spines to resist gusts of wind after attachment. We have built opposed-spine prototypes that sustain normal forces of up to 20 N in benchtop tests and will be adapting these to the airplane. Still further ahead are developments to perform crawling maneuvers while in contact with wall, using a combination of thrust vectoring and leg motions. This will result in a true scansorial hybrid with flying and crawling behavior.

Acknowledgements. Alexis Lussier Desbiens is supported by the Natural Sciences and Engineering Research Council of Canada and the Organization of American States, with additional support from DARPA DSO. We would also like to thank the members of BDML at Stanford for all their help in conducting the experiments reported here.

References

Anderson, M., Perry, C., Hua, B., Olsen, D., Parcus, J., Pederson, K., Jensen, D.: The Sticky-Pad Plane and other Innovative Concepts for Perching UAVs. In: 47 th AIAA Aerospace Sciences Meeting (2009)

Asbeck, A., Kim, S., Cutkosky, M., Provancher, W.R., Lanzetta, M.: Scaling hard vertical surfaces with compliant microspine arrays. International Journal of Robotics Research 25(12), 14 (2006)

Byrnes, G., Lim, N.T.L., Spence, A.J.: Take-off and landing kinetics of a free-ranging gliding mammal, the malayan colugo (galeopterus variegatus). Proceedings of the Royal Society B: Biological Sciences 275(1638), 1007–1013 (2008)

Caple, G., Balda, R.P., Willis, W.R.: The physics of leaping animals and the evolution of preflight. The American Naturalist 121, 455–467 (1983)

Cham, J.G., Bailey, S.A., Clark, J.E., Full, R.J., Cutkosky, M.R.: Fast and robust: Hexapedal robots via shape deposition manufacturing. IJRR 21(10), 869–882 (2002)

Cory, R., Tedrake, R.: Experiments in fixed-wing uav perching. In: Proceedings of the AIAA Guidance, Navigation, and Control Conference (2008)

Frank, A., McGrew, J.S., Valenti, M., Levine, D., How, J.P.: Hover, transition, and level flight control design for a single-propeller indoor airplane. In: AIAA Guidance, Navigation and Control Conference (2007)

Great Planes, Electrifly flatana (2009), http://www.electrifly.com/flatouts/gpma1111.html

Green, W., Oh, P.: Autonomous hovering of a fixed-wing micro air vehicle. IEEE International Conference of Robotics and Automation (2008)

Hoburg, W., Tedrake, R.: System identification of post stall aerodynamics for uav perching. In: Proceedings of the AIAA Infotech@Aerospace Conference (2009)

Howell, L.: Compliant mechanisms. Wiley-Interscience, Hoboken (2001)

Jusufi, A., Goldman, D.I., Revzen, S., Full, R.J.: Active tails enhance arboreal acrobatics in geckos. Proceedings of the National Academy of Sciences 105(11), 4215–4219 (2008)

Kim, S., Spenko, M., Trujillo, S., Heyneman, B., Santos, D., Cutkosky, M.R.: Smooth vertical surface climbing with directional adhesion. IEEE Transactions on Robotics 24(1), 65–74 (2008)

Lukens, J., Reich, G., Sanders, B.: Wing Mechanization Design and Analysis for a Perching Micro Air Vehicle. In: 49th Structures, Structural Dynamics, and Materials Conference (2008)

Lussier Desbiens, A., Cutkosky, M.: Landing and Perching on Vertical Surfaces with Microspines for Small Unmanned Air Vehicles. In: 2nd International Symposium on Unmanned Aerial Vehicles (2009)

Mitiguy, P.C., Banerjee, A.K.: Efficient simulation of motions involving coulomb friction. Journal of Guidance, Control and Dynamics 22(1) (1999)

Moore, J., Tedrake, R.: Powerline perching with a fixed-wing uav. In: Proceedings of the AIAA Infotech@Aerospace Conference (2009)

Paparazzi, Paparazzi, the free autopilot (2008), http://paparazzi.enac.fr

Paskins, K.E., Bowyer, A., Megill, W.M., Scheibe, J.S.: Take-off and landing forces and the evolution of controlled gliding in northern flying squirrels glaucomys sabrinus. Journal of Experimental Biology 210(8), 1413–1423 (2007)

Reich, G., Wojnar, M., Albertani, R.: Aerodynamic Performace of a Notional Perching MAV Design. In: 47 th AIAA Aerospace Sciences Meeting (2009)

Santos, D., Kim, S., Spenko, M., Parness, A., Cutkosky, M.: Directional adhesive structures for controlled climbing on smooth vertical surfaces. In: IEEE ICRA, Rome, Italy (2007)

Roberts, J., Cory, R., Tedrake, R.: On the controllability of fixed-wing perching. In: American Controls Conference (2009)

Spenko, M., Haynes, G., Saunders, J., Cutkosky, M., Rizzi, A., Full, R.: Biologically inspired climbing with a hexapedal robot. Journal of Field Robotics (2008)

Wickenheiser, A.M., Garcia, E.: Optimization of perching maneuvers through vehicle morphing. Journal of Guidance 31(4), 815–823 (2008)

Anthropomorphic Soft Robotics – From Torque Control to Variable Intrinsic Compliance

Alin Albu-Schäffer, Oliver Eiberger, Matthias Fuchs, Markus Grebenstein,
Sami Haddadin, Christian Ott, Andreas Stemmer, Thomas Wimböck,
Sebastian Wolf, Christoph Borst, and Gerd Hirzinger

Abstract. The paper gives an overview on the developments at the German Aerospace Center DLR towards anthropomorphic robots which not only try to approach the force and velocity performance of humans, but also have similar safety and robustness features based on a compliant behaviour. We achieve this compliance either by joint torque sensing and impedance control, or, in our newest systems, by compliant mechanisms (so called VIA - variable impedance actuators), whose intrinsic compliance can be adjusted by an additional actuator. Both approaches required highly integrated mechatronic design and advanced, nonlinear control and planning strategies, which are presented in this paper.

1 Introduction

Soft Robotics is an approach for designing and controlling robots which can interact with unknown environments and cooperate in a safe manner with humans, while approaching their performance in terms of weight, force, and velocity. These robots are expected to push forward not only such new application fields as medical robotics, robotized outer space and planetary exploration, or personal service and companion robotics, but also to drastically move the horizons of industrial automation. Today's industrial robots still operate in their huge majority in blind, position controlled mode, being dangerous to humans and thus having to be enclosed by protective fences. In contrast, this new generation of robots can share the space and the workload with the humans adapting better to product diversity and to short production life cycles. However, it is clear that these human friendly robots will look

A. Albu-Schäffer · O. Eiberger · M. Fuchs · M. Grebenstein · S. Haddadin · Ch. Ott ·
A. Stemmer · T. Wimböck · S. Wolf · Ch. Borst · G. Hirzinger
Institute of Robotics and Mechatronics
German Aerospace Center (DLR)
e-mail: alin.albu-schaeffer@dlr.de

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 185–207.
springerlink.com

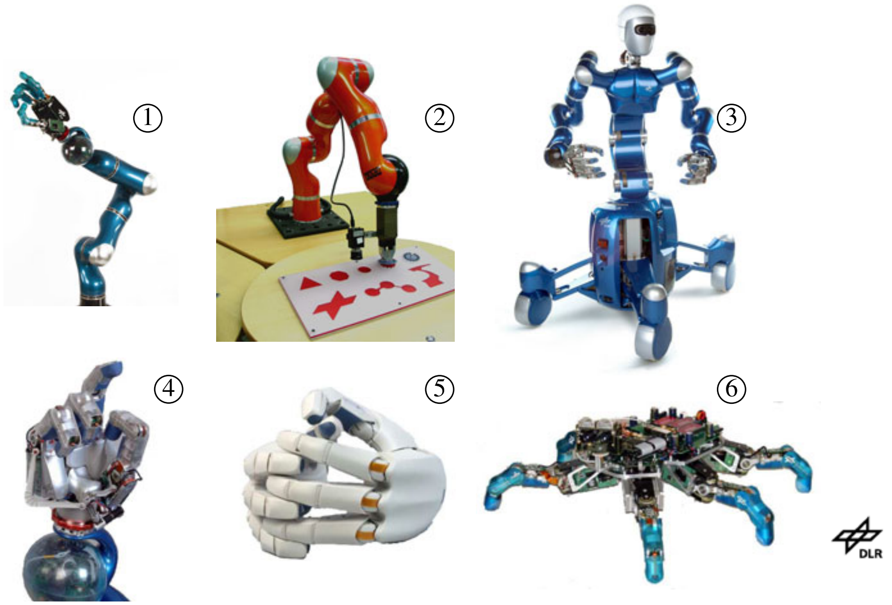


Fig. 1 Overview of the DLR Robots: ①: The DLR-LWRIII equipped with the DLR-HandII. ②: The DLR-KUKA-LWRIII which is based on the DLR-LWRIII. ③: The DLR Humanoid "Rollin' Justin". ④: The DLR-HandII-b, a redesign of the DLR-HandII. ⑤: The Schunk Hand, a commercialized version of the DLR-HandII. ⑥: The DLR-Crawler, a walking robot based on the fingers of the DLR-HandII.

very different from today's industrial robots. Rich sensory information, light-weight design and soft robotics features are required in order to reach the expected performance and safety. In this paper we will address the two approaches followed at DLR for reaching the aforementioned new design goals. The first one is the meanwhile mature technology of torque controlled light-weight robots (see Fig. 1) . Several products resulted from this research and are currently being commercialized through cooperations with various industrial partners (DLR-KUKA Light-Weight Robot LWRIII, DLR-HIT-Schunk Hand, DLR-Brainlab-KUKA medical robot). The second technology, currently a topic of very active research in the robotics community, is variable compliance actuation. It aims at enhancing the soft robotics features by a paradigm change from impedance controlled systems to variable mechanical stiffness and energy storage, in close interplay with innovative control strategies, as suggested by the human motor system.

Regarding the actively compliant controlled systems, we will concentrate on the newest developments in the design and control leading to the humanoid system Rollin' Justin as well as on the steps required to make the technology widely usable in industrial environments. We are considering these robots as a performance reference, which we are currently trying to outperform with new variable stiffness actuators. We will present the main design ideas and some experimental examples providing first validation of the performance and robustness gain of this design approach.

2 Light-Weight, Modular, Torque Controlled Robots

For almost one decade we focused at DLR on the development of torque controlled, light-weight arms and hands. We refined the technology in successive steps in order to obtain high power actuators, a light-weight though robust design, highly integrated, reliable electronics, and torque sensors with low hysteresis, noise, and drift. Moreover, we developed control algorithms which allow both high performance trajectory tracking and safe and efficient compliant interaction with humans and unknown environments. With the LWRIII and the DLR-Hand IIb a state of maturity and performance of the systems was finally reached, which allowed the commercialization of the two systems in cooperation with industrial partners. The arm is manufactured and distributed by the industrial robot manufacturer KUKA Roboter GmbH, while a simplified version of the hands, designed in cooperation with the Harbin Institute of Technology (China) is distributed by the robot gripper manufacturer Schunk GmbH. Moreover, several spin-off companies emerged from these projects, producing components such as torque sensors and high torque motors.

In the last years we started additionally a wide new area of research activities based on this technology by taking advantage of the modular and integrated structure of the components. A fully new line of medical robots was developed, based on both the hand and arm components. The humanoid manipulation system Justin was build up from these components as well, while the modularity of the hands allowed the design of a new crawler robot in only a few months.

In our previous work [1, 2, 3, 4], we presented in detail the design and the control concepts of the LWR-III arm and HandIIb system. In this paper we focus on the evolution of the design and control approaches required for the development of Justin, as well as on the components required for a successful application of the arms in a production assisting environment.

2.1 Interaction Control of DLR Robotic Systems

The control of both the arms and hands makes extensive use of the torque sensing available in each joints. The sensors are placed after the gear-box and allow therefore a very precise measurement of the real joint torque, in contrast to simple current based torque estimations. They are, in the given accuracy and sampling rate, a unique feature of the DLR robots, finally turning into reality the old dream of the robotics control community of having robots with torque interface [5, 6]. The sensors are used to implement both active vibration damping for high performance motion control as well as soft robotics features such as impedance control, collision and failure detection, potential field based collision avoidance and posture control. Due to the the relatively high intrinsic compliance of the harmonic drive gears and of the torque sensors, the classical rigid robot assumption is not acceptable for the DLR arms, if high control performance is sought for. Therefore, a major research contribution was to extend many of the known approaches from classical robot control to the flexible joint case by taking advantage of the joint torque measurement. In the flexible joint model, not only the motor position θ, but also the joint torque τ, as well as their derivatives $\dot{\theta}$ and $\dot{\tau}$ are namely states of the system. The measurement of the former and the numerical computation of the latter provides the state estimation required for full state feedback. For the light-weight arm and hands, these methods were presented, e.g., in [1, 2, 3, 4].

The control framework (for both position and impedance control) is constructed from the perspective of passivity theory (Fig. 2) by giving a simple and intuitive physical interpretation in terms of energy shaping to the feedback of the different state vector components.

- A physical interpretation of the joint torque feedback loop is given as the shaping of the motor inertia *B*.
- The feedback of the motor position can be regarded as shaping of the potential energy.

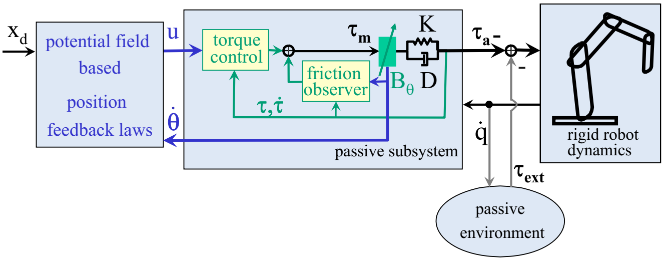


Fig. 2 Representation of passive control structures.

The robustness and performance of the control methods has been extended to product maturity for the commercialization of the light-weight arm in cooperation with KUKA Roboter GmbH and of the Kinemedic/MIRO arms with BrainLab AG. Moreover, during performance tests at the industrial robot manufacturer it turned out that despite of the light-weight, elastic structure, the robot has competitive motion accuracy to an industrial robot of similar payload, according to ISO9283-1998 standard measurements.

2.1.1 Disturbance Observers

Since the control of the DLR robots is fundamentally relying on accurate models of the robot dynamics, friction torques in the gear-box and external interaction torques (from humans or the environment) are a critical source of errors which have to be estimated correctly. Therefore, a new disturbance observer concept was developed [7, 8]. It allows the independent estimation of friction and external collision torques using the same observer structure by exploiting the joint torques signals τ (see Fig. 3). The friction observer allows high performance motion control as mentioned in the previous section, while the external torque observer is used for safe human-robot interaction, described in Section 2.5. Moreover, although it has an active integrator action, the friction observer can be analyzed within the passivity framework, thus allowing convergence statements for the entire nonlinear system [9].

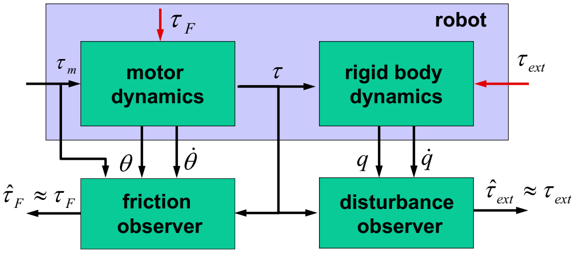


Fig. 3 Disturbance observers for identification of the friction and external interaction torques τ_F and $\tau_{ext}.\tau_m$ and τ are the motor and the measured torques, respectively.

2.2 *Design and Control of the Humanoid Manipulation System Justin*

Justin was designed as a versatile platform for research on two-handed manipulation and service robotics in everyday human environments. Due to the modular design of the LWRIII as well as of Hand-IIb, it was possible to quickly set up both a left-handed and right-handed configuration. The robots' common base holds the arms mounted 60 degrees from the vertical in a sideways direction. This allows the elbow to travel fore and aft below the shoulder and up to horizontal height without passing through singularities. To extend the manipulation range, the robot base is held by a four degrees of freedom (DOF) torso. A vertical roll axis, followed by two pitch joints and a third, passive pitch axis which keeps the arm base upright, allows translations in a vertical plane which can rotate about a vertical axis. Through this configuration Justin is capable of lifting objects from the floor, reaching over tables and even reaching objects on a shelf of about two meters height (Fig. 4).

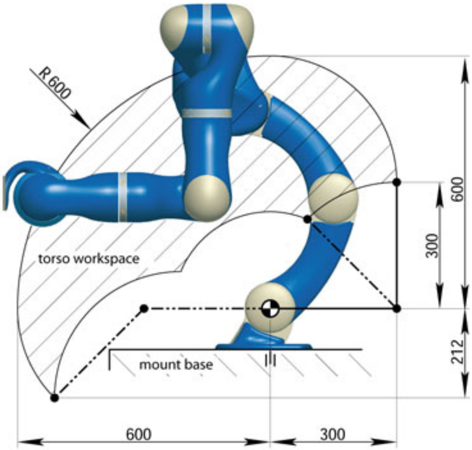


Fig. 4 Workspace design for the humanoid Justin.

To maintain the DLR robot concept, the torso joints consist of the same functional components as the arm joints, allowing full torque control for the setup. In this way, Justin can detect and react to contact forces applied anywhere on its structure.

2.2.1 Justin's Mobile Plattform

Justin's new mobile platform enables the system to interact with humans in a larger workspace and thus brings the development towards a universal service robotics platform [10]. The robot base requires a large supporting polygon in order to take advantage of the large workspace, the high forces, and the dynamics of the upper body, while providing the stability of the overall system. On the other hand, compact dimensions are necessary for a reliable and easy navigation through doors or narrow passages. To meet both requirements, our mobile platform has four legs which can be individually extended via parallelogram mechanisms (Fig. 5), even during platform movement. Each leg carries a steerable wheel for omnidirectional (but nonholonomic) movement. This novel kinematics needs new control and planning algorithms [11], since the wheel system has no longer an instantaneous center of rotation while extending or retracting the legs. Furthermore, each leg incorporates a lockable spring damper system. This enables the whole system to drive over tiny obstacles or to cope with the unevenness of the floor, as well as to sustain reaction forces under heavy load. The mobile platform has a weight of 150kg. Mounted on the mobile platform, Rollin Justin has a shoulder height of up to 1.6m. The whole system is powered by a Lithium-Polymer battery pack and has an operating time of about 3h. For enabling the implementation and evaluation of advanced control algorithms, the whole upper body is controlled in 1ms cycle, while the platform is connected at rate of 16 ms.

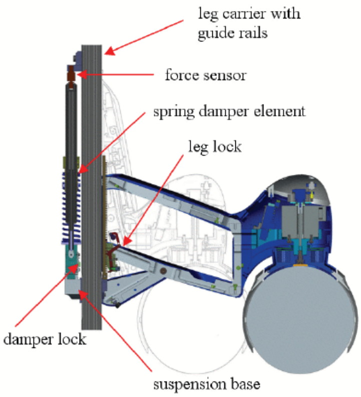


Fig. 5 Variable footprint of Rollin Justins mobile base.

2.2.2 Interaction Control of Justin

All the interaction control methods developed for the arms and the hands were extended and transferred to Justin in the last three years. The Cartesian impedance controller concept was extended [12, 13] to the upper body including hands, arms and the torso (Fig. 6).

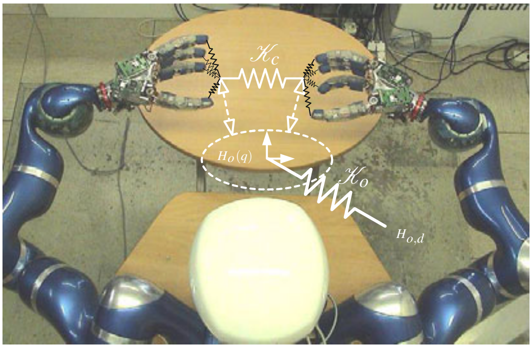


Fig. 6 Two-hand impedance behavior by combining object level impedances of the hands and the arms.

Since the mobile platform has only a velocity interface, but no torque interface, a full body compliance control requires to follow an admittance control approach for the base, as sketched in Fig. 7(Right). Therefore, the virtual wrench resulting on the base of the torso from the impedance controller of the upper body is transformed using a virtual spring and damper into a velocity command.

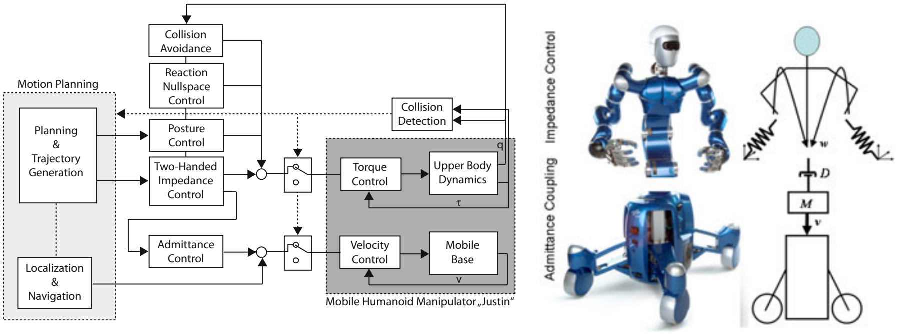


Fig. 7 Left: Torque based Control Structure for Justin. Right: The upper body is impedance controlled in a 1ms cycle. The base is admittance controlled and its desired velocity is related to the virtual force produced by the controller in the base of the torso by a virtual mass-damper dynamics.

In Fig. 7, left, an overview of the entire impedance based control system is shown. A task and trajectory planning stage provides the desired task space motion to the Cartesian impedance controller and a desired posture for the nullspace control. In order to minimize the dynamic reaction forces on the mobile base, a reaction nullspace control approach is integrated into the system [14]. For achieving safety for humans in the workspace of the robot, the system contains two complementary approaches. Firstly, a potential function based collision avoidance is used [15]. Secondly, a disturbance observer based collision detection routine allows to implement different collision reaction strategies (Sec. 2.5, [8]).

2.3 Technology Transfer: Compliant Industrial Assistant

As a result of the technology transfer to KUKA Roboter GmbH, the KUKA lightweight robots are currently used in numerous academic and industrial research labs. The new automation concepts based on this robot allow higher flexibility due to fast work-cell setup and modification, intuitive hands-on programming, and shared workspace for direct interaction and cooperation of humans and robots. The first industrial application was realized by the Daimler factory automation department in Untertürkheim. The system is now used for automatic gear-box assembly in daily production (Fig.8).

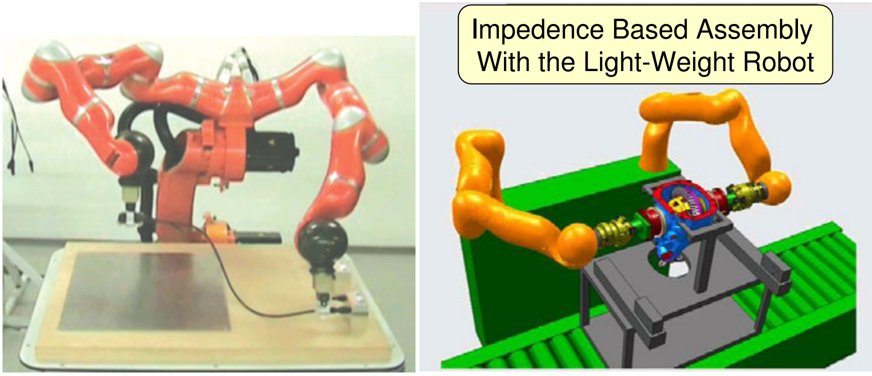


Fig. 8 Left: Demonstration of bimanual flexible object handling by KUKA Roboter GmbH. Right: Impedance based two arm assembly in Mercedes car manufacturing. Courtesy Daimler AG.

In order to establish the new technology in industrial environments, two further key aspects need to be addressed:

- The programmer has to be supported with appropriate toolboxes which help to use and parameterize the various control features of the robot, such as compliance, center of compliance, damping, assembly path, collision detection and reaction strategy, or controller switch for a given application.
- The safety of humans during the permanent interaction with the robots always has to be ensured. The new field of robotic safety in human-robot interaction requires research in biomechanics for understanding injury mechanisms as well as methods for preventing or reducing them.

These two topics are addressed in the next sections.

2.4 Planning Toolbox for Impedance Based Automatic Assembly

Assembly is one of today's the most demanding tasks for industrial robots. Parts have to be brought into contact and aligned properly by the robot despite inevitable uncertainties due to part tolerances, imprecise part feeding and limited robot positioning accuracy. Lack of robustness, extensive setup costs for high-precision part feeding, specialized grippers with so-called *Remote Center Compliance*, and the need for experienced robot programmers are the main reasons, why most assembly tasks are still carried out by humans.

In contrast to current industrial robots, the compliant control features of the DLR light-weight robot allow flexible and robust assembly without additional equipment. The programmer can select high-performance position control for free motion and compliant Cartesian impedance control for highly dynamical interaction with the environment. If desired, the switching between controllers can be triggered by contact detection within 1ms.

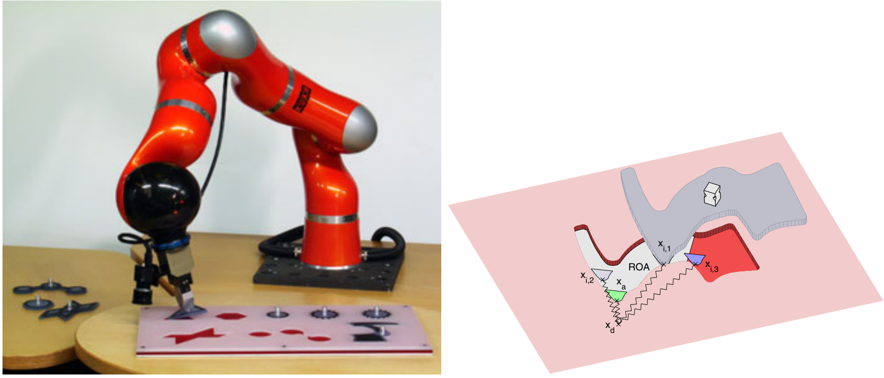


Fig. 9 Left: Experimental setup consisting of a DLR light-weight robot with an industrial gripper, an attached Firewire camera and the pieces and plate on the table. Right: A typical region of attraction (ROA) for a sample part. The inserted corner will be guided automatically to position x_a if the alignment process starts anywhere within the ROA (e.g. from $x_{i,1}$ or $x_{i,2}$). If it starts outside (e.g. from $x_{i,3}$), a successful alignment cannot be guaranteed.

Along with stable contact control, proper alignment of the parts despite inevitable uncertainties is the most challenging part of an assembly task. Usually, this requires tedious and expensive manual optimization of the trajectories for every type of object. In order to simplify this procedure, an algorithm has been developed, which allows automatic planning of robust assembly applications. The algorithm takes the part geometries and information about the expected uncertainties as an input and generates a parameterized robot program for the robust assembly of the parts [16].

The main idea of the insertion planning is visualized in Fig. 9 (Right). Consider the compliance controlled robot having inserted a corner[1] of the part into the hole at the initial configuration x_i. The desired position of the controller is now set to x_d, and the stiffness value to K. For a certain set of starting configurations (called the *region of attraction* - ROA), the inserted part will converge to the desired alignment position x_a. In the given example, $x_{i,1}$ and $x_{i,2}$ belong to the ROA, $x_{i,3}$ does not. The alignment can be seen as the settling of a nonlinear dynamic system with several equilibria, whereof one is the desired configuration. It is possible to determine the ROA for any desired equilibrium x_d and for any stiffness matrix K. Its size can be used as a direct measure for the robustness of the assembly trajectory. The optimal robustness is achieved for those insertion parameters that maximize the ROA.

Obviously, the ROA depends heavily on the inserted corner, the selected desired and initial positions x_d and x_i, the parameters of the impedance control (in particular K), and the shape of the hole. Whereas the latter is given, the remaining parameters can be freely selected and are used for offline optimization. Combined with a user interface for providing the geometries from a CAD system or from sensor data,

[1] Corner in this context means the relevant part of the contour which is involved in a one-point contact.

this toolbox for industrial robot programmers generates robust assembly programs automatically. The output of the toolbox, desired trajectories and control parameters, can then be used in the execution phase without any model knowledge of the parts.

The robustness and performance of the generated assembly strategies were evaluated in extensive experiments with parts having a clearance of less than 0.1mm [17]. The parts are freely placed on a table, located with appropriate image processing, and approached via visual servoing. In order to assess the performance, a comparison with humans in terms of execution time was done. Altogether, 41 persons were tested, whereof 35 were children of age 5–7 and the remaining were adults.

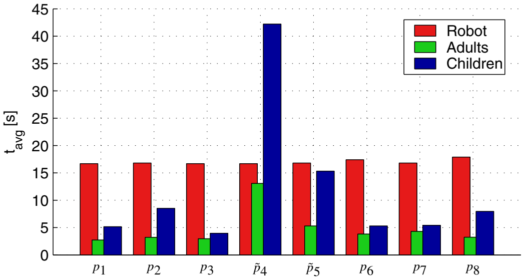


Fig. 10 Average times needed for the different parts. Whereas the robot shows similar performance for all the parts, humans have difficulties especially with the differentiation and insertion of the star shapes as those are difficult to distinguish for humans. $\tilde{p}_4$ represents the star that is inserted first (can be p_4 or p_5), $\tilde{p}_5$ the other one.

Adults needed roughly 30% of the robot's total time for the eight given parts, while children needed about 70%. The variation of the robot performance was low, since the only nondeterministic part of the strategy was the pieces searching (see Fig. 10). Humans, instead, varied their strategy, trying first to solve the problem as fast as possible (accepting failure), and then refined the strategy in subsequent attempts if necessary. Some children needed considerably longer than the robot and were able to fulfil the task only with additional hints.

While free motion and part picking of the humans was considerably faster, in average the robot performed better and more constant in the insertion phase. The experiment shows that the combination of global vision and local force information can be considered as a key to robust and flexible industrial assembly tasks.

2.5 Safety in Human-Robot Interaction

An essential requirement for a robot designed for direct interaction with human users, as e.g. for production assistants, is that it must in no case pose a threat to the human. Until recently, quite few attempts were made to investigate real world threats via collision tests and use the outcome for considerably improving safety during physical human-robot interaction. In this section, we give a brief overview of our systematic evaluation of safety in human-robot interaction with emphasis on aspects related to the LWRIII.

2.5.1 Standardized Crash Tests Experiments for Blunt Impacts

In [18, 19, 20, 21, 22] we analyzed and quantified impact characteristics of blunt robot-human collisions and significantly augmented existing knowledge in this field. The results were obtained and verified with standardized equipment from automobile crash testing, leading to an extensive crash-test report for robots of different size and weight. They range from the LWRIII to heavy duty industrial robots [23, 24].

For the LWRIII all impact tests generated very low injury values by means of standardized severity indices evaluated for the head, neck, and chest. The Head Injury Criterion[2] reached a maximum numerical value of 25 at 2 m/s, which is equivalent to $\approx 0\%$ probability of injury by means of HIC. For both neck and chest similar conclusions could be drawn, since all injury measures were far below any safety critical value [18]. These results were confirmed by impact tests with a human [22]. Even for the case of clamping close to a singularity, which turned out to be the worst-case for the LWRIII, the robot was not able to produce large enough forces to break the frontal bone or endanger the chest of a human, though producing a high quasi-static force of ≈ 1.6 kN.

Apart from such worst-case analysis, we developed effective collision detection and reaction schemes for the LWRIII using the joint torque sensors [25, 26], (see Sec. 2.1.1), which proved to be very effective to reduce the injury potential. Even for the afore-mentioned difficult case[3] we could experimentally verify a reduction of the contact force down to ≈ 500 N for the almost outstretched case. This significantly relaxes the theoretical results of [22].

An important outcome of the extensive experimental campaign is that generally blunt dynamic impacts in free space are, regardless the mass, not dangerous up to an impact velocity of 2 m/s with respect to the investigated severity indices. On the other hand, impacts with (partial) clamping can be lethal, significantly depending on the robot mass. This led us to recommendations for standardized crash-testing procedures in robotics, c.f. Fig. 11. The proposed impact procedures can hopefully provide substantial contributions for future safety standards in physical human-robot interaction.

Apart from blunt impacts, it is of immanent importance to treat soft-tissue injury due to sharp contact.

[2] The Head Injury Criterion (HIC) is the injury severity criterion best known in robotics. Intuitively speaking, a value of 650 corresponds to a 5% probability of staying one day in hospital, while a value of over 1000 can be lethal.

[3] Due to the almost singular configuration, the joint torque sensors are quite insensitive to the clamping force.

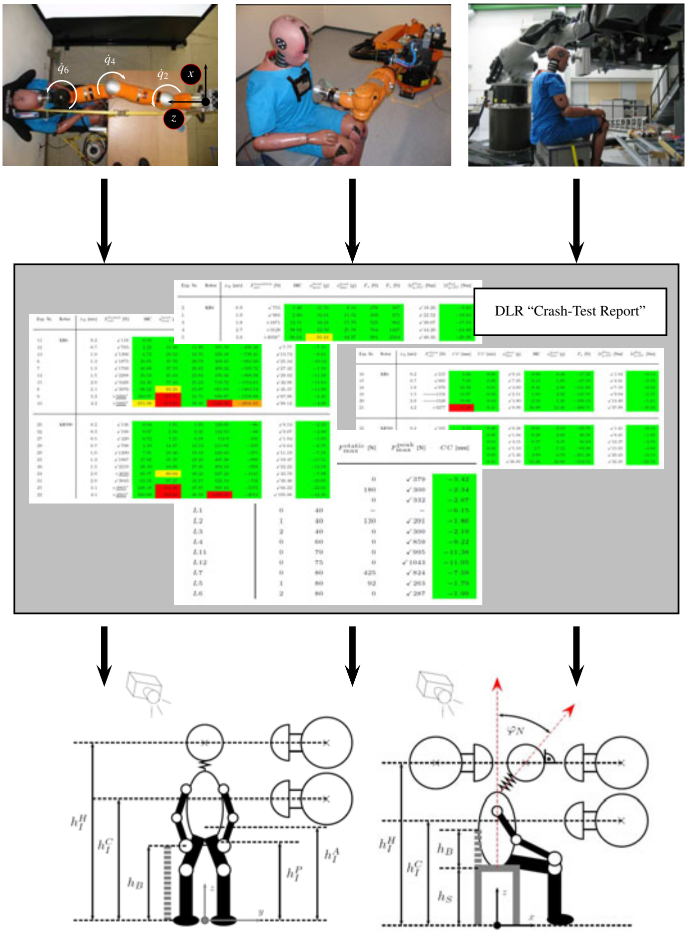


Fig. 11 From impact testing with standardized equipment and evaluation of biomechanical injury criteria to a proposal of standardized impact testing in robotics.

2.5.2 Soft-Tissue Injury Caused by Sharp Tools

A major potential injury source in pHRI are the various tools a robot can be equipped with, see Fig. 12 (left). Their evaluation is still a field with numerous open issues and definitely worth and fruitful to work on. As a first step, we were able to identify the most important injuries and their causes, based on investigations made in the field of forensic medicine and biomechanics. In [27] we presented various experimental results with biological tissue, which validate the analysis. Furthermore, an evaluation of possible countermeasures by means of collision detection and reaction is carried out, c.f. Fig. 12 (right).

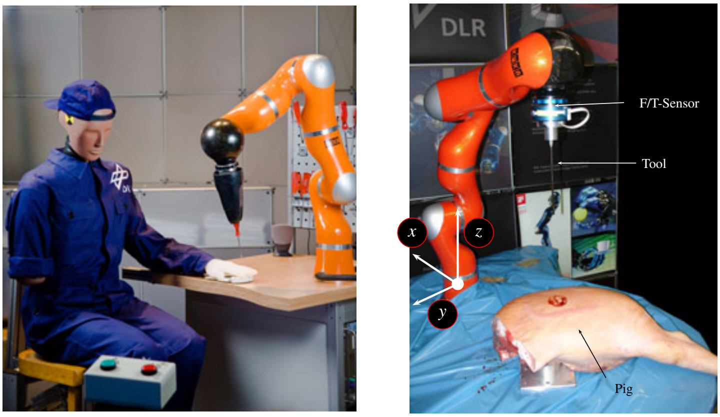


Fig. 12 The co-worker scenario is an example of a robot, which is potentially equipped with dangerous tools, interacting with humans (left). Testing setup for the pig experimental series (right).

It was possible to detect and react to stabbing impacts at 0.64 m/s fast enough to limit the penetration (e.g., of a knife) to subcritical values of several mm's or even prevent penetration entirely, depending on the tool. In case of cutting a full prevention of penetration at a velocity of 0.8 m/s was achieved. Furthermore, we found empirically relevant safety limits for injury prevention for the case of sharp contact, as e.g. the skin deformation before penetration.

3 Increased Performance and Robustness Trough Variable Impedance Actuation

Based on the experience gained with the very successful approach of torque controlled robots, we identified also its limitations and addessed new directions of research for further increasing the robustness, performance and safety of robots. A comparison between current service robots and their human archetype still shows large discrepancy in several aspects. Firstly, relatively tiny impacts can cause severe damages to a robot. The DLR arms and hands are close to their gear-box torque limits when catching a ball of 80g having a speed of 28km/h while for instance a handball goal keeper easily withstands a hit at 120km/h of a 425g ball. In the second case, the impact energy is 100 times larger than in the first case. The "as stiff as possible" mechanical design paradigm and the torque control reach their limits here, because the impact lasts typically only few milliseconds for such a robot. This is too short for the actuator to react and accelerate the motor and gear-box for reducing the impact. This shows that the robustness of robots against impacts can not be addressed by further improvements of torque controlled robots but needs a change of paradigm. The motor has to be partially decoupled from the link side and the induced energy must be stored within the robot joint instead of being dissipated. This directly leads to the necessity of passive elastic elements.

Another important observation is that the velocity and dynamic force capabilities of current robots are by far not good enough to perform dynamic tasks, such as throwing and running, as good as human beings. This can also be improved by the use of mechanical energy storage within the system as exemplified in Sec. 3.3.

Since the specifications for several tasks vary widely regarding position accuracy, speed, and required stiffness, the joint stiffness needs to be variable. This requires an additional motor per joint. To keep the drawbacks of having a second actuator at each joint as low as possible, the joint unit has to be optimized regarding its energy efficiency e.g. at high stiffness presets. The concept of variable impedance actuation[4] (VIA) seems to be a promising solution in this context and its design and control was addressed in numerous publications [28, 29, 30, 31, 32].

Our goal is, based on our experience with torque controlled light-weight robots, to built up a fully integrated VIA hand-arm system for close, safe, high performance interaction with humans while fulfilling the above requirements as close as possible (Fig. 13).

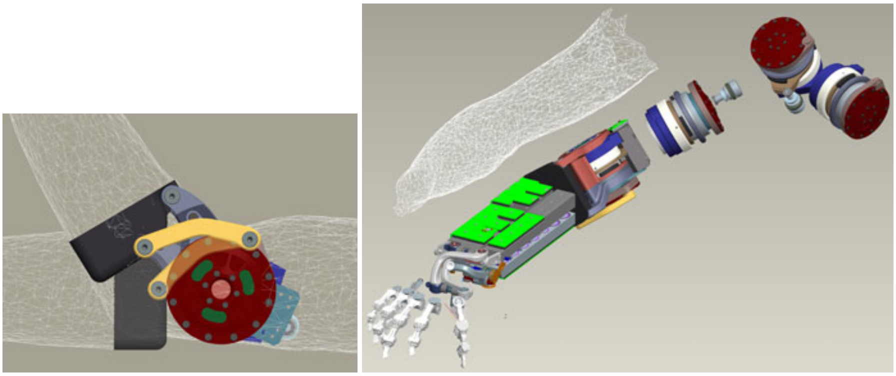


Fig. 13 Current stage of the DLR VIA hand-arm system. Left: Elbow joint. Right: Explosion drawing of the hand-arm system.

3.1 Design of Variable Stiffness Systems

Currently, a hand-arm system with variable compliance is designed at DLR incorporating in a first, concept validation version, several variable compliance joint designs for fingers and arms, see Fig. 13. For the hand, an antagonistic approach is taken, which allows to place the actuators and the variable stiffness mechanics in the forearm and to transmit the motion via tendons through the wrist to the fingers. The fingers and the hand structure are designed to match as close as possible the human

[4] If the joint has only variable stiffness, but no variable damping, the term variable stiffness actuation (VSA) is often used.

hand kinematics and functionality, while finding innovative technological solutions for their implementation [33] (Fig. 13). The wrist is also actuated antagonistically, however in a supporting setup. In such a setup both motors can add their torques to gain the maximum possible torque output or can co-contract to change stiffness for medium load. For the elbow and the shoulder, the focus is on energy efficient and weight minimizing design, such that the mass of the VIA joints do not considerably exceed the weight of an LBWRIII joint. The actuators of these joints are based on an approach in which a tiny motor is primarily used to adapt the stiffness of the joint and a large motor is mainly used to position the link (Fig.14). The currently followed design is a combination of quasi-antagonistic and the variable stiffness joint designs (Fig.15) presented previously in [34, 13].

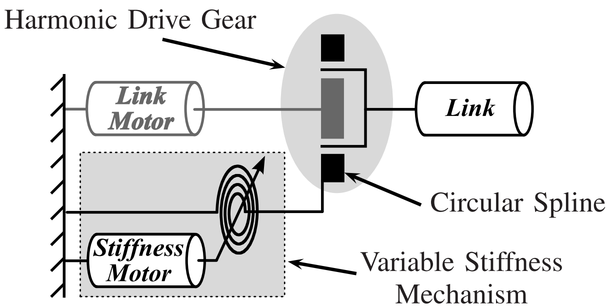


Fig. 14 Actuator and compliance arrangement for the shoulder and elbow joints.

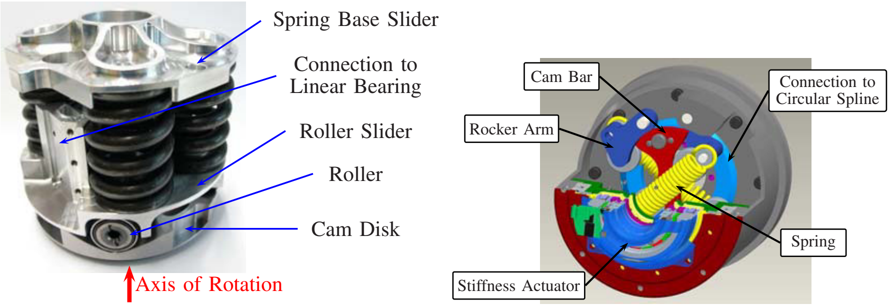


Fig. 15 Design versions of the shoulder and elbow joints. Left: Variable Stiffness Actuator. Right: Quasi-Antagonistic Joint.

3.2 *Control Challenges with VIA Actuators*

The classical control problem formulation for VSA robots is that of adjusting stiffness and position of one actuator and of the entire robotic system (arm, hand) in a decoupled manner, by controlling the position or the torque of the two motors of the actuator [31, 32, 29]. Moreover, in case of VSA structures with many DOF and cable actuation, the decoupling tendon control is treated [35, 36].

In [37] we proposed a new solution for the design of impedance control for coupled tendon systems with exponential stiffness (Fig. 16). The proposed controller provides static decoupling of position and stiffness as well as the exact desired link side stiffness in combination with the intrinsic mechanical compliance, while remaining within the passivity framework of the DLR robots. A second challenging control task is related to the fact that almost all VIA joints designed so far have very low intrinsic damping. While this feature is very useful for movements involving energy storage (e.g. for running or throwing), the damping of the arm for fast, fine positioning tasks has to be realized by control. This can be a difficult task, regarding the strong variation of both inertia and stiffness. Fortunately, the passivity based approaches developed for the torque controlled robots can be adapted for the VIA case. However, it soon became clear from the simulation for the whole arm that a separate control of each joint, by just considering diagonal components of stiffness and inertia matrices as inputs, is not feasible, due to very low stiffness and strong coupling between the compliant joints. New methods for treating the joint coupling were developed starting from [38, 1]. The basic idea for the controller design is the following:

- Consider full coupled inertia and stiffness matrices for the relevant joints.
- Transform the system consisting of link inertia and stiffness to modal coordinates such that the two matrices become diagonal.
- Use torque feedback in order to bring the motor inertia matrix to a structure in correspondence to the double diagonalized matrices, i.e. make it diagonal in the same coordinates.
- Design a decoupled controller in the modal coordinates, independently for each mode. Gains are calculated based on current modal parameters.

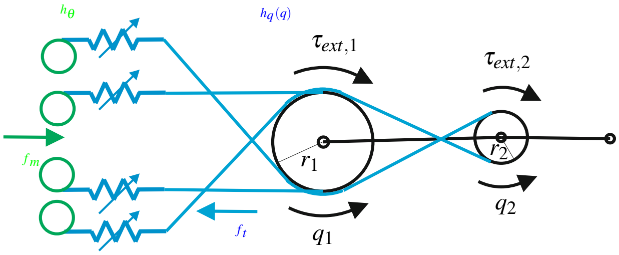


Fig. 16 Example of a tendon network with two joints and four tendons connected by nonlinear springs. h_θ and h_q are the motor and link side tendon displacements, respectively, f_m and f_t are the motor and link side tendon forces.

With this methods, the control proved to work well, as exemplified in the plots from Fig. 17, for one of the three joints. An experimental validation of the controller for high an low stiffness preset on a 1 DOF testbed is shown in Fig. 18.

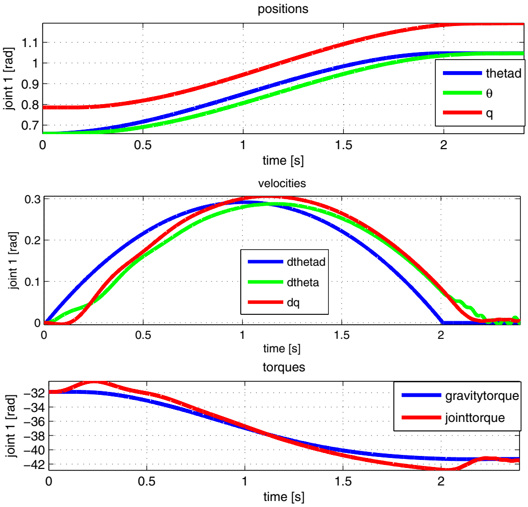


Fig. 17 Motor and link position with state feedback controller.

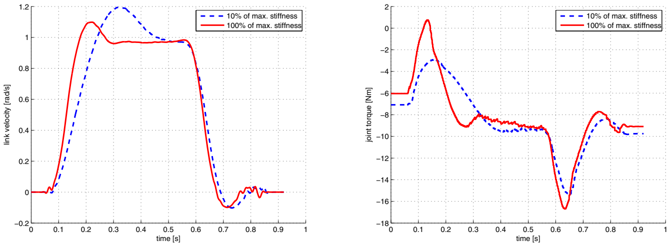


Fig. 18 Motion on a trajectory with rectangular velocity profile for tiny and maximal stiffness. A critically damped velocity step response can be achieved independent from the stiffness and inertia value (upper). The effect of vibration damping is clearly observed in the torque signal (lower).

3.3 Validation of Performance and Robustness

Along with the activities regarding the control of the joint, first analysis and experiments for validating the increase in performance were done.

3.3.1 Throwing

The application of throwing a ball is a good example to show the performance enhancement gained by the VS-Joint in terms of maximal velocity. For throwing a ball as far as possible, it has to be accelerated to the maximum achievable velocity and released at an angle of 45°. The link velocity of a stiff link corresponds to the velocity of the driving motor. In a flexible joint the potential energy stored in the system can be used to accelerate the link relatively to the driving motor. Additional energy can be inserted by the stiffness adjuster of the variable stiffness joint to gain an even faster motion.

Fig. 19 shows simulation results and experimental validation regarding the velocity gain between motor and link for the quasi-antagonistic link. The motor trajectories for optimal performance were generated by an optimal control approach [39]. The link velocity is maximized under constraints on motor velocity and torque, elastic joint deflection range, and controller dynamics.

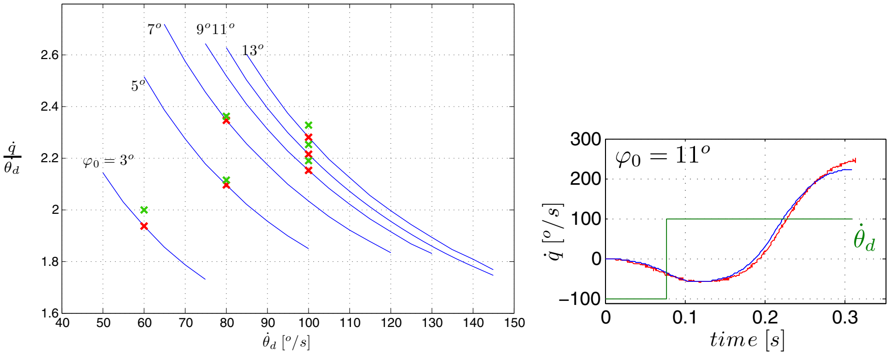


Fig. 19 Left: blue - simulated gain in velocity between motor and link, depending on the maximal desired motor velocity and the stiffness preset. Red - experimentally validated points. Right: Desired motor velocity (grey) and reached link velocity for one experiment (red-simulated, blue-measured).

With the measured maximum link velocity of 572° s^{-1}, the throwing distance for the same experiment with the Variable Stiffness Joint was approximately 6 *m*, corresponding well to the calculated distance of 6.18 *m*. The theoretical throwing distance with an inelastic link of the same setup with the same maximum motor velocity of 216° s^{-1} is 0.88 *m*, also was confirmed experimentally. A speed gain of 265 % for the link velocity between rigid and compliant joint was achieved in the test. Similar results in performance increase have been obtained for kicking a soccer ball, which additionally causes an external impact on the link side, as discussed next.

3.3.2 Experimental Validation of Joint Overload Protection at Impacts

In [40], two series of experiments were conducted to investigate the benefits of passive variable stiffness during impacts. The testing setup for both series was a single DOF joint (with link inertia ≈ 0.57 kg m^2) being hit at a lever length of $\approx$ 0.76 m by a soccer ball (0.45 kg).

In the first series the unloaded joint is kept still and is passively hit by the ball with different impact speeds. The joint torques were recorded for three different setups. Two stiffness setups are realized via the passively compliant VS-Joint. The most compliant as well as the stiffest configuration were chosen. In a third setup a mechanical shortcut is inserted into the test-bed instead of the VS-Joint mechanism, such that a much stiffer joint in the range of the LWRIII elasticity is obtained. Both, increasing impact speed and increasing joint stiffness result in higher peak joint torques as visualized in Fig. 20. The peak torque limit of the joint gear is almost reached with the stiff joint at an impact velocity of ≈ 3.7 m/s, whereas the compliant VS-Joint is still far in the safe torque region.

In the second test series the resting soccer ball was hit by the joint lever at maximum joint velocity. In case of the stiff joint the velocity is limited by the motor. With the VS-Joint, the joint velocity was increased by the energy storage in the joint with a similar trajectory to the one used in Sec. 3.3.1. The results given in Table 1 show a significant increase in joint velocity and kicking range with the VS-Joint which results in a faster impact on the ball. The tests show, however, that the peak joint torque is much tinyer in the flexible joint even though the impact was faster. So the passive flexibility in the VS-Joint does not only help to increase the joint performance, but also reduces the potentially harmful peak joint torques during fast rigid impacts.

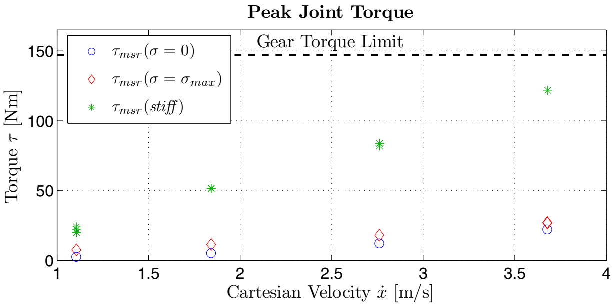


Fig. 20 Peak joint torque during impacts with with a soccer ball. Three different stiffness setups are examined: VS-Joint at low stiffness preset, VS-Joint at high stiffness preset, and an extremely stiff joint without deliberate elasticity (upper). Higher impact velocities result in larger peak torque and passive joint deflection.

Table 1 Results for the different kicking impacts for the VS-Joint and for the rigid joint.

Joint Type	Joint Velocity	Peak Joint Torque	Kicking Range
Stiff Joint	229 deg/s	85 Nm	1.6m
VS-Joint	475 deg/s	10 Nm	4.05m

4 Summary

This paper presented a bird's-eye-view of the paradigm evolution from high performance torque controlled robots to systems with intrinsic variable stiffness. We overviewed the major design and control principles of the torque controlled robot systems developed at the German Aerospace Center (DLR) as an antetype. Torque controlled robots currently represent a technology that is mature for the market. They are used not only as a tools for academic research but also in industrial environments, within new, more flexible automation concepts based on direct cooperation of robots and humans. We believe, however, that impressive research progress can be expected in the area of VSA actuated robots within the next decade. The motivation for variable impedance devices, derived from different performance, robustness, and safety requirements, are highlighted. Possible hardware solutions, which are currently investigated for a newly developed hand-arm system at DLR are described. Finally, first experimental results validating these concepts were presented.

References

1. Albu Schäffer, A., Ott, C., Hirzinger, G.: A unified passivity based control framework for position, torque and impedance control of flexible joint robots. The Int. J. of Robotics Research 26(1), 23–39 (2007)
2. Ott, C., Albu-Schäffer, A., Kugi, A., Hirzinger, G.: On the passivity based impedance control of flexible joint robots. IEEE Transactions on Robotics and Automation 24(2), 416–429 (2008)
3. Albu-Schäffer, A., Haddadin, S., Ott, C., Stemmer, A., Wimböck, T., Hirzinger, G.: The DLR lightweight robot - design and control concepts for robots in human environments. Industrial Robot: An International Journal 134(5), 376–385 (2007)
4. Wimböck, T., Ott, C., Hirzinger, G.: Passivity-based object-level impedance control for a multifingered hand. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 4621–4627 (2006)
5. Pfeffer, L.E., Khatib, O., Hanke, J.: Joint-torque sensory feedback in the control of a puma manipulator. IEEE Trans. Robot. Automation 5, 418–425 (1989)
6. Khatib, O.: A unified approach for motion and force control of robot manipulators: The operational space formulation. RA-3, 43–53 (1987)
7. De Luca, A., Albu-Schäffer, A., Haddadin, S., Hirzinger, G.: Collision Detection and Safe Reaction with the DLR-III Lightweight Manipulator Arm. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS2006), Beijing, China, pp. 1623–1630 (2006)
8. Haddadin, S., Albu-Schäffer, A., De Luca, A., Hirzinger, G.: Collision detection & reaction: A contribution to safe physical human-robot interaction. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 3356–3363 (2008)

9. Tien, L.L., Albu-Schäffer, A., Luca, A.D., Hirzinger, G.: Friction observer and compensation for control of robots with joint torque measurement. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 3789–3795 (2008)
10. Fuchs, M., Borst, C., Robuffo-Giordano, P., Baumann, A., Kraemer, E., Langwald, J., Gruber, R., Seitz, N., Plank, G., Kunze, K., Burger, R., Schmidt, F., Wimböck, T., Hirzinger, G.: Rollin Justin - design considerations and realization of a mobile platform for a humanoid upper body. In: IEEE Int. Conf. on Robotics and Automation, vol. 22, pp. 4131–4137 (2009)
11. Robuffo-Giordano, P., Albu-Schäffer, A., Fuchs, M., Hirzinger, G.: On the kinematic modeling and control of a mobile platform equipped with steering wheels and movable legs. In: IEEE Int. Conf. on Robotics and Automation (2009)
12. Wimböck, T., Ott, C., Hirzinger, G.: Impedance behaviors for two-handed manipulation: Design and experiments. In: IEEE International Conference on Robotics and Automation, pp. 4182–4189 (2007)
13. Albu-Schäffer, A., Eiberger, O., Grebenstein, M., Haddadin, S., Ott, C., Wimbock, G.H.T., Wolf, S.: Soft robotics: From torque feedback controlled lightweight robots to intrinsically compliant systems. IEEE Robotics & Automation Magazine, 20–30 (2008)
14. Wimböck, T., Nenchev, D., Albu-Schäffer, A., Hirzinger, G.: Experimental study on dynamic reactionless motions with DLR's humanoid robot Justin. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (2009)
15. de Santis, A., Albu-Schäffer, A., Ott, C., Siciliano, B., Hirzinger, G.: The skeleton algorithm for self-collision avoidance of a humanoid manipulator. In: IEEE/ASME International Conference on Advanced Intelligent Mechatronics (2007)
16. Stemmer, A., Albu-Schäffer, A., Hirzinger, G.: An analytical method for the planning of robust assembly tasks of complex shaped planar parts. In: Proc. 2007 IEEE Int. Conf. of Robotics and Automation, pp. 317–323 (2007)
17. Robuffo-Giordano, P., Stemmer, A., Arbter, K., Albu-Schäffer, A.: Robotic assembly of complex planar parts: An experimental evaluation. In: IROS, pp. 3775–3782 (2008)
18. Haddadin, S., Albu-Schäffer, A., Hirzinger, G.: Safety Evaluation of Physical Human-Robot Interaction via Crash-Testing. In: Robotics: Science and Systems Conference (RSS 2007), Atlanta, USA, pp. 217–224 (2007)
19. Haddadin, S., Albu-Schäffer, A., Hirzinger, G.: The Role of the Robot Mass and Velocity in Physical Human-Robot Interaction - Part I: Unconstrained Blunt Impacts. In: IEEE International Conference on Robotics and Automation (ICRA 2008), Pasadena, USA, pp. 1331–1338 (2008)
20. Haddadin, S., Albu-Schäffer, A., Hirzinger, G.: The Role of the Robot Mass and Velocity in Physical Human-Robot Interaction - Part II: Constrained Blunt Impacts. In: IEEE Int. Conf. on Robotics and Automation (ICRA 2008), Pasadena, USA, pp. 1339–1345 (2008)
21. Haddadin, S., Albu-Schäffer, A., Hirzinger, G.: Safe Physical Human-Robot Interaction: Measurements, Analysis & New Insights. In: International Symposium on Robotics Research (ISRR 2007), Hiroshima, Japan, pp. 439–450 (2007)
22. Haddadin, S., Albu-Schäffer, A., Hirzinger, G.: Requirements for safe robots: Measurements, analysis, & new insights. International Journal of Robotics Research (2009) (accepted)
23. Haddadin, S., Albu-Schäffer, A., Frommberger, M., Rossmann, J., Hirzinger, G.: The DLR Crash Report: Towards a Standard Crash-Testing Protocol for Robot Safety - Part I: Results. In: IEEE Int. Conf. on Robotics and Automation (ICRA 2008), kobe, Japan, pp. 272–279 (2009)

24. Haddadin, S., Albu-Schäffer, A., Frommberger, M., Rossmann, J., Hirzinger, G.: The DLR Crash Report: Towards a Standard Crash-Testing Protocol for Robot Safety - Part II: Discussions. In: IEEE Int. Conf. on Robotics and Automation (ICRA 2008), Kobe, Japan, pp. 280–287 (2009)
25. De Luca, A., Albu-Schäffer, A., Haddadin, S., Hirzinger, G.: Collision Detection and Safe Reaction with the DLR-III Lightweight Manipulator Arm. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS 2006), Beijing, China, pp. 1623–1630 (2006)
26. Haddadin, S., Albu-Schäffer, A., Luca, A.D., Hirzinger, G.: Collision detection & reaction: A contribution to safe physical human-robot interaction. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS 2008), Nice, France, pp. 3356–3363 (2008)
27. Haddadin, S., Albu-Schäffer, A., De Luca, A., Hirzinger, G.: Evaluation of Collision Detection and Reaction for a Human-Friendly Robot on Biological Tissue. In: IARP International Workshop on Technical challenges and for dependable robots in Human environments (IARP 2008), Pasadena, USA(2008), http://www.robotic.de/Sami.Haddadin
28. Morita, T., Iwata, H., Sugano, S.: Development of human symbiotic robot: Wendy. In: IEEE Int. Conf. of Robotics and Automation, pp. 3183–3188 (1999)
29. Bicchi, A., Tonietti, G.: Fast and Soft Arm Tactics: Dealing with the Safety-Performance Trade-Off in Robot Arms Design and Control. IEEE Robotics and Automation Mag. 11, 22–33 (2004)
30. Migliore, S.A., Brown, E.A., DeWeerth, S.P.: Biologically Inspired Joint Stiffness Control. In: IEEE Int. Conf. on Robotics and Automation (ICRA 2005), Barcelona, Spain (2005)
31. Palli, G., Melchiorri, C., Wimboeck, T., Grebenstein, M., Hirzinger, G.: Feedback linearization and simultaneous stiffness-position control of robots with antagonistic actuated joints. In: IEEE Int. Conf. on Robotics and Automation (ICRA 2007), Rome, Italy, pp. 2928–2933 (2007)
32. Vanderborght, B., Verrelst, B., Ham, R.V., Damme, M.V., Lefeber, D., Duran, B.M.Y., Beyl, P.: Exploiting natural dynamics to reduce energy consumption by controlling the compliance of soft actuators. Int. J. Robotics Research 25(4), 343–358 (2006)
33. Grebenstein, M., Smagt, P.v.d.: Antagonism for a highly anthropomorphic hand arm system. Advanced Robotics 22, 39–55 (2008)
34. Wolf, S., Hirzinger, G.: A new variable stiffness design: Matching requirements of the next robot generation. In: IEEE Int. Conf. on Robotics and Automation, pp. 1741–1746. IEEE, Pasadena (2008)
35. Kobayashi, H., Ozawa, R.: Adaptive neural network control of tendon-driven mechanisms with elastic tendons. Automatica 39, 1509–1519 (2003)
36. Tahara, K., Luo, Z.-W., Ozawa, R., Bae, J.-H., Arimoto, S.: Bio-mimetic study on pinching motions of a dual-finger model with synergistic actuation of antagonist muscles. In: IEEE International Conference on Robotics and Automation, pp. 994–999 (2006)
37. Wimböck, T., Ott, C., Albu-Schäffer, A., Kugi, A., Hirzinger, G.: Impedance control for variable stiffness mechanisms with nonlinear joint coupling. In: IEEE Int. Conf. on Intelligent Robotic Systems, pp. 3796–3803 (2008)
38. Tien, L.L., Albu Schäffer, A., Hirzinger, G.: Mimo state feedback controller for a flexible joint robot with strong joint coupling. In: Int. Conf. on Robotics and Automation (ICRA), pp. 3824–3830 (2007)
39. Weis, M.: Optimalsteuerung eines Robotergelenks mit einstellbarer passiver Steifigkeit. DLR and TU Karlsruhe Master Thesis (2009)
40. Haddadin, S., Laue, T., Frese, U., Wolf, S., Albu-Schäffer, A., Hirzinger, G.: Kick it like a safe robot: Requirements for 2050. Robotics and Autonomous Systems: Special Issue on Humanoid Soccer Robots 57, 761–775 (2009)

Human Robot Interaction

Cédric Pradalier

In this part of the ISRR'09 proceedings, we have assembled a group of papers focusing on the way these complex machines we call robots can interact with their users, how they can show us what they perceive, listen to us, work with us, or even help us record the sense of touch.

- In “Real-Time Photorealistic Virtualized Reality Interface For Remote Mobile Robot Control”, Kelly et al. describe the challenges related to the remote control of a mobile platform based on the transmission of its sensors. The main contributions are on one hand the insight on the importance of latency compensation for high-speed remote driving, validated by user studies, and on the other hand the report on the technologies required to achieve this impressive integration effort.
- In “Robot Audition: Missing Feature Theory Approach and Active Audition”, Okuno et al. report on the development and application of the HARK robot audition software. The importance of this work for the robotics community is its capability to separate multiple sources and to localize moving talkers.
- In “Haptography: Capturing and Recreating the Rich Feel of Real Surfaces”, Kuchenbecker defines haptography for the sense of touch as what photography is to the sense of sight. The challenge here is to be able first to record with high fidelity the high frequency vibration and forces we describe as the sense of touch, and then to reproduce these forces and vibration with high enough accuracy to preserve and transmit the sense of touch.
- In “Towards the Robotic Co-Worker”, Haddadin et al. present a solution for the implementation of a safe working collaboration in pick-and-place industrial scenario. This application is made possible by the integration of soft-robotics, robust sensing, and flexible hybrid automata to react to exceptional situations.

We decided to group these papers together because we believe that they provide an important insight on what technologies will have to be integrated to the future robotic systems, should they one day leave the confines of laboratories to tackle more complex tasks than vacuum cleaning or lawn mowing.

Real-Time Photorealistic Virtualized Reality Interface for Remote Mobile Robot Control

Alonzo Kelly, Erin Capstick, Daniel Huber, Herman Herman, Pete Rander, and Randy Warner

Abstract. The task of teleoperating a robot over a wireless video link is known to be very difficult. Teleoperation becomes even more difficult when the robot is surrounded by dense obstacles, or speed requirements are high, or video quality is poor, or wireless links are subject to latency. Due to high quality lidar data, and improvements in computing and video compression, virtualized reality has the capacity to dramatically improve teleoperation performance — even in high speed situations that were formerly impossible. In this paper, we demonstrate the conversion of dense geometry and appearance data, generated on-the-move by a mobile robot, into a photorealistic rendering database that gives the user a synthetic exterior line-of-sight view of the robot including the context of its surrounding terrain. This technique converts remote teleoperation into line-of-sight remote control. The underlying metrically consistent environment model also introduces the capacity to remove latency and enhance video compression. Display quality is sufficiently high that the user experience is similar to driving a video game where the surfaces used are textured with live video.

1 Introduction

Effective operation of any mobile platform without direct line-of-sight is intrinsically difficult to achieve. In video-based teleoperation, the loss of peripheral vision caused by viewing the world through the soda straw of a video camera reduces driving performance, and it increases the operator's frustration and workload. Wireless

Alonzo Kelly · Dan Huber · Herman Herman · Pete Rander · Randy Warner
Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA 15213
e-mail: name@rec.ri.cmu.edu

Erin Capstick
DCS Corporation. 1330 Braddock Place, Alexandria, VA, USA, 22314
e-mail: ecapstick@dcscorp.com

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 211–226.
springerlink.com

communication links are also subject to dropouts and high levels of latency. Their bandwidth limitations typically cause a large reduction in image quality relative to the fidelity of the underlying video cameras. When the robot undergoes significant or abrupt attitude changes, the operator response may range from disorientation, to induced nausea, to dangerous mistakes. The need for high attention levels also deprives operators of the capacity to pay attention to their surroundings. Wireless communications issues and difficulty controlling the robot also increase time on task, and they increase the time required to become a skilled operator.

1.1 Motivation and Technical Approach

The term virtual<u>ized</u> reality [7] refers to the production of views of a rendering database where the geometry and appearance content is derived from measurements of a real scene. For visualization, such a database makes it possible to render synthetic views of the scene from arbitrary perspectives that may never have been the site of any real sensor. Such techniques represent an extreme on a spectrum of real data content with augmented or mixed reality somewhere in between and virtual reality at the other extreme. Virtualized reality enables a new capacity to address many of the problems described above by providing a photorealistic, synthetic, line of sight view to the robot based on the content of geometry-augmented real-time video feeds.

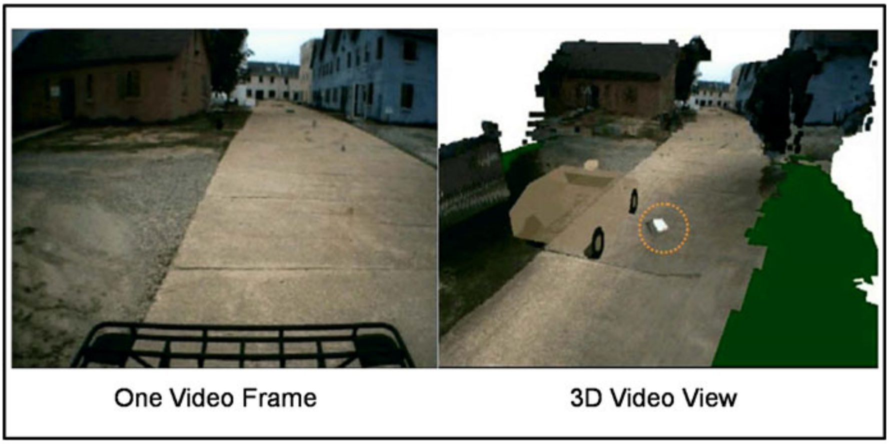


Fig. 1 3D Video View of a Mobile Robot. Left: A video frame produced from a camera on a moving vehicle. Right: The 3D Video view produced from all of the video that has been received in the last few seconds by the vehicle. The operator can look at this database from any angle, at any zoom, while it continues to be updated in real time. The vehicle is synthetic since no sensor has imaged it, but the rest of the content is generated from live video produced by the forward looking sensor mounted on the vehicle roof.

If the processing is performed in real-time, a kind of hybrid 3D Video (Figure 1) is produced that can be viewed from arbitrary perspectives while exhibiting the photorealism and dynamics of live video. The operator experience is equivalent to following the robot in a virtual helicopter that provides arbitrary viewpoints including an overhead viewpoint and the over-the-shoulder view that is popular in video games.

A large number of benefits can be realized with this approach to user interfaces.

- The operator can see any part of the hemisphere around the vehicle at any time.
- The display provides a natural mechanism to introduce augmented reality operator aids.
- The viewpoint is stabilized regardless of the attitude (pitch and roll) changes of the real vehicle.
- Viewpoints can be customized and switched for each task. For example, parallel parking is much easier when the vehicle is viewed from above, and objects can be examined closely by zooming in on them.
- Multiple viewpoints can be shown at once, and multiple operators or observers can view the scene independently.
- The frame rate of the display can be adjusted independently from that of the underlying video feed.
- Dropped frames can be easily tolerated by rendering the most recent model.
- Deliberate dropping of frames or many other schemes can be used to compress the data transmission.
- Even with a vehicle moving in the display, latency can be essentially eliminated by rendering the predicted vehicle state.
- A photorealistic map of the area traversed is produced as a byproduct of system operation.

The engineering difficulty of producing and processing such data is considerable, and it is even more difficult if it must be photorealistic and produced in part from data feeds of scanning lidar sensors derived from a continuously moving vehicle in natural terrain. Nonetheless, the rewards of such efforts are also considerable as we hope to show in the sequel.

1.2 Related Work

The technique of view interpolation was introduced in computer graphics [2] as a mechanism to exploit depth information in order to efficiently produce synthetic imagery while bypassing many of the computationally expensive geometric aspects of rendering. View interpolation is one of several approaches to image-based rendering. Such techniques achieve remarkable realism through the use of natural imagery to texture surfaces. Given depth, a purely synthetic view of a real scene can be produced by projecting the pixels of an image to their proper 3D locations and reprojecting them onto a new image plane. Kanade coined the term virtualized reality [7] to emphasize that the image data was natural rather than the synthetic imagery used in virtual reality. Initial virtualized reality work was based on off-line stereo ranging and stationary sensors that surrounded a localized dynamic scene. More recently real-time image-based rendering has been accomplished for a single discrete object and fixed cameras based on a visual hull method for computing depth [11]. While computer vision and computer graphics evolved toward virtualized reality, telerobotics was simultaneously developing augmented reality displays for more effective remote control. Numerous techniques for supervisory control and teleoperation of manipulators, and even telepresence, were clearly outlined as early as the mid 1980s [16]. The same concepts were considered early for legged vehicles [12] and wheeled Mars rovers [1]. Given the sensor data needed, the earliest approaches to teleoperation simply displayed the raw sensor data or showed the robot in a 2D overhead view in the context of its surrounding perceived objects. Applications like space exploration generated a strong impetus to develop more realistic virtual displays as early as 1991 [5]. The potential of augmented reality environments has been explored in both nuclear servicing [13] and space [10] contexts. In these cases, a small amount of virtual information was rendered over natural video. Innovations included registration of the virtual model to reality using vision, and latency compensation using motion preview and predictive displays.

These techniques can be applied with more effort to robot vehicles. One sustained research effort in the use of virtual environments for the control of robot vehicles was the Virtual Environment Vehicle Interface (VEVI) described in [6]. This system was tested terrestrially [4], and derivatives were ultimately used on the Mars Pathfinder mission. Contemporary developments include more emphasis on sensor fusion [3] as well as efforts that display both forms of data (appearance and geometry) in a less integrated but more useable way [18]. The VEVI system is a clear landmark in related work and it is closest to the work we present here. Our work is distinct from VEVI in that VEVI did not perform image-based rendering and hence did not use virtualized reality. VEVI did render false color terrain maps produced by on-board lidar sensing for a slow moving legged vehicle and this achievement was unprecedented using the technology of that period. VEVI used a classical form of latency compensation based on vehicle autonomy and supervisory control interfaces but it did not perform the kind of high fidelity continuous motion prediction in virtualized reality that we will present here. We also achieve results in data compression and unprecedented vehicle speeds that derive respectively from the commitment to virtualize the entire scene and the use of custom photogeometric sensing as described below.

2 Hardware and Architecture

Virtualized reality constructs a computer graphics model of a real scene. The set of geometrically consistent graphics primitives to be displayed will be referred to as the model. For teleoperation, a key design decision is the location of the model building process. If modeling is performed on the vehicle processor, then model updates can be communicated to the remote operator control station and communications bandwidth requirements can presumably be reduced. Reduction is possible because

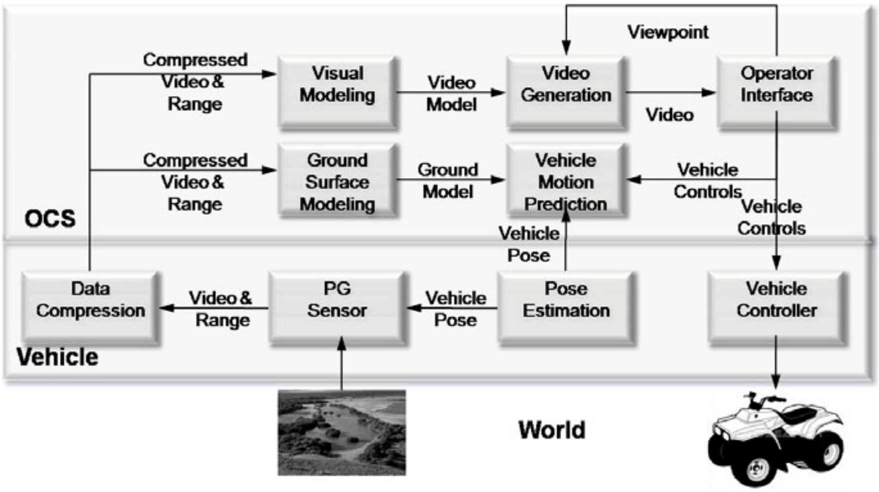


Fig. 2 System Architecture. The system includes an operator control station (OCS) and a remote-control retrofit of a standard all terrain vehicle.

it takes less bandwidth to transmit a fused model that lacks the redundant content of video. If modeling is performed on the remote operator control station, raw sensor data must be communicated, and bandwidth requirements are higher. Despite this bandwidth cost, we chose the second option (Figure 2) in our implementation, in part, because the latency compensation process, discussed later, is more straightforward. In this case, operator commands to the robot can be observed (at the operator control station) without significant latency.

The basic hardware setup at the system level involves an operator control station (OCS) that communicates over wireless to a remotely located mobile robot equipped with photogeometric sensing.

2.1 Sensing

The term appearance will be used to refer to sensing modalities that are sensitive to the intensity of incident radiation including visible color, visible intensity, and visible or invisible infrared modalities. Conversely, geometry will be used to refer to modalities that register any of depth, range, shape, disparity, parallax, etc. The term photogeometric (PG) sensor will refer to a sensing device that produces both kinds of data in a deeply integrated manner. For our purpose in this paper, the data is deeply integrated if the spatial correspondences of the data are known. Ideally, as shown in Figure 3, the resolutions are matched as well so that a one-to-one mapping exists between geometry and appearance pixels.

Computational stereo vision is a natural and classical approach to photogeometric sensing because range is produced for every pixel in the reference appearance image. However, its utility in our application is limited due to the relatively poor

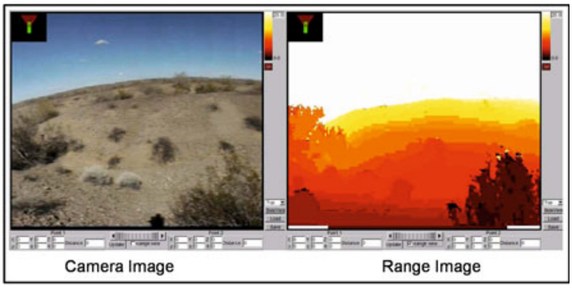


Fig. 3 Photogeometric Data Set. Every color pixel in the left image has an associated range pixel in the right image. Sensors that produce such data do not exist on the market today but they can be constructed by integrating more basic components.

quality of the range data. Our implementation approach therefore produces an integrated data set of appearance and geometry data from two distinct sensors: lidar and video.

Due to many considerations including the numerous robotic platforms that we construct annually and the desire to standardize solutions across programs, we have been continuously refining our photogeometric sensor concept for many years. A recent sensor design is shown in Figure 4. For scanning lidars, we typically purchase an off the shelf scanning lidar that scans in one degree of freedom (called the fast axis), and then we actuate the housing in a second degree of freedom (called the slow axis) in order to produce a scanning pattern that spans a large angle in both azimuth and elevation.

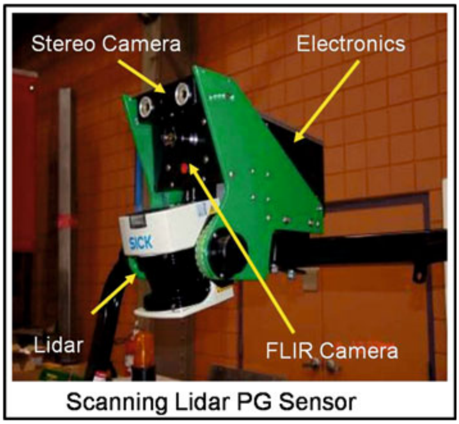


Fig. 4 Custom Photogeometric Sensor. The device fuses data from a commercial scanning lidar by SICK, stereo cameras, and a forward looking infrared (FLIR) camera. The interface to the composite device is a combination of fast Ethernet (used for high bandwidth data) and CAN Bus (used for low latency control).

The lidar pointing control system provides precisely timed feedback on the angle of rotation. This data stream is merged with the range and angle data coming from the lidar to form a 2D scanning lidar data stream. This stream is then optionally merged with any camera data and transmitted to the host computer system. In our autonomy systems, it is useful to merge the data at the level of individual imagery. However, for visualization, we instead merge the data later in the model building process at the level of an integrated geometric model of the scene.

2.2 *Vehicle*

Our latest vehicle test bed (Figure 5) is a custom retrofitted LandTamer® amphibious remote access vehicle. We chose this vehicle for its terrainability, ease of transport, and (deliberately) for the difficulty of modeling its skid steering.

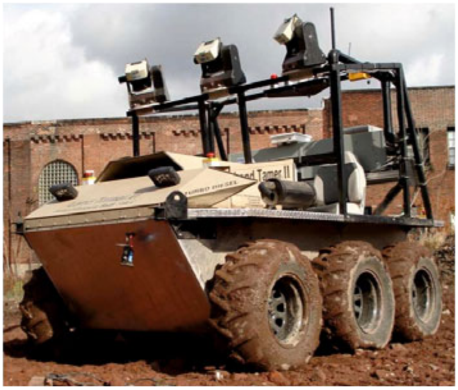


Fig. 5 Robot Vehicle. A LandTamer® vehicle was retrofitted for remote control. Three custom colorized range (CR) sensors with a total field of view of 160° are mounted high on the vehicle looking forward. The lidars are manufactured by SICK providing range points at 2 KHz separated by ½ degree of angle over 180° of field of view. The cameras are the Firefly® by Pt. Grey Research Inc., and they provide color imagery at 720 X 500 resolution over a 60° field of view.

A custom field programmable gate array (FPGA) board is used to implement the servos that control the nodding motion of the lidars. It is also used to integrate the data into time tagged colorized range data sets and to provide the results over an ethernet link to the main vehicle computer.

A Novatel SPAN INS-GPS system is used for pose estimation, including the vendor's Kalman filter. The system is augmented by a portable real-time kinematic (RTK) differential base station. Under favorable satellite viewing conditions 2cm accuracies are achievable. A small computing cage houses the sensor control and data acquisition FPGA board and two Intel® Core™ Duo processors. These processors concentrate the data from all sensors and send it to the OCS over 802.11g wireless. They also receive the OCS commands over the same wireless link and pass them to the vehicle controller.

2.3 Operator Control Station (OCS)

The OCS (Figure 6) incorporates a steering wheel and throttle and brake pedals as well as a large LCD monitor. The OCS also contains a processor capable of both communicating with the robot and rendering the data on the display.

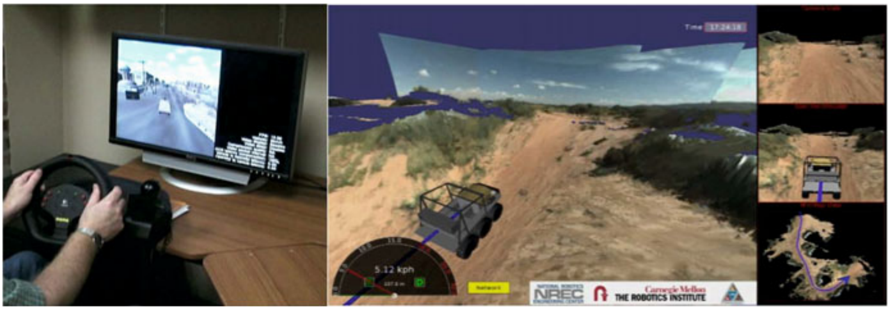


Fig. 6 OCS. The Operator Control Station includes a steering wheel equipped with selection buttons, foot pedals, and a large LCD display. The display provides selectable views including the raw video, over-the-shoulder, and birds-eye (direct overhead). The selected view is enlarged and the others are reduced in size and shown to the right.

3 Modeling and Visualization Algorithms

In order to achieve photorealism, we aspire to produce geometry for every camera pixel (rangefied color). Rather than do this in the image, however, results are improved if geometry is interpolated in the scene by fitting surfaces to the lidar (geometry) data and projecting the camera (appearance) data onto those surfaces.

Numerous effects give rise to situations where the color of a scene point is known, though its range is not. For many reasons, lidar data produced on a ground vehicle ceases to be reliable beyond a range on the order of 30 meters. Let the region beyond this range be known as the far field, and let that inside this range be known as the near field. Even in the near field, the reduced angular resolution of lidar relative to cameras implies that the vast majority of near field color pixels in a camera image will not have a lidar pixel that corresponds directly.

A second important issue is range shadows. It is necessary in general to depth buffer the range data from the camera perspective in order to ascertain which ranged points are occluded by others and therefore have unknown color. When the viewpoint differs significantly from that of the lidar sensor, substantial missing parts in the model become possible. For our purposes, the required precision of geometry depends on the offset of the viewpoint from the original sensor viewpoint. When the offset is small, substantially incorrect geometry will still look highly realistic. When the offset is large, differences in parallax of scene points from their correct parallax will result in distortion that is noticeable to the operator.

In general, four classes of points can be distinguished. The system uses several strategies described below to process them.

- **Surface and texture known.** This is the easiest case where the texture is projected onto the surface.
- **Texture only known.** In this case, the geometry has to be assumed or the data rejected. Two cases of practical interest are under-sampled smooth surfaces, and regions beyond the lidar maximum range.
- **Geometry only known.** Enough cameras can be used to ensure that this case does not occur with two exceptions. First, the vehicle does not normally produce a lidar image of itself, but its geometry can be measured or coded offline. Second, regions occluded by other surfaces can be drawn in an unusual color to identify them to the operator, and both sensor separations in space and image separations in time can be minimized to the degree possible to mitigate this effect.
- **Nothing known.** Once the possibility exists to place a viewpoint anywhere, scenes with complex geometry will often show holes in the model that correspond to regions that no sensor was able to see for occlusion reasons. This is the cost of arbitrary viewpoints applied to data imagery from a specific viewpoint. There is no way in general to generate the missing data, but the advantages of arbitrary viewpoints can outweigh this imperfection.

Of course, regions of the scene may become unknown over the passage of time when the scene is dynamically changing. In such cases, omnidirectional lidars and cameras may be used to continuously update the view in all directions.

3.1 Near Field Surface Modeling

The application to ground vehicles justifies the assumption that the environment around the vehicle includes a ground surface and optional objects that may lie on it. In many environments, lidar data is sufficiently dense, out to 20 to 30 meters, to sample the terrain surface adequately for its reproduction. For this reason, the implementation segments all lidar points into those that lie on the ground and those that lie above it.

In forested environments, situations like overhanging branches invalidate the assumption that height is a single-valued function of position. Therefore, all lidar data is initially accumulated in a 3D voxelized, gravity-aligned data structure, called the point cube, before it is segmented. Each voxel counts the number of lidar beams that have terminated in the voxel (called hits), and the number that have passed through (called pass-throughs) to terminate in voxels further from the sensor. After each full sweep of the lidar beam, the point cube is analyzed to determine the lowest cell in each vertical column with enough hits to determine a ground surface. The average height of these hits is used to define the ground height at the position of the voxel in a new data structure called the terrain map. This structure is a horizontal 2D array arranged into cells (20 cm on a side) that store the ground height at the center of each cell. The terrain map accumulates all data from multiple lidar scans. Its spatial extent, like the point cube, is limited to some adjustable region around the present vehicle position defined in terms of 3D space, distance, or time.

3.2 Projective Texture Mapping

Each cell in the terrain map is converted to two untextured triangles that must then be textured from the camera imagery. Densely populated voxels are those containing a large number of lidar hits. Those that are above the ground surface, but are not dense enough to define a surface, have their geometry hallucinated as the sides of the voxel. The points in sparsely populated voxels are enclosed in very small cubes to create geometry onto which to render their textures.

The baseline separation between camera and lidar can unfortunately be enlarged significantly due to asynchrony of the camera and lidar during periods of vehicle motion. Also, camera imagery may overlap due to multiple overlapping fields of view or multiple frames captured over time. Unless depth buffering is performed, the same textures will be painted onto foreground objects as well as those background objects that are occluded by them. This would not be a problem if the terrain map was the only surface in the scene, but there are others above it. Therefore, we instead use projective texture mapping [15] implemented in the OCS graphics processing unit (GPU) to paint the video onto the geometry. The system maintains a list of the most recent images from all cameras. Each image is used to apply texture to the geometry in the scene in temporal order so that cells that fall outside the field of view of more recent images will retain the last texture painted onto them.

3.3 Far Field Modeling and Visualization

Often, the far field region of space corresponds to the higher parts of the images, and it extends to the horizon. For such data, we erect a temporary surface (a billboard) that is normal to each camera's optical axis. The camera data is then projectively textured onto the surface.

The billboards move with the vehicle (Figure 7). Provided the viewpoint is not significantly different from the camera, the parallax error is tolerable, and operators overwhelmingly prefer their use. In the case of an overhead view, the billboards become normal to the viewing axis, and they mostly disappear.

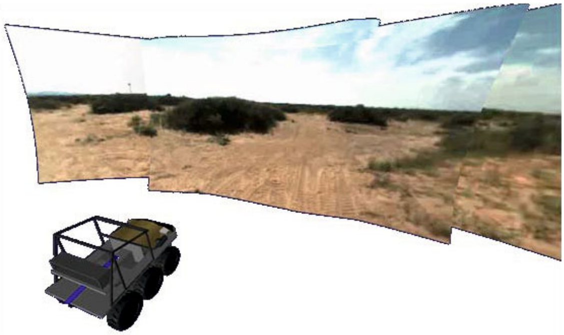


Fig. 7 Billboards Used to Display Far Field Video. This view shows the geometry of the three billboards and how video frames are pasted onto them. The technique of using billboards for complex scenes has been used for many years [14].

4 Teleoperation Algorithms

So far, we have described the basic mechanisms for producing and rendering a photorealistic model that surrounds a moving vehicle. This section describes how this basic mechanism is augmented to produce a teleoperation system.

4.1 Latency Compensation and Simulated Leader-Follower

Any real wireless communications system will introduce bidirectional latency into the telemetry passing between the vehicle and the OCS. It is well known that such latency is one factor that makes teleoperation difficult. The capacity to render the vehicle from an external viewpoint not only provides hemispherical exterior context to the operator but it also provides the opportunity to remove latency using prediction (Figure 8). Our vehicle display is virtual anyway so it is straightforward to draw the vehicle in any position. We render the vehicle at its predicted position at the time in the future when commands issued by the operator will arrive at, and be acted upon, by the vehicle. This technique produces a continuously predictive display that appears to respond with no latency. Given the capacity to predict the future, a potentially more useful technique is to predict slightly beyond the command arrival time to produce a display of a vehicle slightly more into the future. In this case, some of the prediction error has not happened yet, and the real robot can be given the responsibility to reduce the error before it occurs. This is accomplished by considering the simulated vehicle to be a lead vehicle, which the real one is obliged to follow. The simulated lead vehicle is rendered in the context of the 3D Video feed that is updated to include new information as the real (but usually not displayed) vehicle moves. Hence, the operator has the sensation that a real vehicle is being controlled. In this case, the path followed by the leader is passed to the real vehicle as the command input.

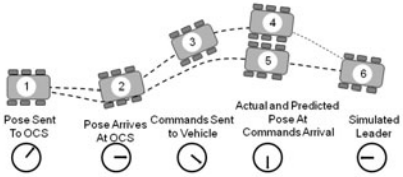


Fig. 8 Prediction for Latency Compensation. Prediction of station 5 can be used for latency-free rendering. Errors in the initial state and prediction process may place the vehicle at station 5 when it will really arrive at station 4. Such errors can be treated as following errors if a simulated leader is rendered at station 6.

4.2 Telemetry Compression

Virtualized reality creates an opportunity to implement effective data compression in remote control applications. The fact that the rendering of the model is decoupled from the frame rate of the cameras means that the update of the display, showing a moving vehicle, can proceed at high rates regardless of the camera rates and the pose update rates. This capacity also implies a high degree of robustness to dropped frames of video or pose data since each appears to be a momentary increase in input latency for which the system is already configured to compensate. The input pose data and output control signals are small enough to be irrelevant to the compression issue. Lidar data is converted from floating point to 16 bit integers. For the video, we use an Xvid (MPEG-4) encoder on the vehicle and a decoder on the OCS to perform compression. Visible parts of the vehicle are cropped. The three streams from the sensor pods are reduced to 10Hz frame rate before they are compressed. Based on just these measures, we are able to produce very high quality 3D Video displays that compete with full frame video using only 1 Mbit /sec of communication data rates.

4.3 Augmented Reality and Mixed Initiative Interactions

The capacity to produce metrically accurate photorealistic maps follows from a decade of work in using colorized range data for robot autonomy systems. Many of the data structures involved are identical or similar to those used to represent the scene for autonomy purposes. In particular, both the point cube and the terrain map are standard components of our autonomy systems, which are produced for the purpose of obstacle and hazard detection [8]. Given such algorithms, it is natural to wonder how they can be used to help the operator drive. Figure 9 shows a simple augmented reality display where the classifications of simple slope based obstacle detection algorithms are used to partially color the terrain. The colors are blended with the video such that reddish areas are to be avoided, and greenish ones are safe for driving. In benign terrain, in broad daylight, this augmented reality facility may not add much value. However, when terrain is difficult or lighting or visibility is poor, such an autonomy system could add value if the human interface were configured correctly. Lidar works better at night due to reduced solar interference, and infrared appearance data can be processed like daytime video to produce a system that permits an operator to drive in total darkness. The 3D Video system also uses many of the same data structures for rendering and autonomy, so the operator and the autonomy system can interact more readily through the display; augmented reality is but one such mechanism. It is possible to have autonomy veto operator commands or bring the vehicle to stop, and the display can likely provide the operator with the reason for the robot initiative.

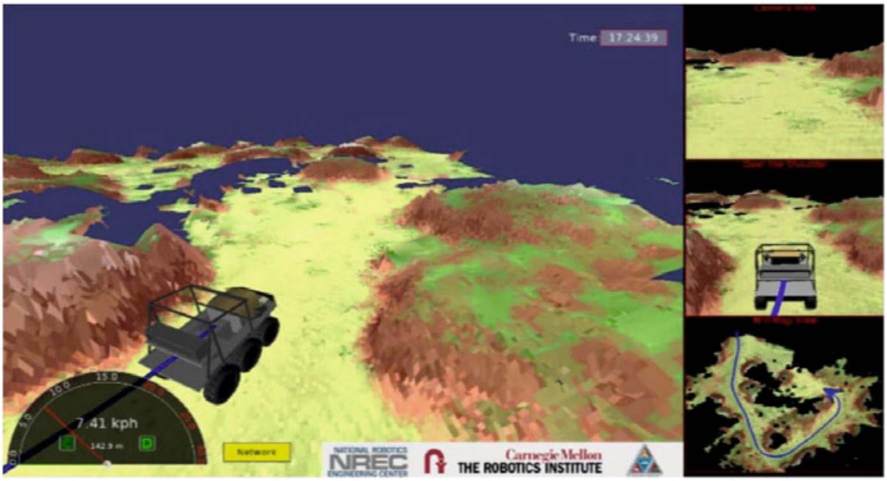


Fig. 9 Augmented Reality Display for Autonomy Assisted Hazard Avoidance. The photo-realistic display is augmented with false color obstacle annotations. Billboards are turned off.

5 User Study Results

The 3D Video system has been under continuous testing for the last two years including three week-long tests in the northeast and southwest of the US in both winter and summer conditions. In one test, a formal user study was conducted over a week in Pittsburgh in December of 2006. The details are described more fully in [9]. Five operators of different skill levels were tested on an obstacle course designed to elicit errors known to occur commonly in teleoperation. The participants averaged 20 years of automobile driving experience. Three subjects had prior experience teleoperating a live vehicle, including one with a 3D video system. One subject had never played a driving based video game. Course features included slaloms, decision gates, discrete obstacles, and loose and tight turns. The course was difficult enough to induce errors even when driving a vehicle manually from the driver's seat. Each operator drove the course in 4 different ways including sitting in the vehicle using standard controls, basic teleoperation with live video, and 3D video with and without latency. Each driving mode was assigned in random sequence to remove bias associated with learning the course. Driving performance was measured in terms of completion time, speed, and error rates.

Table 1 User Study Results

Metric	Live Video	3D Video with Latency	3D Video without Latency	Manual Drive
Completion Time (min)	9.3	7.2	6.4	2.2
Average Speed (m/s)	1.0	1.3	1.6	4.2
Errors	9.6	5.0	7.9	2.4

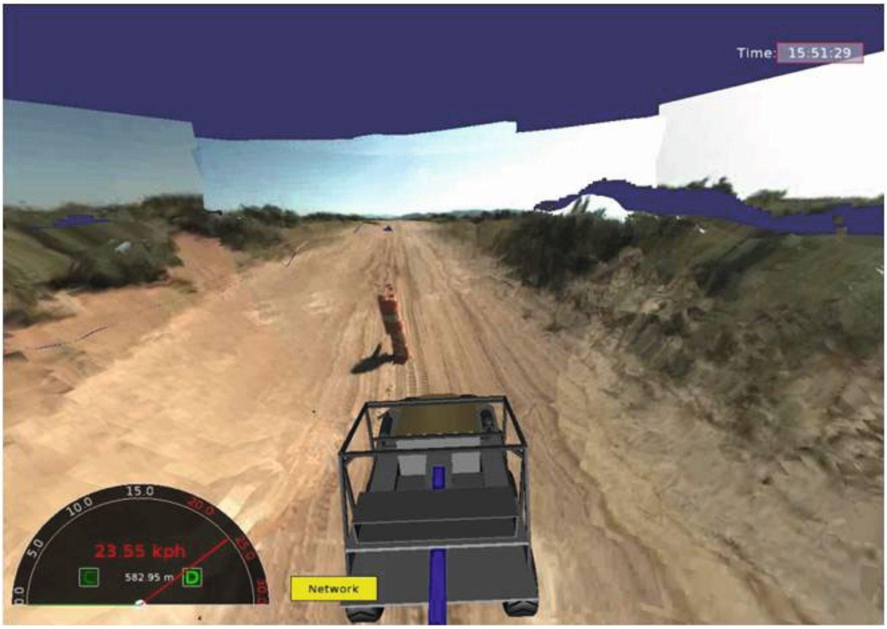


Fig. 10 High Speed Obstacle Avoidance. Latency compensation is most valuable during high speed driving. Here, the operator avoids an obstacle to the right, fitting the vehicle into a narrow space. A custom fly-behind view was used. The speedometer reads 24.55 km/hr. The operator control station is about 1 km further down the road.

In all cases, 3D video produced driving performance that was significantly (30% to 60%) better than standard teleoperation but not as good as manual driving. Furthermore, operators uniformly preferred 3D video to standard video teleoperation. A bug in the latency compensation process resulted in poor performance of this aspect in the test. However, the latency compensation techniques have subsequenctly proven to be very valuable in practice after this bug was removed. In a second test conducted over a one week period in south Texas in November of 2008, 50 novice operators drove the system (based on no training) in natural terrain using augmented reality waypoint guidance. There were no mishaps or incidents but we did notice that many operators with no teleoperation experience tended to forget that a real vehicle was being controlled. During this test, one of our skilled staff also tried to drive the system at high speed for a kilometer on dirt roads while avoiding discrete obstacles. Figure 10 indicates how we achieved unprecedented speeds in this test using our latency compensation while driving from up to a kilometer away from the real vehicle.

6 Conclusion

This paper has proposed a method to expend significant engineering effort in order to convert the task of robot teleoperation into a synthetic form of line-of-sight remote control. User studies have verified substantial gains in the effectiveness of the man-machine system. Along the way, we have produced improved solutions to problems like latency compensation and data compression for which there has been little hope of significant progress for some time. While many component technologies have made this possible, the most novel is photogeometric sensing applied to virtualized reality. Photogeometric sensing has the capacity to produce displays with both the photorealism of video and the interactivity of a video game. We expect that as sensors, rendering, guidance, and communications technologies continue to evolve, such displays and their derivatives will become a standard part of our toolbox. Technologies like flash lidar with bore-sighted video for ground vehicles will hopefully come on-line and reduce the engineering effort significantly. Even in the meantime, we find the effort is worthwhile in those very difficult applications for which robots and remote control are justified in the first place. Although we have not developed the idea here in detail, our 3D Video system is basically a projective texturing engine added to visualize colorized range data sets that were already being produced for the purposes of autonomy. The mental model used by both operator and robot is virtually identical in our system and this suggests many more derived advantages will be possible in contexts where autonomy shares more of the load and human and robot cooperate more fully.

Acknowledgements

This work was funded by the US Army Tank Automotive Research, Development, and Engineering Command (TARDEC). The colorized ranging sensor concept described here was based on a concept originally developed under funding from John Deere Corporation.

References

1. Chatila, R., Lacroix, S., Simion, T., Herrb, M.: Planetary exploration by a mobile robot: mission teleprogramming and autonomous navigation. Autonomous Robots (1995)
2. Chen, S.E., Williams, L.: View interpolation for image synthesis. In: Proc. SIGGRAPH 1993, pp. 279–288 (1993)

3. Fong, T., Thorpe, C., Baur, C.: Advanced Interfaces for Vehicle Teleoperation: Collaborative Control, Sensor Fusion Displays, and Remote Driving Tools. Autonomous Robots 11, 77–85 (2001)
4. Fong, T., Pangels, H., Wettergreen, D.: Operator Interfaces and Network-Based Participation for Dante II. In: SAE 25th International Conference on Environmental Systems, San Diego, CA (July 1995)
5. Hine, B., Stoker, C., Sims, M., Rasmussen, D., Hontalas, P.: The Application of Telepresence and Virtual Reality to Subsea Exploration. In: Proceedings ROV 1994, the 2nd Workshop on Mobile Robots for Subsea Environments, Monterey, CA (May 1994)
6. Hine, B., Hontalas, P., Fong, T., Piguet, L., Nygren, E., Kline, A.: VEVI: A Virtual Environment Teleoperations Interface for Planetary Exploration. In: SAE 25th International Conference on Environmental Systems, San Diego, CA (July 1995)
7. Kanade, T., Rander, P., Narayanan, P.J.: Virtualized Reality: Constructing Virtual Worlds from Real Scenes. In: IEEE MultiMedia, vol. 4(1). IEEE Computer Society Press, Los Alamitos (1997)
8. Kelly, A., Stentz, A., Amidi, O., Bode, M., Bradley, D., Diaz-Calderon, A., Happold, M., Herman, H., Mandelbaum, R., Pilarski, T., Rander, P., Thayer, S., Vallidis, N., Warner, R.: Toward Reliable Off-Road Autonomous Vehicles Operating in Challenging Environments. The International Journal of Robotics Research 25(5-6), 449–483 (2006)
9. Kelly, A., Anderson, D., Capstick, E., Herman, H., Rander, P.: Photogeometric Sensing for Mobile Robot Control and Visualization Tasks. In: Symposium on New Frontiers in Human-Robot Interaction, Edinburgh, Scotland, April 8-9 (2009)
10. Kim, W.S.: Virtual Reality Calibration and Preview / Predictive Displays for Telerobotics. Presence: Teleoperators and Virtual Environments 5, 2, 173–190 (1996)
11. Matusik, W., Beuhler, C., Raskar, R., Gortler, S., McMillan, L.: Image-Based Visual Hulls. In: SIGGRAPH 2000, pp. 369–374 (July 2000)
12. Messuri, D.A., Klein, C.A.: Automatic Body Regulation for Maintaining Stability of a Legged Vehicle during Terrain Locomotion. IEEE Journal of Robotics and Automation, RA 1, 132–141 (1985)
13. Milgram, P., Yin, S., Grodski, J.: An Augmented Reality Based Teleoperation Interface For Unstructured Environments. In: Proc. American Nuclear Society (ANS) 7th Topical Meeting on Robotics and Remote Systems Augusta, Georgia, USA, April 27-May 1, pp. 966–973 (1997)
14. Rohlf, J., Helman, J.: IRIS Performer: A high performance multiprocessing toolkit for real-time 3D graphics. In: Glassner, A. (ed.) Proceedings of SIGGRAPH 1994, pp. 381–394 (1994)
15. Segal, M., Korobkin, C., van Widenfelt, R., Foran, J., aeberli, P.: Fast shadows and lighting effects using texture mapping. In: Proceedings of SIGGRAPH 1992, pp. 249–252 (1992)
16. Sheridan, T.: Human Supervisory Control of Robot Systems. In: Proceedings of the 1986 IEEE International Conference on Robotics and Automation (April 1986)
17. Stewart, B., Ko, J., Fox, D., Konolige, K.: The Revisiting Problem in Mobile Robot Map Building: A Hierarchical Bayesian Approach. In: Proc. of the Conference on Uncertainty in Artificial Intelligence (2003)
18. Terrien, G., Fong, T., Thorpe, C., Baur, C.: Remote driving with a multisensor user interface. In: Proceedings of the SAE ICES, Toulouse, France (2000)

Robot Audition: Missing Feature Theory Approach and Active Audition

Hiroshi G. Okuno, Kazuhiro Nakadai, and Hyun-Don Kim

Hark, hark, I hear! The Tempest, Willian Shakespeare

Abstract. Robot capability of listening to several things at once by its own ears, that is, *robot audition*, is important in improving interaction and symbiosis between humans and robots. The critical issue in robot audition is real-time processing and robustness against noisy environments with high flexibility to support various kinds of robots and hardware configurations. This paper presents two important aspects of robot audition; Missing-Feature-Theory (MFT) approach and active audition. HARK open-source robot audition incorporates MFT approach to recognize speech signals that are localized and separated from a mixture of sound captured by 8-channel microphone array. HARK is ported to four robots, Honda ASIMO, SIG2, Robovie-R2 and HRP-2, with different microphone configurations and recognizes three simultaneous utterances with 1.9 sec latency. In binaural hearing, the most famous problem is a front-back confusion of sound sources. Active binaural robot audition implemented on SIG2 disambiguates the problem well by rotating its head with pitting. This active audition improves the localization for the periphery.

1 Robot Audition – Why, What and How?

Speech recognition plays an important role in communication and interaction, and people with normal hearing capabilities can listen to many kinds of sounds under various acoustic conditions. Robots should have hearing capability equivalent to humans to realize human-robot communication, when they are expected to help us

Horoshi G. Okuno · Hyun-Don Kim
Graduate School of Informatics, Kyoto University, Yoshida-Honmachi, Sakyo,
Kyoto 606-8501 Japan
e-mail: {okuno,hyundon}@kuis.kyoto-u.ac.jp

Kazuhiro Nakadai
Honda Research Institute Japan Co. Ltd. 8-1 Honmachi, Wako, Saitama 351-0188 Japana
and
Tokyo Institute of Technology
e-mail: nakadai@jp.honda-ri.com

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 227–244.
springerlink.com

in a daily environment. In daily environments, there exist a lot of noise sources including robot's own motor noises besides a target speech source. Many robot systems avoided this problem by forcing interaction parcitipants to wear a headset microphone [7]. For smoother and more natural interactions, a robot should listen to sounds by its own ears instead of using parcitipants' headset microphones.

"Robot Audition" research we have proposed [18] aims to realize recognition of noisy speech such as simultaneous speech by using robot-embedded microphones. As the robot audition research community is gradually growing, we have organized sessions on robot audition at IEEE/RSJ IROS 2004-2009 to promote robot audition research worldwide and special session on robot audition at IEEE-Signal Processing Society ICASSP-2009 to trigger cooperation between IEEE-RAS and -SPS.

Robot audition is expected to facilitate capabilities similar to those of human. For example, people can attend one conversation and switch to another even in a noisy environment. This capability is known as the *cocktail party effect*. For this purpose, a robot should separate a speech stream from a mixture of sounds. It may realize the hearing capability of "*Prince Shotoku*" that, according to the Japanese legend, could listen to 10 people's petitions at once.

Since a robot produces various sounds and should be able to "understand" many kinds of sounds, auditory scene analysis is the process of simulating useful intelligent behavior, and even required when objects are invisible. While traditionally, auditory research has been focusing on human speech understanding, understanding auditory scenes in general is receiving increasing attention. *Computational Auditory Scene Analysis (CASA)* studies a general framework of sound processing and understanding [30]. Its goal is to understand an arbitrary sound mixture including speech, non-speech signals, and music in various acoustic environments.

Two key ideas for CASA are 1) *the Missing Feature Theory (MFT) approach* [29] and 2) *active audition*. MFT approach treats each feature as either reliable or unreliable. Since noise or distortion still carries information, unreliable features may have some information. In Figure 1a), People can not see a letter "A." On the contrary, other information such as occlusion and noise helps the organization of fragments as is shown in Figure 1b). It is known that in the human auditory system noises that pad temporal gaps between sound fragments help auditory perception organization [8]. This is called "perceptual closure"in Gestalt psychology.

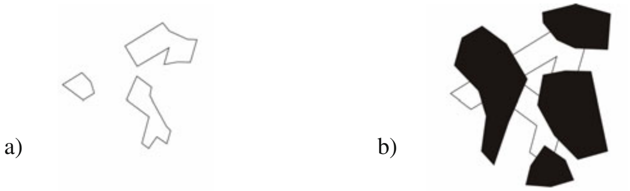


Fig. 1 Perceptual closure in Gestalt psychology. *Noises provide information on preception.* For the left part a), without noises, people cannot recognize the letter easily. Three fragments do not organize. For the right part b), with noises, people can do it easily. Noises help organization.

Active audition [18] is the auditory equivalent of acive vision. Similar to active vision, a robot may move its microphone or body to improve auditory perception. For binaural hearing, like human with two microphones, it is usually difficult to determine whether the sound source is ahead of or at the rear. This ambiguity is referred to as "front-rear" confusion. Suppose that the head moves to right. If the sound source moves to the same direction, it is behind. Otherwise, it is ahead of. The problem with active audition is motor noises caused by robot's own movements.

Three primitive functions for CASA are *sound source localization (SSL)*, *sound stream separation (SSS)*, and its recognition including *automatic speech recognition (ASR)*. Although robot audition share CASA's primitive functions as listed above, the critical requirements in robot audition are *real-time processing* and *robustness against diversity of acoustic environments*. These requirements are pursued in implementation on robots and their deployment to various acoustical environments.

Several groups have studied robot audition, in particular, SSL and SSS [11, 13, 16, 20, 34, 36, 38, 40]. Since they focused on *their own robot platform*, their systems are neither available nor portable for other research groups. Thus, researchers who want to incorporate robot audition in their robot had to make their own robot audition system from scratch. Valin released SSL and SSS software for robots called "ManyEars"[1] as GPL open-sourced software. This is the first software which can provide generally-applicable and customizable robot audition systems. ManyEars convers only SSL and SSS, but ASR is not included. Robot audition software should support ASR by integrating SSL and SSS, because ASR has a lot of parameters that affect the performance of a total robot audition system severely.

This paper describes how various modules are integrated into the whole robot audition system and presents a portable robot audition open-source software called "HARK"[2] (HRI-JP Audition for Robots with Kyoto University).

The rest of the paper is organized as follows. Section 2 describes the HARK open-source Robot Audition Software. Section 3 evaluates the performance of HARK. Section 4 describes the active binaural robot audition system. Section 5 presents the evaluation of active audition to disambiguate the front-rear confusion . Section 6 concludes the paper.

2 Open-Source Robot Audition Software HARK

HARK provides a complete module set for robot audition (see Fig 2). The modules are categorized into six categories as is shown in Table 1: multi-channel audio input, sound source localization and tracking, sound source separation, acoustic feature extraction for automatic speech recognition (ASR), automatic missing feature mask (MFM) generateion, and ASR interface. MFT based ASR (MFT-ASR) is also provided as a patched source code for a Japanese/English open source ASR,

[1] http://ManyEars.sourceforge.net/

[2] The word "hark" is an old English that stands for "listen." HARK of the current verison 0.1.17 is available at the following URL. HARK 1.0.0 will be released this mid-autumn . http://winnie.kuis.kyoto-u.ac.jp/HARK/

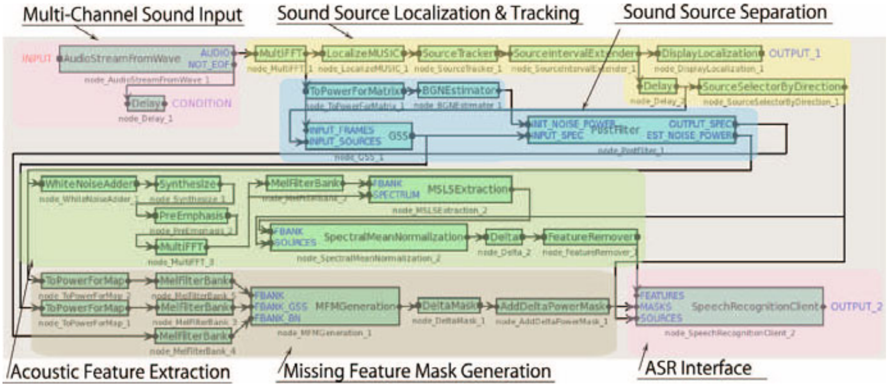


Fig. 2 An Instance of robot audition system using HARK 1.0.0 with FlowDesigner Interface. It comprises six main modules from multi-channel sound input to interface to Automatic Speech Recognition (ASR), which runs as a separate module.

Julius/Julian[3]. All modules except MFT-ASR run on FlowDesigner[4], because these modules and MFT-ASR do not share audio data. MFT-ASR runs separately through ASR interface over the network between PCs. HARK also provides a set of support modules under the category of data conversion and operation.

FlowDesigner is open-sourced and integrates modules by using shared objects, that is, function-call-based integration. It allows us to create reusable modules and to connect them together using a standardized mechanism to create a network of modules, as is shown in Figure 2. The modules are connected dynamically at run time. A network composed of multiple modules can be used in other networks. This mechanism makes the whole system easier to maintain since everything is grouped into smaller functional modules. If a program is created by connecting the modules, the user can execute the program from the GUI interface or from a shell terminal.

When two modules have matching interfaces, they are able to be connected regardless of their internal processes. One-to-many and many-to-many connections are also possible. A module is coded in programming language C++ and implemented as an inherited class of the fundamental module. Dynamic linking at run time is realized as a shared object in the case of Linux. Since data communication is done by using a pointer, it is much faster than socket communication. Therefore, FlowDesigner maintains a well-balanced trade-off between independence and processing speed. We have extended FlowDesigner to be more informable and robust against erros to use it as a programming environment for HARK.

In the remaining of this section, we explain main categories. Since a lot of audio signal processing methods and technologies have been developed for particular conditions under particular assumptions, HARK is designed to provide some of

[3] http://julius.sourceforge.jp

[4] http://flowdesigner.sourceforge.net/

Table 1 Modules provided by HARK 1.0.0
HARK 1.0.0 has six categories of FlowDesigner modules, one non-FlowDesigner module and one miscellaneous modules.

Category Name	Module Name
Multi-channel Audio I/O	AudioStreamFromMic
	AudioStreamFromWave
	SaveRawPCM
Sound Source Localization and Tracking	LocalizeMUSIC
	ConstantLocalization
	SourceTracker
	DisplayLocalization
	SaveSourceLocation
	LoadSourceLocation
	SourceIntervalExtender
Sound Source Separation	DSBeamformer
	GSS
	Postfilter
	BGNEstimator
Acoustic Feature Extraction	MelFilterBank
	MFCCExtraction
	MSLSExtraction
	SpectralMeanNormalization
	Delta
	FeatureRemover
	PreEmphasis
	SaveFeatures
Automatic Missing Feature Mask Generation	MFMGeneration
	DeltaMask
	DeltaPowerMask
ASR Interface	SpeechRecognitionClient
	SpeechRecognitionSMNClient
MFT-ASR	Multiband Julius/Julian (non-FlowDesigner module)
Data Conversion and Operation	MultiFFT
	Synthesize
	WhiteNoiseAdder
	ChannelSelector
	SourceSelectorByDirection
	SourceSelectorByID
	MatrixToMap
	PowerCalcForMap
	PowerCalcForMatrix

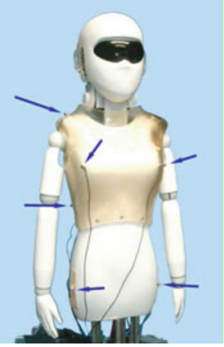


Fig. 3 SIG2 has 8 microphones on its body.

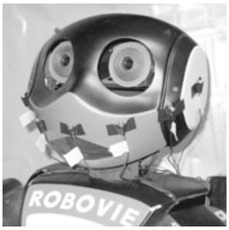


Fig. 4 Robovie R2 has 8 mics around its head.

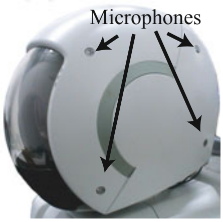


Fig. 5 ASIMO has two pair of 4 mics on each side.

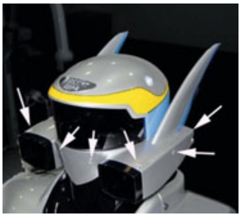


Fig. 6 HRP-2 has 8 microphones around the face.

promising methods for robot audition. Please note that there is no panacea for robot audition and the tradeoff between generality and performance is critical.

2.1 Microphone Configurations and Multi-channel Audio Input

The configurations of eight microphones for SIG2, Robovie R2, ASIMO and HRP-2 are shown in Figure 3–6, respectively.

HARK has two kinds of input methods, a microphone array and a file. The AudioStreamFromWave module reads audio signals from a file of WAVE format (RIFF waveform Audio Format). The AudioStreamFromMic module captures audio signals using a sound input device. It supports three types of devices: ALSA (Advanced Linux Sound Architecture) based sound cards[5], RASP series[6] and TD-BD-8CSUSB[7]. ALSA is an open source sound I/O driver for multi-channel sound cards such as RME Hammerfall. RASP series are multi-channel audio signal processing devices with CPUs produced by JEOL System Technology Co., Ltd. TD-BD-8CSUSB is an 8ch audio input systems with USB interface produced by Tokyo Electron Device Limited. SaveRawPCM saves audio data as a raw PCM file.

2.2 Sound Source Localization (SSL) and Tracking

HARK provides two kinds of SSL; MUSIC (MUltiple Signal Classification) [2] and fixed direction. MUSIC, an adaptive beamformer, localizes multiple sound sources robustly in real environments. The LocalizeMUSIC module calculates directions of arrival of sound sources from a multi-channel audio signal input, and outputs the number of sound sources and their directions by each time frame. ConstantLocalization outputs fixed sound directions without localization. It is mainly used either for performance evaluation and debugging or for interfacing visual localization.

SourceTracker tracks sound sources from the localization results, and it outputs sound source directions with source IDs. When the current direction and the previous direction derive from the same sound source, they are given the same source ID. DisplayLocalization is a viewer module for LocalizeMUSIC, ConstantLocalization, and SourceTracker. SaveSourceLocation stores localization results to a text file in every time frame. LoadSourceLocation loads source location information from the text file. SourceIntervalExtender extends tracking results forward to deal with start-point-misdetection of a sound source.

2.3 Sound Source Separation (SSS)

HARK provides two kinds of SSS, delay-and-sum beamformer and Geometric Source Separation (GSS). The DSBeamformer module separates sound sources by

[5] http://www.alsa-project.org/

[6] http://jeol-st.com/mt/2007/04/rasp2.html (in Japanese)

[7] http://www.inrevium.jp/eng/news.html

using sound source tracking results and a mixture of sound sources in the frequency domain. This uses a simple delay-and-sum beamformer. Thus, it is easy to control beamformer parameters, and it shows high robustness for environmental noises. GSS is a more sophisticated sound source separation module using a Geometric Source Separation (GSS) algorithm. GSS is a kind of hybrid algorithm of Blind Source Separation (BSS) [28] and beamforming. It relaxes BSS's limitations such as permutation and scaling problems by introducing *geometric constraints* obtained from the locations of microphones and sound sources.

Our implementation has two unique characteristics for robot audition; *suppressing robot's own noises* and *separation of moving talkers*. To deal with robot's noises such as fans, we specify a fixed noise direction in GSS. Then, GSS always removes the corresponding sound source as a robot's noise in spite of sound source localization results. In separating of moving talkers, a separation matrix in GSS should be initialized whenever the location of talker is changed, because the separation matrix is generated based on a geometrical relationship between a talker and each microphone. This kind of initialization sometimes causes slow convergence of the separation matrix. Thus, criteria and timing of separation matrix initialization can be specified at any time in our implementation. Currently, we are trying to add two more new features to GSS, that is, *adaptive stepsize control* that provides faster convergence of the separation matrix [22] and *Optima Controlled Recursive Average* [23] that controls window size adaptively to improve separation performance. We are testing them to confirm some promising results [24].

Postfilter enhances the output of GSS by using a spectral filter based on an optimal noise estimator described in [10]. We extend the original noise estimator so as to estimate both stationary and non-stationary noise by using multi-channel information [39]. Most post-filters address the reduction of a certain type of noise, that is, stationary background noise. Another advantage of Postfilter is parameter tuning. Since Postfilter consists of several signal processing techniques, there are a large number of parameters. Such parameters are mutually-dependent, and the best parameter setting depends on the surrounding acoustic environment and a situation. In HARK, most parameters are able to be controlled from FlowDesinger GUI/CUI. The best parameter setting is, then, obtained by using a parameter search system based on genetic algorithm.

The reason why we have separated GSS and post-filter is based on our experience with ManyEars' SeparGSS that unifies GSS and a multi-channel post-filter as a whole. We extend orignal SeparGSS by changing I/O IF to be able to use as a HARK module. However, the problem with SeparGSS is that the number of its controllable parameters is too small, when we encounter different acoustic situations caused by robot own noises, simultaneous speech recognition and moving talkers. In fact, a combination of GSS and Postfilter provides more flexible solution and better performance for a wide variety of acoustic situations.

BGNEstimator estimates the averaged background noise spectrum which is used for a noise reduction technique called *Minima Controlled Recursive Average (MCRA)* included in Postfilter.

2.4 Acoustic Feature Extraction for ASR

Acoustic feature extraction is an advantage of HARK because it is quite flexible to find the best acoustic feature. Everything can be done in a GUI environment although famous packages having feature extraction functions such as HTK[8] require text file editing. Most conventional ASR systems use *Mel-Frequency Cepstral Coefficient (MFCC)* as an acoustic feature. MelFilterBank consumes an input of a spectrum, analyzes it using Mel-frequency filter banks, and produces an output of a Mel-scale spectrum. MFCCExtraction extracts the MFCC acoustic feature from the Mel-scal spectrum.

HARK provides *Mel-Scale Log Spectrum (MSLS)* [26] as a primary acoustic feature instead of MFCC. Usually noises and distortions are concentrated in some areas in spectro-temporal space. MSLS can keep such locality, while MFCC spreads such contamination to all coefficients. MSLSExtraction consumes an input of the Mel-filter spectrum, performs liftering, and produces an output of MSLS features. SpectralMeanNormalization subtracts the averaged spectrum of the previous 5-sec utterance from the current utterance. It works well in real-time ASR systems to improve noise robustness of MSLS features. Finally, Delta consumes an output of the MSLS features, calculates the linear regression of each spectral coefficient, and produces an output of the MSLS features with delta features. FeatureRemover removes any coefficient (e.g. power) in an acoustic feature. PreEmphasis provides a low-pass filter to emphasize speech characteristics in the time or frequency domain according to user's preference. SaveFeatures stores a sequence of MSLS features into a file per utterance.

2.5 Automatic Missing Feature Mask Generation

HARK exploits the MFT approach for noise-robust ASR [5, 9]. MFT uses only *reliable* acoustic features in speech recognition and masks out unreliable parts caused by interfering sounds and preprocessing. MFT thus provides smooth integration between preprocessing and ASR. The mask used in MFT is called *missing feature mask (MFM)* represented as a spectro-temporal map. Three modules are prepared to generate MFM without using prior information.

The inputs of MFMGeneration are Mel-filtered spectra of the separated sound, the post-filtered sound, and the background noise estimated by the BGNEstimator. It estimates a leak noise by using the fact that the post-filtered sound is similar to clean speech while the separated sound includes a background and a leak noise. When the estimated leak noise is lower than a specified threshold, such a frequency bin is regarded as reliable, otherwise it is unreliable. MFMGeneration thus produces MFM as a binary mask. DeltaMask consumes an input of MFMs, and produces an output of delta MFMs. DeltaPowerMask generates a MFM for a delta power coefficient if an acoustic feature includes the coefficient. A set of examples of MFM are shown in Figure 7.

[8] http://htk.eng.cam.ac.uk/

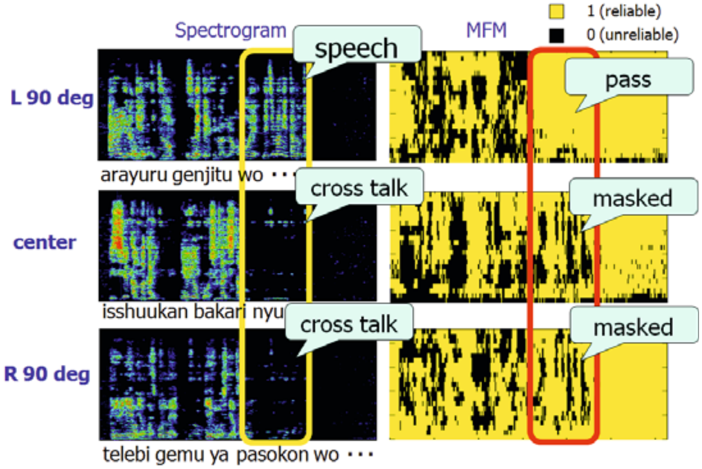


Fig. 7 Spectrograms and Missing Feature masks for Separated Sounds. HARK separates a mixture of three simultaneous utterances in Japanese. A pair of left and right columns depict the pectrogram and missing feature mask of each separated sound, respectively. The yellow and black parts of a mask indicates that the feature is reliable and unreliable, respectively. Since the leftmost talker speaks the longest, some crosstalks can be observed in the spectrograms of the center and rightmost talkers, but most of them are masked out by missing feature masks.

2.6 *ASR Interface for MFT-ASR with White Noise Addition*

SpeechRecognitionClient consumes inputs of acoustic features, MFMs and sound source directions. It sends them to Multiband Julius/Julian via TCP/IP communication. SpeechRecognitionSMNClient has almost the same function as SpeechRecognitionClient. The difference is that it supports an offline version of spectral mean normalization. SpeechRecognitionSMNClient uses a whole input signal to estimate the averaged spectrum, while SpectralMeanNormalization uses the previous 5-sec utterance. Thus, a robot audition system with SpectralMeanNormalization is used in offline systems.

Missing Feature Theory (MFT) approaches use MFM of reliability to improve ASR. The estimation process of output probability in the decoder is modified in MFT-ASR. We adopted a classifier-modification method with marginalization, because other approaches such as cluster-based reconstruction of feature-vector imputation is not robust against mel-frequency based features [29]. Unreliable acoustic features caused by errors in preprocessing are masked using MFMs, and only reliable ones are used for a likelihood calculation in the ASR decoder. As such an MFT-ASR, we use “Multi-band Julis/Julian” [26], an extension of original Julius/Julian that supports various types of HMMs such as shared-state triphones and tied-mixture models. Both statistical language model and network grammar are supported. In decoding, an ordered word bi-gram is used in the first pass, and a reverse ordered word tri-gram in the second pass. It works as a standalone or client-server application. To run as a server, we modified the system to be able to communicate acoustic features and MFM via a network.

HARK exploits two ways of perceptual closure; *MFT-ASR* and *white noise addition*. By the latter, HARK tries to recover distortion in any frequency band by adding a white noise, a kind of broad-band noise, to separated speech signals. This idea is motivated by the psychological evidence that noise may help human perception, which is known as *auditory induction.* The WhiteNoiseAdder of Data Conversion and Operation Category adds white noise basically in order to relax distortion problems caused by the Postfilter.

3 Evaluation of HARK

We evaluated the robot audition system in terms of the following three points; (1) recognition performance of simultaneous speech, (2) improvement of ASR by localization, and (3) processing speed. We also presents two applications of the HARK.

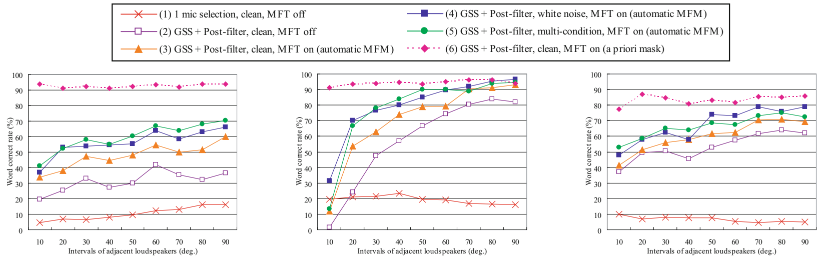
3.1 Recognition of Simultaneous Speech Signals

To evaluate how MFT and white noise addition improve the performance of ASR, we conducted isolated word recognition of three simultaneous speech. In this experiment, Humanoid SIG2 with an 8-ch microphone array was used in a 4 m × 5 m room. Its reverberation time (RT_{20}) was 0.3–0.4 seconds.

Three simultaneous speech for test data were recorded with the 8-ch microphone array of SIG2 by using three loudspeakers (Genelec 1029A). The distance between each loudspeaker and the center of the robot was 2 m. One loudspeaker was fixed to the front (center) direction of the robot. The locations of left and right loudspeakers from the center loudspeaker varied from ±10° to ±90° at the intervals of 10°. ATR phonemically-balanced word-sets were used as a speech dataset. A female (f101), a male (m101) and another male (m102) speech sources were used for the left, center and right loudspeakers, respectively. Three words for simultaneous speech were selected at random. In this recording, the power of robot was turned off.

The recognition performance of three simultaneous talkers is evaluated with the following six conditions:

(1) The raw input captured by the left-front microphone was recognized with the clean acoustic model trained with 10 male and 12 female ATR phonemically-balanced word-sets excluding the three word-sets (f101, m101, and m102).
(2) The sounds separated by SSS were recognized with the clean acoustic model.
(3) The sounds separated by SSS were recognized with MFM generated automatically and the clean acoustic model.
(4) The sounds separated by SSS were recognized with automatically generated MFM and the **WNA acoustic model** trained with the same ATR wordsets, and the clean speech to which white noise was added by 40 dB of peak power.



(a) The left talker (b) The center talker (c) The right talker

Fig. 8 Word correct rates of three simultaneous talkers with our system

(5) The sounds separated by SSS were recognized with automatically generated MFM and the **MCT acoustic model** trained with the same ATR word-sets and separated speech datasets.
(6) The sounds separated by SSS were recognized with ***a priori* MFM** generated manually and the clean acoustic model . Since this mask is *ideal*, its result may indicate the potential upper limit of HARK.

Each acoustic models was trained as 3-state and 4-mixture triphone HMM, because 4-mixture HMM had the best performance among 1, 2, 4, 8, and 16-mixture HMMs.

The results were summarized in Figure 8. MFT-ASR with Automatic MFM Generation outperformed the normal ASR. The **MCT acoustic model** was the best for MFT-ASR, but the **WNA acoustic model** performed almost the same. Since the **WNA acoustic model** does not require prior training, it is the most appropriate one for robot audition. The performance at the interval of 10° was poor in particular for the center talker, because any current sound source separation methods fails in separating such close three talkers. The fact that *A priori* mask showed quite a high performance may suggest possibilities to improve automatic MFM generation.

3.2 *Sound Source Localization Effects and Processing Speed*

This section evaluates how the quality of sound source localization methods including manually given localization, steered Beamformer and MUSIC affects the performance of ASR. SIG2 used steered BF. Since the performance MUSIC depends on the number of microphones on the same plane, we used Honda ASIMO shown in Figure 5, which was installed in a 7 m × 4 m room. Its three walls were covered with sound absorbing materials, while the other wall was made of glass which makes strong echoes. The reverberation time (RT_{20}) of the room is about 0.2 seconds. We used the condition (4), and used three methods of sound source localization with clean and WNA acoustic models.

The results of word correct rates were summarized in Table 2. With the **clean acoustic model**, MUSIC outperformed steered BF, while with the **WNA acoustic mode**, both the performances were comparable. In case of **given localization**,

Table 2 Word correct rate of the center talker (in %) according to each localization method

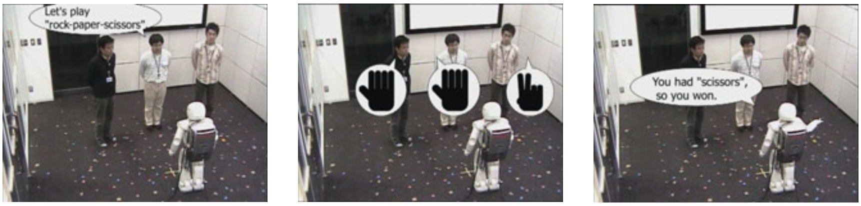
Acoustic model	White noise addition			Clean model		
direction \ *Interval*	*30°*	*60°*	*90°*	*30°*	*60°*	*90°*
manually given	90.0	88.5	91.0	85.0	84.5	87.0
steered BF	82.3	90.5	89.0	65.5	70.6	72.4
MUSIC	86.0	83.3	86.7	57.0	74.0	64.5

improvement by white noise addition training was small. On the other hand, training with white noise addition improved word correct rates greatly for both steered beamformer and MUSIC. We think that the ambiguity in sound source localization caused voice activity detection to be more ambiguous, which degraded recognition performance with the clean acoustic model. On the other hand, white noise addition to separated sounds with the WNA acoustic model reduced such degradation.

The processing time when HARK separated and recognized speech signals of 800 seconds on a Pentium 4 2.4 GHz CPU is 499 second consisting of 369 sec for FlowDesigner and 130 sec for MFT-ASR. The output delay is 0.446 second. As a whole, our robot audition system ran in real time.

3.3 Listen to Three Simultaneous Talkers

One application is a referee for a rock-paper-scissors sound game that includes a recognition task of two or three simultaneous utterances. ASIMO was located at the center of the room, and three talkers stood 1.5 m away from ASIMO at 30° intervals (Figure 9). A speech dialog system specialized for this task was connected with the HARK. ASIMO judged who won the game by using only speech information, i.e., without using visual information. Because they said rock, paper, or scissors simultaneously in an environment where robot noises exist, the SNR input sound was less than -3 dB. All of the three utterances had to be recognized successfully to


Let's play "rock-paper-scissors"
You had "scissors", so you won.

a) U2:Let's play an RPS game. b) U1, U2: paper, U3: scissors c) A: U3 won.

Fig. 9 Snapshots of rock-paper-scissors (RPS) game (A: ASIMO, U1:left user, U2:center user, U3: right user). ASIMO plays a referee. Three men play a RPS game by uttering one of rock, paper, or scissors. Then ASIMO separates and recognizes each utterance to determine who wins or draw.

complete the task. The completing rate of referee task is around 60% and 80% in the cases of three and two talkers, respectively.

Another application is that Robovie accepts simultaneous meal orders that three actual human talkers place (Figure 10). The HARK recognizes each meal order and confirms their orders one by one and tells the total amount of the orders. The FlowDesigner implementation reduces the response time from 8.0 sec to 1.9 sec. If the same input is given by an audio file, the response time is about 0.4 sec.

May I take your order?
I'll have Katsudon, please.
I'd like a set meal with Tonkatsu, please.
Full course French dinner.
Starting from the left, that's one Katsudon, one set meal with Tonkatsu, and one full course French.

a) Robovie: May I take your orders?b) Each one places an order. c) Robovie rephrases each order.

Fig. 10 Snapshots of Meal Orders (Roboview and three talkers). Robovie asks each meal order. Three men placed each meal order at once. Robovie separates and recognizes each utterance for 1.9 msec. Then it rephrases each meal order and tell their total price.

4 Active Binaural Robot Audition

Human and most animals have only a pair of ears, that is, binaual hearing, but they can do functions of CASA. We attribute their competence of binaural listening and hearing to their body, in particular, active audition. For example, if the head with ears moves, the number of microphones virtually increases. If the head approaches to a sound source, the resolution of SSL may improve. Nakadai et al. proposed active audition in 2000 [18] and a small number of research groups are engaged in active audition research [4, 14, 19, 27].

Another merit of binaural audition is the availability of stereo input devices. Stereo AD devices are inexpensive, while multi-channel AD devices are very expensive. In addition, every PC has a stereo input device as a stardard equipment. Thus, portable open-source binaural robot audition is expected to open a new market of either robot audition or auditory capability of electronic appliances.

Motional theories have been investigated for human's active audition [6]. According to the thorough investigation undertaken by Thurlow, Mangels, and Runge (1967), *rotating* is the largest movements among three main ones, rotating (yawing), tipping (pitching) and pivoting (rolling) movements when the subjects are blindfolded in an anechoic chamber. The most frequent combination of movements involves *rotating and tipping movements*.

People usually rotate their heads at 42° ±20.4° and the most frequent combination of classes of movements involves rotating and tipping movements (0.5∼1 kHz) so that they determine the accurate direction of sound sources. Consequently, we designed our system that can distinguish between sounds from the front and from

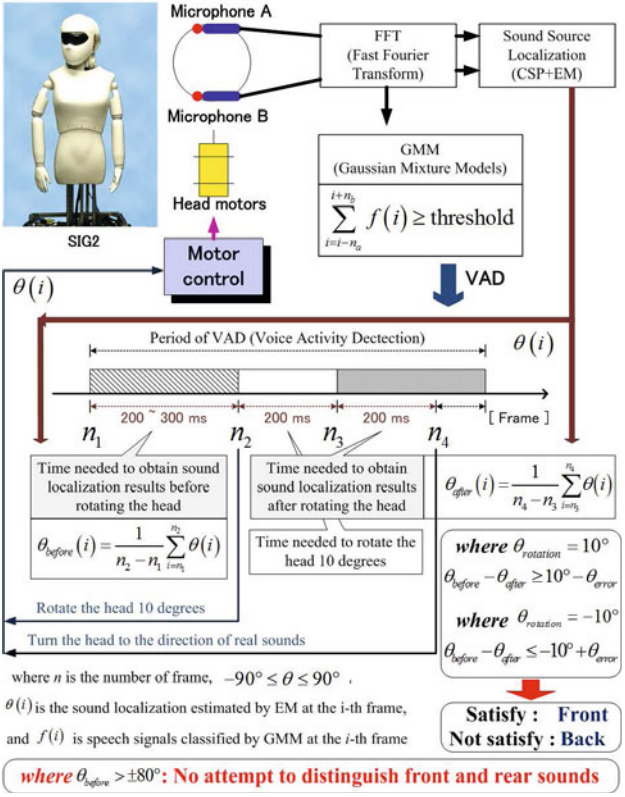

Microphone A
Microphone B
FFT (Fast Fourier Transform)
Sound Source Localization (CSP+EM)
GMM (Gaussian Mixture Models)
$\sum_{i=i-n_a}^{i+n_b} f(i) \geq$ threshold
VAD
Head motors
Motor control
SIG2
$\theta(i)$
Period of VAD (Voice Activity Dectection)
$\theta(i)$
200 ~ 300 ms
200 ms
200 ms
[Frame]
n_1
n_2
n_3
n_4
Time needed to obtain sound localization results before rotating the head
$\theta_{before}(i) = \frac{1}{n_2 - n_1} \sum_{i=n_1}^{n_2} \theta(i)$
Time needed to obtain sound localization results after rotating the head
Time needed to rotate the head 10 degrees
$\theta_{after}(i) = \frac{1}{n_4 - n_3} \sum_{i=n_3}^{n_4} \theta(i)$
where $\theta_{rotation} = 10°$
$\theta_{before} - \theta_{after} \geq 10° - \theta_{error}$
where $\theta_{rotation} = -10°$
$\theta_{before} - \theta_{after} \leq -10° + \theta_{error}$
Rotate the head 10 degrees
Turn the head to the direction of real sounds
where n is the number of frame, $-90° \leq \theta \leq 90°$,
$\theta(i)$ is the sound localization estimated by EM at the i-th frame,
and $f(i)$ is speech signals classified by GMM at the i-th frame
Satisfy : Front
Not satisfy : Back
where $\theta_{before} > \pm 80°$: No attempt to distinguish front and rear sounds

Fig. 11 System overview of localizing sounds in a horizontal space. The robot detects speech signals classified by Gaussian mixture model (GMM). Then Voice activity detection (VAD) descriminates speech signals from noises to calculates the average of sound localization events. Now the robot rotates its head by 10° in the direction of the detected signals. After rotation, it obtains the average of sound localization events. Finally, it repeats the above procedures to get the right direction. Another head movement strategy is to do tipping by 10° before rotating.

the rear by simply rotating its head at least 10°. The reason that it can distinguish front from rear sources by rotating 10° is that the error margin for a single moving sound slower than 1 rad/sec is about 10°. Figure 11 depicts the process of localizing sounds over the entire azimuth range, or in a horizontal plane, for a humanoid robot.

First, we adopt cross-power spectrum phase (CSP) analysis [12, 25] for SSL, because it can calculate the Time Delay of Arrival (TDOA) between two microphones without impulse response data. In addition, to cope with the ambiguity and sparseness of acquired information picked up by the two microphones, we applied an expectation-maximization (EM) algorithm [17] to localizing several sound sources and reduce localization errors. The assumption for this SSL is that there exists at most one predominant frequency element at each time frame.

After classifying speech signals according to their TD0A, motion planning is performed. Since motion is executed during the periods of speech signals, we develop voice activity detection (VAD) [15] based on Gaussian mixture model (GMM). Gaussian mixture model (GMM) is a powerful statistical method widely used for speech classification [3, 32]. We applied the 0 to 12th coefficients (total 13 values) and the $\delta 1$ to $\delta 12$th coefficients (total 12 values) of Mel frequency cepstral coefficients (MFCCs) [32].

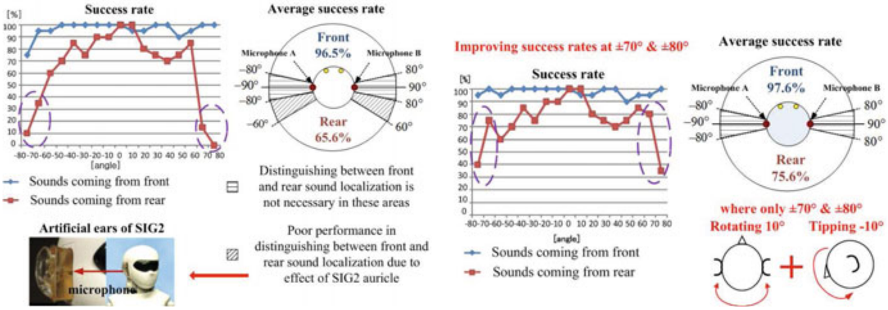
Finally, robots need to improve their cognition abilities (active audition) concerning changing location of sounds while they are in motion. For example, robots should be able to distinguish whether sound signals are coming from the front or the rear if they rotate or move only two microphones placed in the robot's head or body. To solve front-rear confusion in binaural sound source localization, we detect the change in sound localization by slightly rotating and/or tipping the two microphones. Since such motions have the effect of increasing the number of microphones virtually, a binaural audition system can estimate sound localization over the entire azimuth range.

We also develop another motion strategy by rotating and tipping instead of rotating only according to the observation of human perception mentioned above.

5 Evaluation of Binaural Active Robot Audition

Figure 12a) shows the results of success rate of localization by active audition specified by Figure 11. The sound source repeats a 0.75 sec-utterance of "sig" for 20 times. The loudspeaker is at the distance of 1.5 m from the SIG2 humanoid. Utterances are replayed at 85 dB, while the background noise is about 55 dB. The speed of head movement is 2.5 rd/sec.

In the upper left part of Figure 12a), the success rate for sound localization in the front area is 96.5%, while the one in the rear area is 65.6% due to the sound



a) Rotating only

b)Rotating with Tipping

Fig. 12 Results of localizing sounds in a horizontal space. The success rate of sound source localization by combined head movement of rotating and tipping outperforms thaat of rotating only.

diffraction causedd by the artificial auricle used in SIG2 (see the left bottom of Figure 12a)). To avoid this degrading performance in the rear area, active source localization by combining rotating with tipping is developed. This movement is motivated by how blind people localize the sound. Their most frequent combination of classes of movements involves rotating and tipping movements. The success rate for sound localization by combining rotating with tipping movement improves about 10 points at only 70° and 80° in the rear area as shown in Figure 12b).

6 Conclusion

This paper described the design and application of HARK open-source robot audition software. We are currently at the final testing phase of HARK 1.0.0. Some of new features under development include semi-blind source separation for barge-in interactions, that is, the user can speak at any time by interrupting system's utterances, and extension of SSL and SSS for moving talkers. The HARK is also applied to musical robots that can listen to music by their own ears. The real-time beat tracking will be incorporated into the HARK as a FlowDesigner module.

Binaural active robot audition is now studied under the Strategic Japanese-French Cooperative Program on "Information and Communcations Technology Including Computer Sciences" from this August to March, 2012. The project will investigate binaural active robot audition from deives, software to cognitve science, physiology and brain sciences.

Acknowledgements. This research was partially supported by JSPS Grant-in-Aid for Scientific Research (S), and Global COE program of MEXT, Japan. We thank to Mr. Hiroshi Tsujino of Honda Research Institute Japan Co. Ltd. Profs. Tetsuya Ogata, Kazunori Komatani and Toru Takahashi of Kyoto University and students of our lab at Kyoto University for their fruitful discussion.

References

1. Aloimonos, Y., Weiss, I., Bandyopadhyay, A.: Active vision. Intern'l J. of Computer Vision 1(4), 333–356 (1999)
2. Asano, F., Asoh, H., Matsui, T.: Sound source localization and signal separation for office robot "Jijo-2". In: Proc. of IEEE Intern'l Conf. on Multisensor Fusion and Integration for Intelligent Systems, pp. 243–248 (1999)
3. Bahoura, M., Pelletier, C.: Respiratory Sound Classification using Cepstral Analysis and Gaussian Mixture Models. In: IEEE/EMBS Intern'l Conf., San Francisco, USA (2004)
4. Berglund, E.J.: Active Audition for Robots using Parameter-Less Self-Organising Maps. Ph. D Thesis, The University of Queensland, Australia (2005)
5. Barker, J., Cooke, M., Green, P.: Robust ASR Based on Clean Speech Models: An Evaluation of Missing Data Techniques for Connected Digit Recognition in Noise. In: 7th European Conference on Speech Communication Technology, pp. 213–216 (2001)
6. Blauert, J.: Spatial Hearing – The Psychophysics of Human Sound Localization. The MIT Press, Cambridge (1996) (revised edition)

7. Breazeal, C.: Emotive Qualities in Robot Speech. In: Proceeding of IEEE/RSJ Intern'l Conf. on Intelligent Robots and Systems, Hawaii, pp. 1389–1394 (2001)
8. Bregman, A.S.: Auditory Scene Analysis. The MIT Press, Cambridge (1990)
9. Cooke, M., Green, P., Josifovski, L., Vizinho, A.: Robust Automatic Speech Recognition with Missing and Unreliable Acoustic Data. Speech Communication, vol. 34, pp. 267–285. Elsevier, Amsterdam (2001)
10. Ephraim, Y., Malah, D.: Speech enhancement using minimum mean-square error short-time spectral amplitude estimator. IEEE Trans. on ASSP 32(6), 1109–1121 (1984)
11. Hara, I., Asano, F., Kawai, Y., Kanehiro, F., Yamamoto, K.: Robust speech interface based on audio and video information fusion for humanoid HRP-2. In: Proceeding of IEEE/RSJ Intern'l Conf. on Intelligent Robots and Systems, Sendai, Japan, pp. 2404–2410 (2004)
12. Kim, H.D., Komatani, K., Ogata, T., Okuno, H.G.: Real-Time Auditory and Visual Talker Tracking through integrating EM algorithm and Particle Filter. In: Okuno, H.G., Ali, M. (eds.) IEA/AIE 2007. LNCS (LNAI), vol. 4570, pp. 280–290. Springer, Heidelberg (2007)
13. Kim, H.D., Komatani, K., Ogata, T., Okuno, H.G.: Human Tracking System Integrating Sound and Face Localization using EM Algorithm in Real Environments. Advanced Robotics, 23(6):629–653 (2009) doi: 10.1163/156855309X431659
14. Kim, H.D., Komatani, K., Ogata, T., Okuno, H.G.: Binaural Active Audition for Humanoid Robots to Localize Speech over Entire Azimuth Range. Applied Bionic and Biomechanics (2009) (to appear)
15. Lu, L., Zhang, H.G., Jiang, H.: Content Analysis for Audio Classification and Segmentation. IEEE Trans. on Speech and Audio Processing 10(7), 504–516 (2002)
16. Michaud, F., et al.: Robust Recognition of Simultaneous Speech by a Mobile Robot. IEEE Trans. on Robotics 23(4), 742–752 (2007)
17. Moon, T.K.: The Expectation-Maximization algorithm. IEEE Signal Processing Magazine 13(6), 47–60 (1996)
18. Nakadai, K., et al.: Active audition for humanoid. In: Proc. of 17th National Conference on Artificial Intelligence, pp. 832–839. AAAI, Menlo Park (2000)
19. Nakadai, K., Hidai, K., Okuno, H.G., Kitano, H.: Real-Time Speaker Localization and Speech Separation by Audio-Visual Integration. In: Proc. of IEEE-RAS Intern'l Conf. on Robotics and Automation, May 2002, pp. 1043–1049 (2002), doi:10.1109/ROBOT.2002.1013493
20. Nakadai, K., Matasuura, D., Okuno, H.G., Tsujino, H.: Improvement of recognition of simultaneous speech signals using AV integration and scattering theory for humanoid robots. Speech Communication 44(4), 97–112 (2004), doi:10.1016/j.specom.2004.10.010
21. Nakadai, K., Okuno, H.G.: An Open Source Software System for Robot Audition HARK and Its Evaluation. In: IEEE/RAS Intern'l Conf. on Humanoid Robots, pp. 561–566 (2008), doi:10.1109/ICHR.2008.4756031
22. Nakajima, H., Nakadai, K., Hasegawa, Y., Tsujino, H.: Adaptive Step-Size Parameter Control for Real-World Blind Source Separation. In: IEEE Intern'l Conf. on Acoustics, Speech and Signal Processing, pp. 149–152 (2008)
23. Nakajima, H., Nakadai, K., Hasegawa, Y., Tsujino, H.: High performance sound source separation adaptable to environmental changes for robot audition. In: IEEE/RSJ Intern'l Conf. on Intelligent Robots and Systems, pp.2165–2171 (2008)
24. Nakajima, H., Nakadai, K., Hasegawa, Y., Tsujino, H.: Sound source separation of moving speakers for robot audition. In: IEEE Intern'l Conf. on Acoustics, Speech and Signal Processing, pp. 3685–3688 (2009)

25. Nishiura, T., Yamada, T., Nakamura, S., Shikano, K.: Localization of multiple sound sources based on a CSP analysis with a microphone array. In: Proceeding of IEEE Intern'l Conf. on Acoustics, Speech and Signal Processing, Istanbul, Turkey, pp. 1053–1056 (2000)
26. Nishimura, Y., Shinozaki, T., Iwano, K., Furui, S.: Noise-robust speech recognition using multi-band spectral features. In: 148th ASA Meeting, 1aSC7, ASA (2004)
27. Okuno, H.G., Nakadai, K., Hidai, K., Mizoguchi, H., Kitano, H.: Human-Robot Non-Verbal Interaction Empowered by Real-Time Auditory and Visual Multiple-Talker Tracking. Advanced Robotics 17(2), 115–130 (2003), VSP and RSJ, doi:10.1163/156855303321165088
28. Parra, L.C., Alvino, C.V.: Geometric source separation: Mergin convolutive source separation with geometric beamforming. IEEE Trans. on SAP 10(6), 352–362 (2002)
29. Raj, H., Sterm, R.M.: Missing-feature approaches in speech recognition. IEEE Signal Processing Magazine 22(5),101–116 (2005)
30. Rosenthal, D., Okuno, H.G.: Computational Auditory Scene Analysis. Lawrence Erlbaum Associates, Mahwah, New Jersey (1998)
31. Schmidt, R.O.: Multiple Emitter Location and Signals Parameter Estimation. IEEE Transactions on Antennas and Propagation, AP-34, 276–280 (1986)
32. Shah, J.K., Iyer, A.N., Smolenski, B.Y., Yantormo, R.E.: Robust Voiced/Unvoiced classification using novel feature and Gaussian Mixture Model. In: Proc. of IEEE Intern'l Conf. on Acoustics, Speech, and Signal Processing, Montreal, Canada (2004)
33. Valin, J.-M., Michaud, F., Hadjou, B., Rouat, J.: Localization of simultaneous moving sound sources for mobile robot using a frequency-domain steered beamformer approach. In: IEEE Intern'l Conf. on Robotics and Automation, pp. 1033–1038 (2004)
34. Valin, J.-M., Michaud, F., Hadjou, B., Rouat, J.: Enhanced Robot Audition Based on Microphone Array Source Separation with Post-Filter. In: IEEE/RSJ Intern'l Conf. on Intelligent Robots and Systems, pp.2123–2128 (2004)
35. Valin, J.-M., Michaud, F., Hadjou, B., Rouat, J., Nakadai, K., Okuno, H.G.: Robust Recognition of Simultaneous Speech by a Mobile Robot. IEEE Transactions on Robotics 23(4), 742–752 (2007), doi:10.1109/TRO.2007.900612
36. Valin, J.-M., Michaud, F., Hadjou, B., Rouat, J., Nakadai, K., Okuno, H.G.: Robust localization and tracking of simultaneous moving sound sources using beamforming and particle filtering. Robotics and Autonomous Systems J. 55(3), 216–228 (2007)
37. Yamada, S., Lee, A., Saruwatari, H., Shikano, K.: Unsupervided speaker adaptation based on HMM sufficient statistics in various noisy environments. In: Proc. of Eurospeech 2003. ESCA, pp. 1493–1496 (2003)
38. Yamamoto, S., Valin, J.-M., Nakadai, K., Ogata, T., Okuno, H.G.: Enhanced Robot Speech Recognition Based on Microphone Array Source Separation and Missing Feature Theory. In: IEEE-RAS Intern'l Conf. on Robotics and Automation, pp. 1477–1482 (April 2005)
39. Yamamoto, S., et al.: Making A Robot Recognize Three Simultaneous Sentences in Real-Time. In: IEEE/RSJ Intern'l Conf. on Intelligent Robots and Systems, pp. 4040–4045 (2005)
40. Yamamoto, S., et al.: Real-time robot audition system that recognizes simultaneous speech in the real world. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5333–5338 (2006)

Haptography: Capturing and Recreating the Rich Feel of Real Surfaces

Katherine J. Kuchenbecker, Joseph Romano, and William McMahan

Abstract. Haptic interfaces, which allow a user to touch virtual and remote environments through a hand-held tool, have opened up exciting new possibilities for applications such as computer-aided design and robot-assisted surgery. Unfortunately, the haptic renderings produced by these systems seldom feel like authentic re-creations of the richly varied surfaces one encounters in the real world. We have thus envisioned the new approach of haptography, or haptic photography, in which an individual quickly records a physical interaction with a real surface and then recreates that experience for a user at a different time and/or place. This paper presents an overview of the goals and methods of haptography, emphasizing the importance of accurately capturing and recreating the high frequency accelerations that occur during tool-mediated interactions. In the capturing domain, we introduce a new texture modeling and synthesis method based on linear prediction applied to acceleration signals recorded from real tool interactions. For recreating, we show a new haptography handle prototype that enables the user of a Phantom Omni to feel fine surface features and textures.

1 Introduction

When you touch objects in your surroundings, you feel a rich array of haptic cues that reveal each object's geometry, material, and surface properties. For example, the vibrations and forces experienced by your hand as you stroke a piece of fabric or write on a sheet of corrugated cardboard are easily identifiable and distinct from those generated by gripping a foam ball or tapping on a hollow bronze sculpture. Humans excel at eliciting and interpreting haptic feedback during such interactions, naturally leveraging this wealth of information to guide their actions in the physical world (Klatzky and Lederman, 2003).

Motivated by the richness and usefulness of natural haptic feedback, we have envisioned the new approach of **haptography**. Like photography in the visual

Katherine J. Kuchenbecker · Joseph Romano · William McMahan
Haptics Group, GRASP Lab, University of Pennsylvania, Philadelphia, PA, USA
e-mail: {kuchenbe,jrom,wmcmahan}@seas.upenn.edu
website: http://haptics.grasp.upenn.edu

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 245–260.
springerlink.com

domain, haptography enables an individual to quickly record the feel of an interesting object and reproduce it at another time and/or place for someone to interact with as though it was real. The idea for haptography was first articulated by Kuchenbecker in 2008, and this paper provides an overview of its goals and methods. Haptographic technology involves highly sensorized handheld tools, haptic signal processing for model synthesis, and uniquely actuated haptic interfaces, all focused on capturing and recreating the rich feel of real surfaces.

Once these capabilities are available, a wide variety of practical applications will benefit from haptography. For example, it will provide a fast, simple way to store the current feel of a physical object (such as a unique marble statue or a dental patient's tooth), compare it with a database of other recordings, and analyze surface changes over time. Haptographs will also allow a wide range of people to touch realistic virtual copies of objects that are not directly accessible, such as archaeological artifacts and merchandise being sold online. Furthermore, haptography has the potential to significantly increase the realism of medical simulators and video games by incorporating object models built from quantitative contact data captured during real interactions. Beyond virtual environments, haptography can have a beneficial impact on teleoperation, where the operator uses a haptic interface to control the movement of a remote robot and wants to feel the objects being manipulated as though they were locally present. Finally, the haptographic focus on recording, analyzing, and recreating everything felt by the human hand will probably yield new insights on the sense of touch, which may help robotic hands achieve human-like dexterity and sensitivity in interactions with real physical objects.

Enabling the art and science of haptography requires us to answer two main questions: *How can we characterize and mathematically model the feel of real surfaces?* and *How can we best duplicate the feel of a real surface with a haptic interface?* Building on knowledge of the human haptic sensory system, haptography research uses measurement-based mathematical modeling to derive perceptually relevant haptic surface models and dynamically robust haptic display methods. The following sections of this paper explain the envisioned system paradigm, our initial work on capturing the feel of surfaces, and our continuing work on recreating such surfaces realistically.

2 Overview of Haptography

Despite its ubiquitous importance in human life, we currently lack a formal method for analyzing and understanding the feel of touch-based interaction with physical objects. Furthermore, fundamental surface modeling and device design choices prevent the vast majority of existing haptic interfaces from compellingly duplicating the feel of real objects.

Target Interactions. Direct-touch haptography would enable an individual to capture natural interactions between their fingertip and an interesting surface and then recreate that exact feel with a programmable tactile interface that can be freely explored. While fascinating and useful, there are currently many technological

Table 1 The user experience of haptography can be understood via an analogy to photography.

Photography	Haptography
digital SLR camera	highly sensorized handheld tool
interchangeable lenses	interchangeable tool tips
framing a shot and focusing the camera	exploring an object's surface
planar distribution of light intensities	stream of positions, forces, and accelerations
optics and the human eye	haptics and the human hand
LCD monitor	uniquely actuated handheld tool
viewing the digital image	freely exploring the digital model
spatial resolution and focus	high frequency accelerations

challenges that preclude the realization of such an ambitious objective; it will take many years for today's most promising noninvasive tactile sensors, e.g., (Sun et al, 2007), and high resolution tactile displays, e.g., (Koo et al, 2008), to mature to the level needed for such an approach. Thus, we focus our research on **tool-mediated contact**, where the user touches the target surface through an intermediate tool such as a metal probe, a ball-point pen, or a surgical scalpel.

Restricting haptography to tool-mediated interactions is not as limiting as it might initially seem. First, many everyday activities are conducted with a tool in hand, rather than with bare fingertips; tools extend the hand's capabilities for a specific task and protect it from damage. Second, humans are surprisingly good at discerning haptic surface properties such as stiffness and texture through an intermediate tool (Klatzky and Lederman, 2008; Yoshioka and Zhou, 2009). This acuity stems partly from the human capability for distal attribution, in which a simple hand-held tool comes to feel like an extension of one's own body (Loomis, 1992).

Haptographic Process. Haptography intentionally parallels modern photography, but the interactive nature of touch sets the two apart in several ways. To help clarify the differences, Table 1 lists analogous elements for the two domains, and Fig. 1 depicts an overview of the haptic capturing and recreating processes.

A haptographer begins by identifying an object with a unique or important feel; a museum curator might select an interesting historical relic, and a doctor might target

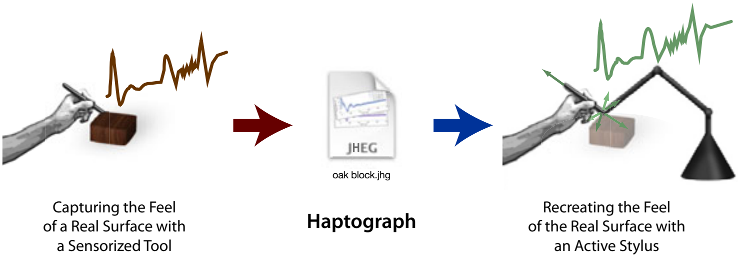

JHEG
oak block.jhg
Capturing the Feel of a Real Surface with a Sensorized Tool
Haptograph
Recreating the Feel of the Real Surface with an Active Stylus

Fig. 1 The envisioned approach of haptography will enable individuals to quickly capture, analyze, and recreate the exquisite feel of any surface they encounter in the real world.

an in vivo sample of tissue or bone. Working in his or her standard surroundings, the haptographer attaches a chosen tool tip to a highly sensorized hand-held instrument. For real-time haptography in teleoperation, the slave robot's tool is instrumented in the same way as a hand-held haptography tool. The operator then uses the tool to explore the object's surface via natural motions. The system collects multiple data streams throughout this interaction—all of the touch-based sensations that can be felt by a human hand holding a tool—including quantities such as the translation and rotation of the stylus and the object, the forces and torques applied to the object's surface, and the three-dimensional high frequency accelerations of the tool tip.

In teleoperation, the haptic interface held by the user seeks to recreate the measured sensations as they occur, and the challenge lies in perfecting this connection. In non-real-time applications, the haptographic processor needs to distill the recorded data into a general surface model so that future users can explore the virtual object in a natural way. Because the global shape of an object can be captured efficiently with optical sensors or reconstructed via computer-aided design (CAD) tools, haptography focuses on capturing surface attributes that are not readily apparent by sight, such as friction, texture, stiffness, and stickiness. Section 3 describes this identification problem in more detail, along with our preliminary work on texture modeling. We plan to store haptographs of different surface-tool interactions in a public online database so that virtual environment designers can apply haptographic surface swatches to chosen areas of synthetic objects.

An acquired haptographic model can be explored via any kinesthetic haptic interface, but the flexibility of the interaction and the quality of the haptic response will greatly depend on the mechanical, electrical, and computational design of the chosen system. Commercially available haptic devices generally struggle to duplicate the full feel of real surfaces. Thus, the second major aim of haptography research is to discover and refine high fidelity methods for rendering haptographs. As described in Section 4, tool-mediated haptography centers on the use of a dedicated high frequency vibration actuator, and we have tested this approach through creation of a prototype system. We want any individual to be able to use this "haptography handle" to explore 3D virtual surfaces and feel rich, natural sensations that are indistinguishable from those one would experience when touching the original item.

The Key to Haptographic Realism. Researchers studying virtual and remote environments have long sought to replicate the feel of real objects with a haptic interface. Arguably, the most important advance toward this goal came in 1994 when Massie and Salisbury presented the Phantom haptic interface. The design of this device evolved from three essential criteria, namely that "free space must feel free," "solid virtual objects must feel stiff," and "virtual constraints must not be easily saturated" (Massie and Salisbury, 1994, p. 296). Prioritization of these goals and clever mechanical design yielded a lightweight, easily backdrivable, three-degree-of-freedom robot arm actuated via base-mounted brushed DC motors equipped with high resolution optical encoders and smooth capstan cable drives. This invention inspired a wave of similar impedance-type haptic interfaces, many of which are now widely available as commercial products. Such systems are typically programmed to

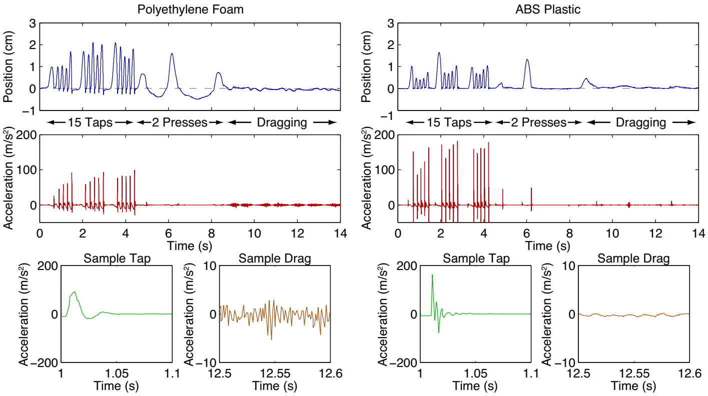


Fig. 2 Sample data from interactions with two real materials through a stylus instrumented with an accelerometer. One can quickly observe that the plastic is stiffer and smoother than the foam.

generate interaction forces via one-sided linear springs: when the measured device tip position intersects a region occupied by a virtual or remote object, the device outputs a restoring force that is proportional to the penetration vector.

While haptic interfaces designed and programmed in this way do succeed at conveying the global shape of virtual and remote items, the surfaces of these objects typically have only a weak haptic resemblance to real objects. Instead, haptically rendered shapes tend to feel soft and undefined, strangely slippery or peculiarly active and vibratory, and largely devoid of the coherent dynamic cues that one associates with natural surface properties. In one study targeted at this problem, human subjects used a Phantom to blindly tap on the stiffest possible spring-based virtual surface, among other real and virtual samples (Kuchenbecker et al, 2006). The spring-based surface received a realism rating of two on a scale from one to seven, where a seven denotes the feel of a real wooden block. Clearly something important is missing from these traditionally rendered haptic surfaces: we believe this deficiency stems from a reliance on haptic object models and interface hardware that prioritize low-frequency behavior over the **naturalistic high frequency accelerations** that give real objects their distinctive feel.

Human haptic capabilities are inherently asymmetric, allowing motion at just 8 to 10 Hz (Loomis and Lederman, 1986) and vibration perception up to 1000 Hz (Bell et al, 1994). As illustrated in Fig. 2, tool-mediated interactions with hard and textured objects create vibrations that strongly excite the Pacinian corpuscle mechanoreceptors in the glabrous skin of the human hand (Bell et al, 1994). It is clear that high frequency accelerations are a rich source of feedback during tool use, encoding information about surface material, surface texture, tool design, downward force, and tool velocity; a user naturally expects a haptic virtual environment to provide these same cues, but they are generally absent. When appropriate acceleration transients were added to the spring-based virtual surfaces in (Kuchenbecker et al, 2006), subjects responded with realism ratings of five out of seven, a significant improvement. Haptography is thus founded on the belief that only a haptic interface that authentically recreates these salient accelerations will be able to fool a human into believing that a virtual object is real, or a remote object is present.

3 Capturing the Feel of Real Surfaces

The first aim of haptography is to enable an individual to quickly and easily capture the feel of a real surface through a hand-held or teleoperated tool. This process yields a stream of interaction data that is distilled into a mathematical model of the surface's salient haptic properties for further analysis and subsequent re-creation.

Prior Work. Haptography takes a nontraditional approach to modeling object surfaces. The most carefully controlled attribute of a typical haptic virtual object is its global shape (Salisbury et al, 1995, 2004). As in computer graphics, these geometric models are usually composed of a polygonal mesh that defines the surface of the object. Contact with this mesh is then rendered as a spring force that seeks to push the user's virtual tool perpendicularly out of the surface. The high computational load incurred by fine meshes is avoided by blending the orientation of larger adjacent facets so that the normal force varies smoothly across the surface. The behavior of the coupling impedance can be modulated somewhat to change the feel of the surface, although nonidealities (e.g., position sensor quantization) cause instability at high stiffness and damping values. The additional surface properties of texture and friction are included in such models as a parametric relationship between the virtual tool's motion (position and velocity) and an additional force that is added to the spring response. For example, (Salisbury et al, 1995) use a sum of cosines for synthetic texture, (Minsky and Lederman, 1996) simulate roughness with a variety of lateral-force look-up tables based on surface location, and (Basdogan et al, 1997) create height-field textures inspired by the "bump map" approach from computer graphics. Numerous other hand-tuned surface representations have been developed, but most struggle to capture the rich variety of sensations caused by contact with real objects because they are not based on physical principles.

Rather than relying on hand-tuned parametric relationships, haptography derives virtual object models from real-world data. The idea of using a stream of physical measurements to create a haptic virtual environment is not new, but the central involvement of the human haptographer and the focus on high frequency accelerations are significant departures from previous work. The first discussion of a measurement-based modeling approach occurs in MacLean's 1996 paper on the "Haptic Camera," a fully automated one-degree-of-freedom probe that interacts with a mechanical system while recording position and force data to enable automatic fitting of a piecewise linear dynamic model. Autonomous interaction and identification techniques have since been applied to several other simple mechanical systems, such as switches and buttons (Colton and Hollerbach, 2005), and also to whole object contact through ACME, the robot-based Active Measurement Facility (Pai et al, 2000). In contrast to a robot, a human haptographer holding an instrumented tool can quickly and safely explore the surfaces of almost any physical object with natural motions that are fine-tuned in real time. However, there have been only a few previous efforts to generate haptic surface models from data recorded with a hand-held tool, typically involving either simple parametric relationships or direct playback (Okamura et al, 2008). For example, (Okamura et al, 2001) fit a decaying sinusoid model to acceleration transients recorded from a series of taps, while (Kuchenbecker et al, 2006) explicitly stored such recordings. Others have created hand-held tools fitted with sensors, e.g., (Pai and Rizun, 2003), but little has been done to distill the resulting data into surface models.

Our Approach. Given the limitations of traditional models and the success of several previous data-driven studies, we believe a sensorized hand-held tool and a sophisticated signal processing algorithm can be used to create very accurate models of the tool-mediated feel of any real surface. Haptographic probes are designed to record high bandwidth haptic data (tool position, velocity, acceleration, force, etc.) during natural human exploration of a surface. The recorded signals must then be segmented by interaction state (e.g., no contact, stationary contact, and sliding contact) and analyzed to yield mathematical models for the characteristic feel of each state, as well as for the transitions between states. The high sensory bandwidth of the human hand makes us believe that the realism of a haptographic surface model will strongly depend on its ability to encapsulate the high frequency accelerations of tool–surface contact. A full suite of haptographic capturing tools and algorithms will require a significant body of research, such as the physics-based modeling of tapping in (Fiene and Kuchenbecker, 2007); here, we present a new, general method for modeling the response of real textured surfaces felt through a tool.

Texture Modeling. We have developed a new method for using recorded data to obtain a predictive model of the tool accelerations produced during real texture exploration. As one can determine through quick experimentation, dragging a certain hand-held tool across a given surface creates vibrations that vary with both normal force and scanning velocity, as well as contact angle, hand configuration, and grip force. We are beginning our characterization of this complex dynamic system by analyzing the vertical acceleration experienced by a hand-held stylus as it is dragged across a variety of surfaces under different conditions. Our data set was recorded using a custom designed data collection apparatus from (Yoshioka, 2009) in well-controlled human subject trials where mean contact force, scanning velocity, and the other relevant variables were all held constant. The data collection system allows for precision recording of probe–texture interaction data including all contact forces and torques, three-dimensional tool acceleration, tool velocity, and the subject's grip force at a rate of 5000 Hz. For each recorded trial, we start by seeking a model that can generate an optimal prediction of the next real value in the acceleration time series given the previous n data points. We have found that this problem is best addressed with forward linear prediction, a common technique from system identification.

Forward Linear Prediction. The speech synthesis community has known for over thirty years that the complex dynamic vibrations created by the human vocal tract can be modeled by a form of the Wiener filter, the forward linear predictor (Atal and Hanauer, 1971). The standard procedure in speech synthesis is to treat the vocal tract response as an unknown filter that shapes a white noise excitation signal, which comes from air passed through the system by the glottis. The output is the spoken sound wave, which can be recorded with a microphone. Similarly, we record contact vibrations with an accelerometer and treat the dynamic response of the tool–texture interaction as a filter we wish to identify.

Fig. 3(a) shows the block diagram used for this system identification process, and the following mathematical analysis follows the conventions of (Benesty et al, 2008) as closely as possible. Our input signal $\mathbf{a}(k)$ is the original recorded time series of accelerations. The filter's output vector is defined as $\hat{\mathbf{a}}(k)$, which represents the forward linear prediction. $H(z)$ is assumed to be an IIR filter of length n of the form $H(z) = [-h_1 z^{-1} - h_2 z^{-2} \ldots -h_n z^{-n}]$. The residual of these two signals is the error vector $\mathbf{e}(k)$, and the transfer function $P(z)$ is:

$$\frac{E(z)}{A(z)} = 1 - H(z) = P(z) \tag{1}$$

We define the vector of filter coefficients as $\mathbf{h} = [h_1\ h_2\ h_3\ \ldots\ h_n]^T$, and the n-length time history of our input signal as $\mathbf{a}(k-1) = [a(k-1)\ a(k-2)\ \ldots\ a(k-n)]$. We then write the residual at each step in time with the following difference equation:

$$e(k) = a(k) - \hat{a}(k) = a(k) - \mathbf{h}^T \mathbf{a}(k-1) \tag{2}$$

Optimal filter values of $\mathbf{h}$ can be found by defining a suitable cost function. We use the standard choice of mean-square error, $J(\mathbf{h}) = \mathrm{E}\{\mathbf{e}^2(k)\}$, where $\mathrm{E}\{\cdot\}$ denotes mathematical expectation, as defined by (Benesty et al, 2008). When the gradient of $J(\mathbf{h})$ is flat, $\mathbf{h}$ is at an optimal value, $\mathbf{h}_o$. By algebraic manipulation we can derive the following result for the gradient:

$$\frac{\partial J(\mathbf{h})}{\partial \mathbf{h}} = -2\mathrm{E}\{(e(k)\mathbf{a}(k-1))\} \tag{3}$$

When the gradient is flat at $\mathbf{h}_o$, the error is at a minimum $\mathbf{e}_o(k)$, and we can simplify the problem to:

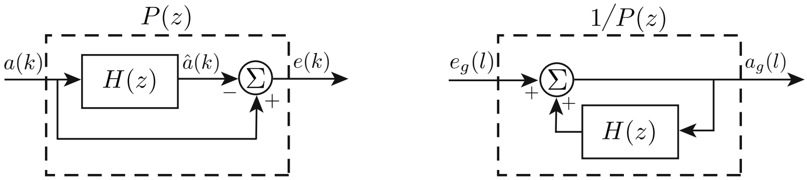
$$\mathrm{E}\{e_o(k)\mathbf{a}(k-1)\} = \mathbf{0}_{nx1} \tag{4}$$

By substituting values for the cross-correlation matrix $(\mathbf{R} = \mathbf{a}(k-1)\mathbf{a}^T(k-1))$ and the cross-correlation vector $(\mathbf{p} = \mathbf{a}(k-1)a(k))$ into (4), we arrive at the Wiener-Hopf equation:

$$\mathbf{R}\,\mathbf{h}_o = \mathbf{p} \tag{5}$$

Assuming non-singular $\mathbf{R}$, the optimal forward predictor coefficients can be found by simply inverting the cross-correlation matrix, such that $\mathbf{h}_o = \mathbf{R}^{-1}\mathbf{p}$. Alternatively, we can use a more efficient recursive method, such as the Levinson-Durbin algorithm (Durbin, 1960), to solve for $\mathbf{h}_o$ from (5). For demonstration, Fig. 4 shows a sample plot of $\mathbf{a}(k)$, $\hat{\mathbf{a}}(k)$, and $\mathbf{e}(k)$ for the optimal filter $H(z)$ of order $n = 120$.

Signal Generation. The previous section details a process for finding the linear transfer function $H(z)$ that is best able to predict the acceleration response of a



(a) LPC forward prediction model.

(b) Acceleration synthesis model.

Fig. 3 Block diagrams for prediction of the next contact acceleration $\hat{a}(k)$ given the recorded series $a(k)$ and synthesis of an acceleration signal $a_g(l)$ from the white noise input $e_g(l)$.

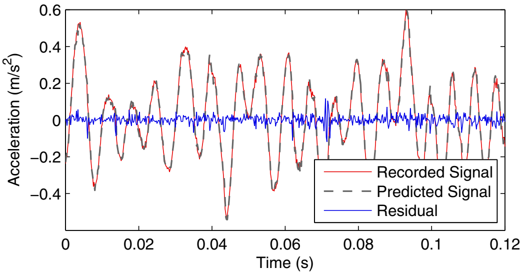


Fig. 4 Forward prediction signal generation and residual error. The data shown are from a real sample of organza fabric mounted on a stiff substrate and touched with a plastic probe at a velocity of 4.0 cm/s and a downward force of 1.5 N. The linear prediction filter $H(z)$ includes 120 coefficients ($n = 120$).

texture based on its previous n acceleration values. Subtracting the predicted response from the recorded signal removes almost all its spectral components, leaving only the noise signal $\mathbf{e}(k)$, which is ideally white and Gaussian. This section describes how to reverse this process and obtain a completely new (but spectrally similar) acceleration signal based on a white noise input, given an identified filter $H(z)$.

As seen in Fig. 3(b), the input signal $\mathbf{e}_g(l)$ is a white noise vector that we generate in real time. The output vector is $\mathbf{a}_g(l)$, a synthesized acceleration signal with spectral properties that are very close to those of the real data signal $\mathbf{a}(k)$ for which the filter $H(z)$ is tuned; higher order filters generally result in a better spectral match. By rewriting (1), we can formulate this new transfer function as follows:

$$\frac{A_g(z)}{E_g(z)} = \frac{1}{1 - H(z)} = \frac{1}{P(z)} \tag{6}$$

We now observe that the difference equation for the synthesized acceleration is:

$$\mathbf{a}_g(l) = \mathbf{e}_g(l) + \mathbf{h}^T \mathbf{a}_g(l-1) \tag{7}$$

During texture synthesis, we generate white noise with a Gaussian distribution of amplitudes and apply it to (7). One should note that the signal power of the white noise input is important for creating an acceleration signal $\mathbf{a}_g(l)$ with the proper magnitude. The power of the generated noise signal $P\{\mathbf{e}_g(l)\}$ must be equivalent to

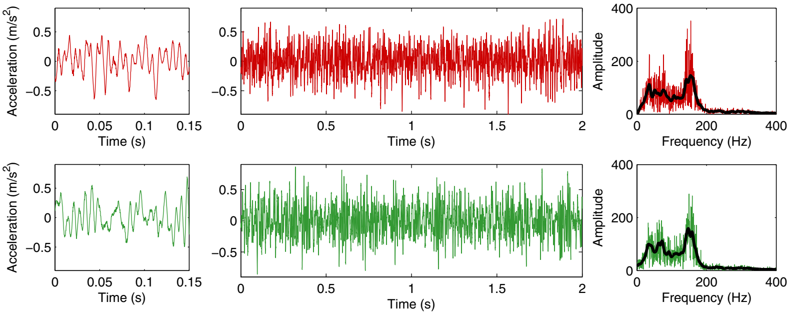


Fig. 5 Time- and frequency-domain views of a recorded acceleration and a signal synthesized to emulate that interaction using our novel texture-modeling techniques. The real setup and the synthesis filter are the same as those used in Fig. 4.

that of the power remaining in the residual signal, $P\{\mathbf{e}(k)\}$, after filter optimization. We have achieved good results when applying this acceleration modeling technique to data from many surfaces and at many levels of downward force and translational velocity. Fig. 5 shows one such sample in both the time and frequency domains.

Future Work. We are encouraged by the expressiveness and versatility of linear prediction for synthesizing realistic texture acceleration waveforms, and we are in the process of investigating many additional aspects of this approach. For example, how many filter coefficients are required to capture the haptically salient properties of an individual texture trial? And how should one synthesize accelerations for values of downward force and scanning velocity that were not explicitly tested? Currently, we fit data sparsely sampled from this parameter space and then use two-dimensional linear interpolation to choose coefficients of $H(z)$ for unique combinations of these parameters. In the future we intend to look into interpolating between filter poles or cepstral coefficients, both of which are directly related to the filter coefficients $\mathbf{h}$. More generally, we need to develop methods for processing data from interactions that are less controlled and for making models that go beyond texture to include other salient surface properties. In addition to increasing our knowledge of tool–surface contact, we hope that these haptographic modeling methods can be used to provide sensations that closely mirror those of real interactions, and also to evaluate the fidelity of virtually rendered haptic surfaces.

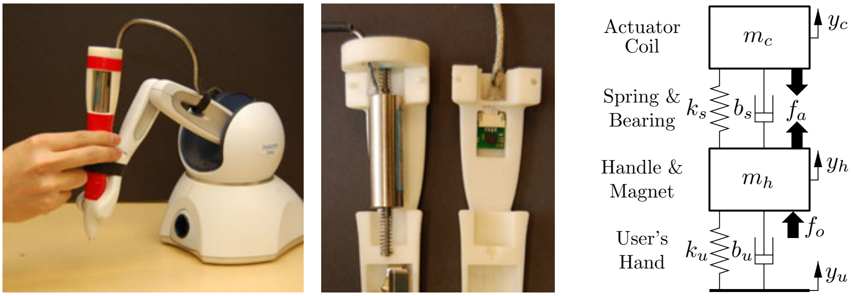
4 Recreating the Feel of Real Surfaces

The second aim of haptography is to enable an individual to freely explore a virtual or remote surface via a haptic interface without being able to distinguish its feel from that of the real object being portrayed. Realistically recreating haptographic models requires haptic device hardware and control algorithms that excel at delivering high frequency tool accelerations without impeding free-space hand motion.

Prior Work. During contact with a virtual or remote object, traditional haptic systems employ the device's actuators (usually base-mounted DC motors) to apply a force to the user's hand based on tool tip penetration, which is inherently a slowly changing signal. The mechanical elements that connect the motor to the hand would ideally be massless, frictionless, and infinitely stiff, but no real device can meet these requirements; instead, the dynamics of the intervening linkages, joints, and cables distort the output of the motors, which especially interferes with the display of any high frequency vibrations (Campion and Hayward, 2005; Kuchenbecker et al, 2006). Still, some previous work has shown that vibrations displayed with such actuators can improve the perceived realism of spring-based virtual surfaces, but these approaches require either extensive human-subject tuning (Okamura et al, 2001) or exhaustive device characterization (Kuchenbecker et al, 2006). Furthermore, high frequency base motor actuation is susceptible to configuration-based and user-based changes in the system's high-frequency dynamics, so it cannot achieve the consistent, high fidelity feel needed for haptography.

A viable alternative can be found in (Kontarinis and Howe, 1995)'s approach to teleoperation, where high frequency slave tool accelerations were overlaid on standard low frequency force feedback via an inverted speaker mounted near the user's fingertips. The slave acceleration was amplified by an empirically determined gain to drive the actuator, and the system's acceleration output was reported to vary by a factor of 2.24 across the frequency range of interest. Human subject tests indicated that this simple dual-actuator feedback strategy increased user performance in several tasks and also improved the "feel" of the interface, one of the main goals of haptography. Since this encouraging early work, several groups have created interesting active styli meant to be used without a force-feedback device, e.g., (Yao et al, 2005). The only project closer to our interests is that of (Wall and Harwin, 2001), who made a vibrotactile display stylus to study the effect of device output bandwidth on virtual grating perception. Their system uses a voice-coil actuator between the stylus and the end-effector of a desktop haptic device, with a controller that seeks to regulate actuator displacement using high-resolution measurements from a parallel LVDT sensor. The associated human-subject study found that the additional actuator between the hand and the haptic device significantly improved the rendering of virtual gratings but also reduced the system's ability to render stiff springs.

Our Approach. Considering the limitations of base-mounted motors and the results others have achieved with auxiliary actuators, we believe that haptographic models can be excellently recreated by attaching a high bandwidth bidirectional linear actuator to the handle of a typical haptic interface. This "haptography handle" should be designed and controlled to enable the system to significantly accelerate the user's hand at high vibrotactile frequencies (20–1000 Hz) in real time, while it is being held and moved around by the user. Imposing a grounded force at the handle is very challenging, so instead we attach an additional mass to the handle through a spring and a sliding joint. The auxiliary actuator applies equal and opposite forces on the handle and this mass, thereby pushing and pulling them relative to one another. Such a system can be carefully controlled only by understanding its mechanical dynamics and their impact on the user's experience. One final benefit to this approach is that we believe it will require only one linear actuator (rather



(a) Haptography handle in use (b) Handle interior (c) Dynamic model

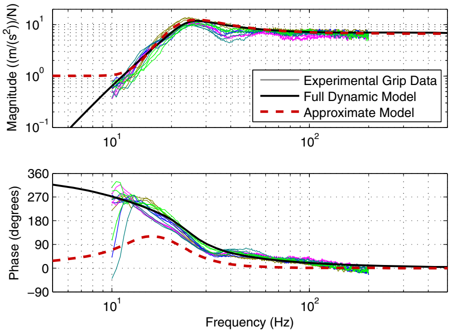
Fig. 6 A prototype haptography handle for use with the SensAble Phantom Omni. The voice coil actuator applies a high frequency force f_a between the coil and the magnet to accelerate the handle.

than three) because the human hand is not particularly sensitive to the direction of high-frequency vibrations (Bell et al, 1994).

Sample Implementation: Haptography Handle for the Phantom Omni. To evaluate the merits of our approach, we developed the prototype shown in Fig. 6 to act as an interchangeable handle for the Phantom Omni, a widely available impedance-type haptic device from SensAble Technologies, Inc.

Prototype. At the heart of our design is an NCM02-05-005-4JB linear voice coil actuator from H2W Technologies. We have installed this actuator in a moving coil configuration, where the permanent magnet core is rigidly attached to a handle and the coil is free to slide relative to this core along internal jeweled bearings. Additionally, we place compression springs at both ends of the coil to center it within the actuator's travel limits. The actuator is driven with a high bandwidth linear current amplifier, and it can output a peak force of 6.6 N. For more details on this actuator and our experimental procedures, please consult (McMahan and Kuchenbecker, 2009b), which describes an earlier prototype. Mounting this haptography handle to an Omni allows for measurement of the position and velocity of the handle, as well as the exertion of low-frequency forces, via the Omni's base-mounted encoders and DC motors. The addition of a dedicated voice coil actuator gives this low cost haptic device the capability of providing the high frequency contact accelerations that are essential to haptography.

System Dynamics. In order to accurately control the handle accelerations felt by the user, we must characterize the dynamics of our system. We use the lumped-parameter model shown in Fig 6 to represent our system: m_c is the mass of the actuator coil, k_s is the combined stiffness of the centering springs, b_s represents viscous friction in the linear bearings, f_a is the electromagnetic force exerted by the actuator, m_h is the effective mass of the handle and magnet, and f_o represents the forces provided by the Omni. The user is modeled as a parallel spring and damper



Full Dynamic Model

Parameter	Value
m_c	0.018 kg
k_s	399 N/m
b_s	1.2 N/(m/s)
m_h	0.145 kg
k_u	1000 N/m
b_u	20 N/(m/s)

Fig. 7 Frequency-domain system identification validates the structure of our dynamic model and enables the selection of appropriate values for parameters that cannot be directly measured.

(k_u and b_u) that connect the handle mass to the hand's set-point position, y_u. We can then derive the transfer function from actuator force to handle acceleration:

$$\frac{A_h(s)}{F_a(s)} = \frac{m_c s^4}{(m_c s^2 + b_s s + k_s)(m_h s^2 + (b_s + b_u)s + (k_s + k_u)) - (b_s s + k_s)^2} \tag{8}$$

Note that the Omni force f_o and the hand set-point y_u are both low frequency and thus will not affect the high frequency accelerations felt by the user.

We empirically validate and tune this transfer function by sending a repeating 10–200 Hz swept sinusoid force command to the linear voice coil actuator and recording the resulting accelerations at the handle with an accelerometer. We performed three trials of this test with five different users, each lightly holding the stylus with their right hand in a three-fingered pinch grip. Frequency-domain analysis of these tests guides the selection of model parameters. Fig. 7 shows the empirical transfer function estimates from the grip experiments, as well as the parameters chosen for the full dynamic model and its frequency-domain response.

This model enables us to design a *dynamically compensated controller* targeted at good acceleration tracking; our present controller consists of a feedforward term that inverts our estimate of the transfer function $A_h(s)/F_a(s)$ in order to determine the proper actuator force needed to achieve a desired handle acceleration. A careful look at (8) shows that naively inverting this transfer function will result the placement of four poles at the origin, which corresponds with a quadruple integrator in the controller. A controller with a quadruple integrator has infinite gain at steady-state and very high gain at low frequencies. These large gains pose a problem because they will immediately saturate the maximum force and deflection capabilities of our linear actuator. As a result, we approximate this transfer function with one that has finite DC gain, but still manages to capture the magnitude response of the full dynamic model in the important frequency range of 20-1000 Hz. The frequency-domain response of this approximate model is also shown in Fig. 7.

Teleoperation Testing. We tested our handle's performance at recreating realistic contact accelerations by conducting master-slave teleoperation experiments; the

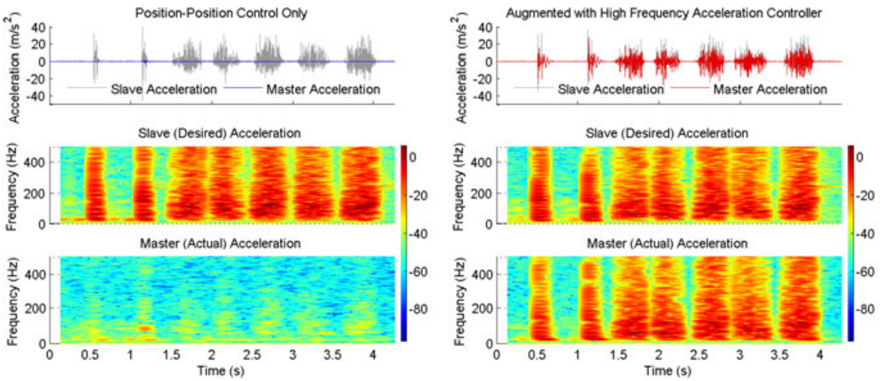


Fig. 8 Time- and frequency-domain results for the teleoperation experiments.

operator (grasping the haptography handle) uses a master Omni to command a slave Omni to perform exploratory tapping and dragging motions on a remote piece of unfinished plywood through a position-position controller. This configuration allows us to obtain real contact accelerations from an accelerometer mounted to the slave Omni's end effector and to render these accelerations to the user in real-time via the haptography handle. This experiment also serves as a proof-of-concept demonstration for haptography's potential use in teleoperation applications.

Attempting to recreate only the high frequency accelerations measured along the longitudinal axis of the slave's tool tip, we ran the experiment twice: once without using the voice coil actuator and once driving it with our dynamically compensated controller. In both cases, the operator tapped twice on the surface and then laterally dragged the tool tip five times. Fig. 8 shows time domain plots of the slave (desired) and master (actual) accelerations recorded during these experiments, as well as spectrograms of these signals. Visual inspection of these plots shows that the Omni's native motors and the implemented position-position controller do a poor job of transmitting high frequency accelerations to the user. However, augmenting the system with our dedicated vibration actuator and dynamically compensated controller provides a substantial improvement. Without this actuation, the normalized RMS error between actual and desired acceleration spectrograms is 100%, while auxiliary actuation brings this strict error metric down to 48%. Still, there is room for further refinement of the controller, as one can observe a general trend of underactuation and also some phase lag at lower frequencies.

Hands-On Demonstration. To obtain qualitative feedback about the feel of this system, we demonstrated the haptography handle in bilateral teleoperation at the 2009 IEEE World Haptics Conference (McMahan and Kuchenbecker, 2009a). Conference attendees were invited to use the master–slave Omni system to remotely explore textured samples both with and without acceleration feedback from the dedicated actuator. The demonstration was well received and participants provided a great deal of positive feedback, especially that the accelerations allowed them to feel small details and surface textures that were not detectable with only the position-position controller. Several participants thus commented that their hand felt "numb" when they explored the samples without haptographic feedback. The contact accelerations were also noted to make the surfaces feel "harder" even though the normal force provided by the Omni remained constant. This demonstration was honored to be selected by a panel of experts for the conference's Best Demonstration award.

Future Work. As we continue this research, we hope to improve the fidelity of our haptographic rendering by investigating more sophisticated acceleration controllers. We are also working to determine the perceptually correct mapping of three-dimensional accelerations to a one-dimensional actuator. Lastly, we are preparing to run human subject experiments to study the perceptual requirements for discrimination of realistic contact accelerations, as well as the potential benefits the approach of haptography may have on common applications for haptic interfaces.

Acknowledgements. We thank Rebecca Pierce for her initial work on haptographic data capture. We thank Takashi Yoshioka for providing texture contact data and for many useful conversations on haptography. This work was supported by the University of Pennsylvania and the National Science Foundation (grant IIS-0845670).

References

Atal, B.S., Hanauer, S.L.: Speech analysis and synthesis by linear prediction of the speech wave. Journal of the Acoustic Society of America 50(2), 637–655 (1971)

Basdogan, C., Ho, C.H., Srinivasan, M.A.: A ray-based haptic rendering technique for displaying shape and texture of 3D objects in virtual environments. Proc. ASME Dynamic Systems and Control Division, DSC 61, 77–84 (1997)

Bell, J., Bolanowski, S., Holmes, M.H.: The structure and function of Pacinian corpuscles: A review. Progress in Neurobiology 42(1), 79–128 (1994)

Benesty, J., Sondhi, M.M., Huang, Y. (eds.): Springer Handbook of Speech Processing. Springer, Berlin (2008)

Campion, G., Hayward, V.: Fundamental limits in the rendering of virtual haptic textures. In: Proc. IEEE World Haptics Conference, pp. 263–270 (2005)

Colton, M.B., Hollerbach, J.M.: Identification of nonlinear passive devices for haptic simulations. In: Proc. IEEE World Haptics Conference, pp. 363–368 (2005)

Durbin, J.: The fitting of time series models. Revue de l'Institut International de Statistique / Review of the International Statistical Institute 28(3), 233–244 (1960)

Fiene, J.P., Kuchenbecker, K.J.: Shaping event-based haptic transients via an improved understanding of real contact dynamics. In: Proc. IEEE World Haptics Conference, pp. 170–175 (2007)

Klatzky, R.L., Lederman, S.J.: Touch. In: Healy, A.F., Proctor, R.W. (eds.) Handbook of Psychology. Experimental Psychology, ch. 6, vol. 4, pp. 147–176. John Wiley and Sons, Chichester (2003)

Klatzky, R.L., Lederman, S.J.: Perceiving object properties through a rigid link. In: Lin, M., Otaduy, M.(eds.) Haptic Rendering: Algorithms and Applications, vol. 1, pp. 7–19 (2008)

Kontarinis, D.A., Howe, R.D.: Tactile display of vibratory information in teleoperation and virtual environments. Presence 4(4), 387–402 (1995)

Koo, I.M., Jung, K., Koo, J.C., Nam, J.D., Lee, Y.K., Choi, H.R.: Development of soft-actuator-based wearable tactile display. IEEE Transactions on Robotics 24(3), 549–558 (2008)

Kuchenbecker, K.J.: Haptography: Capturing the feel of real objects to enable authentic haptic rendering (invited paper). In: Proc. Haptic in Ambient Systems (HAS) Workshop, in conjunction with the First International Conference on Ambient Media and Systems (2008)

Kuchenbecker, K.J., Fiene, J.P., Niemeyer, G.: Improving contact realism through event-based haptic feedback. IEEE Transactions on Visualization and Computer Graphics 12(2), 219–230 (2006)

Loomis, J.M.: Distal attribution and presence. Presence: Teleoperators and Virtual Environments 1(1), 113–119 (1992)

Loomis, J.M., Lederman, S.J.: Tactual perception. In: Boff, K.R., Kaufman, L., Thomas, J.P. (eds.) Handbook of Perception and Human Performance. Cognitive Processes and Performance, ch. 31, vol. II, pp. 1–41. John Wiley and Sons, Chichester (1986)

MacLean, K.: The ‘Haptic Camera’: A technique for characterizing and playing back haptic properties of real environments. Proc. ASME Dynamic Systems and Control Division, DSC 58, 459–467 (1996)

Massie, T.H., Salisbury, J.K.: The PHANToM haptic interface: A device for probing virtual objects. Proc. ASME Dynamic Systems and Control Division, DSC 55(1), 295–301 (1994)

McMahan, W., Kuchenbecker, K.J.: Displaying realistic contact accelerations via a dedicated vibration actuator. In: Proc. IEEE World Haptics Conference, pp. 613–614 (2009a)

McMahan, W., Kuchenbecker, K.J.: Haptic display of realistic tool contact via dynamically compensated control of a dedicated actuator. In: Proc. IEEE/RSJ International Conference on Intelligent RObots and Systems (2009b)

Minsky, M., Lederman, S.J.: Simulated haptic textures: Roughness. Proc. ASME Dynamics Systems and Control Division, DSC 58, 421–426 (1996)

Okamura, A.M., Cutkosky, M.R., Dennerlein, J.T.: Reality-based models for vibration feedback in virtual environments. IEEE/ASME Transactions on Mechatronics 6(3), 245–252 (2001)

Okamura, A.M., Kuchenbecker, K.J., Mahvash, M.: Measurement-based modeling for haptic rendering. In: Lin, M., Otaduy, M.(eds.) Haptic Rendering: Algorithms and Applications, ch. 21, pp. 443–467 (2008)

Pai, D.K., Rizun, P.: The WHaT: a wireless haptic texture sensor. In: Proc. IEEE Haptics Symposium, pp. 3–9 (2003)

Pai, D.K., Lang, J., Lloyd, J., Woodham, R.J.: ACME, a telerobotic active measurement facility. In:Experimental Robotics VI, Proc. Int. Symposium on Experimental Robotics. LNCS, vol. 250, pp. 391–400. Springer, Heidelberg (1999)

Salisbury, K., Conti, F., Barbagli, F.: Haptic rendering: Introductory concepts. IEEE Computer Graphics and Applications 24(2), 24–32 (2004)

Salisbury, Z., Zilles, C.B., Salisbury, J.K.: A constraint-based god-object method for haptic display. Proc. IEEE/RSJ International Conference on Intelligent RObots and Systems 3, 146–151 (1995)

Sun, Y., Hollerbach, J.M., Mascaro, S.A.: EigenNail for finger force direction recognition. In: Proc. IEEE International Conference on Robotics and Automation, pp. 497–502 (2007)

Wall, S.A., Harwin, W.: A high bandwidth interface for haptic human computer interaction. Mechatronics 11(4), 371–387 (2001)

Yao, H.Y., Hayward, V., Ellis, R.E.: A tactile enhancement instrument for minimally invasive surgery. Computer-Aided Surgery 10(4), 233–239 (2005)

Yoshioka, T.: Probe–texture interaction dataset. Personal communication (2009)

Yoshioka, T., Zhou, J.: Factors involved in tactile texture perception through probes. Advanced Robotics 23, 747–766 (2009)

Towards the Robotic Co-Worker

Sami Haddadin, Michael Suppa, Stefan Fuchs, Tim Bodenmüller,
Alin Albu-Schäffer, and Gerd Hirzinger

Abstract. Recently, robots have gained capabilities in both sensing and actuation, which enable operation in the proximity of humans. Even direct physical interaction has become possible without suffering the decrease in speed and payload. The DLR Lightweight Robot III (LWR-III), whose technology is currently being transferred to the robot manufacturer KUKA Roboter GmbH, is such a device capable of realizing various features crucial for direct interaction with humans. Impedance control and collision detection with adequate reaction are key components for enabling "soft and safe" robotics. The implementation of a sensor based robotic co-worker that brings robots closer to humans in industrial settings and achieve close cooperation is an important goal in robotics. Despite being a common vision in robotics it has not become reality yet, as there are various open questions still to be answered. In this paper a sound concept and a prototype implementation of a co-worker scenario are developed in order to demonstrate that state-of-the-art technology is now mature enough to reach this aspiring aim. We support our ideas by addressing the industrially relevant bin-picking problem with the LWR-III, which is equipped with a Time-of-Flight camera for object recognition and the DLR 3D-Modeller for generating accurate environment models. The paper describes the sophisticated control schemes of the robot in combination with robust computer vision algorithms, which lead to a reliable solution for the addressed problem. Strategies are devised for safe interaction with the human during task execution, state depending robot behavior, and the appropriate mechanisms, to realize robustness in partially unstructured environments.

Sami Haddadin · Michael Suppa · Stefan Fuchs · Tim Bodenmüller ·
Alin Albu-Schäffer · Gerd Hirzinger
Institute of Robotics and Mechatronics
DLR e.V. - German Aerospace Center
P.O. Box 1116, D-82230 Wessling, Germany
e-mail: sami.haddadin@dlr.de

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 261–282.
springerlink.com

1 Motivation and Introduction

The idea of human and robot working together is of major interest for both the academic community and industrial robot manufacturers. Pioneering examples of intimate collaboration between human and robot, which origin can be found in [1], are Intelligent Assist Devices (IADs), as the skill assist described in [2]. In 1983 a method was proposed at DLR for allowing immediate "programming by touch" of a robot through a force-torque-sensor-ball [3], see Fig. 1 (left).

In this paper, we propose an approach to effectively combine human and robot capabilities for performing tasks in a partially unknown workcell, i.e. a semi-structured environment. We elaborate the theoretical basis, prerequisites regarding task execution and safe interaction mainly relying on sensor based reaction strategies. The concept requires flexibility from the robot in terms of sensor integration and programming. This flexibility is currently not available in the state-of-the-art first generation industrial robots, designed mainly to position objects or tools in six degrees of freedom (DoF).

However, for the second generation industrial robot, a fundamental paradigm shift is required to enable the implementation of the robotic co-worker. This concept is derived from a meaningful fusion of robots with innovative and robust control concepts, so called "soft robotics" features, and exteroceptive sensing as e.g. 3D vision sensing modalities for safely perceiving the environment of the robot. Together with additional sensing capabilities for surveillance such technology will open entirely new application fields and manufacturing approaches. In order to develop and evaluate the proposed concept, the DLR Co-Worker was constructed as a demonstration platform, see Fig. 1 (right).

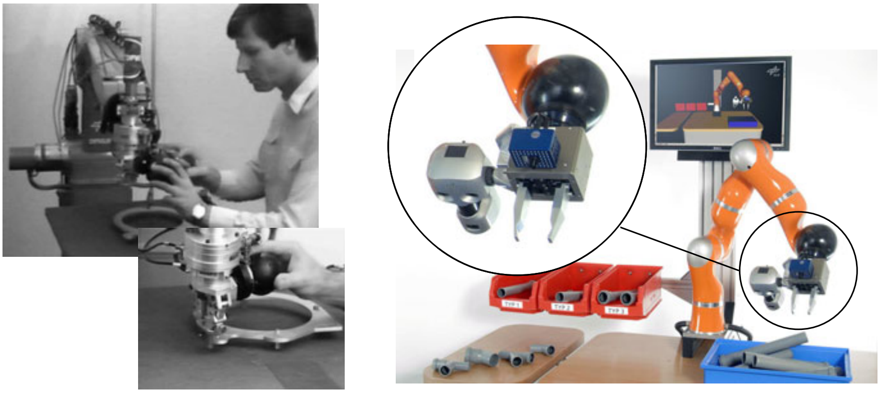


Fig. 1 The concept of sensor programming was developed at DLR in 1983 for teaching robot paths and forces/torques simultaneously (left). The DLR Co-Worker consisting of the DLR Lightweight Robot III, the DLR 3D-Modeller (DLR-3DMo), and a Time-of-Flight Camera (ToF-camera) (right).

Complementary sensor fusion[1] plays a key role in achieving our desired performance by the combination of complementary input information. As demonstrated in [4] a prioritized and sequential use of vision and force sensor based control leads to robust, fast, and efficient task completion using the appropriate sensor information depending on the particular situation. Currently, we believe that parallel use of both is needed mainly for very specific problems, which are usually irrelevant for industrial settings.

Presently, industrial robot applications require complete knowledge of the process and environment. This approach is prone to errors due to model inaccuracies. Our central approach is to use intelligent sensor-based reaction strategies to overcome the weaknesses of purely model-based techniques. Thus, we can deal with sensor noise and limited robot positioning accuracy. The robot task is described in high-level functions encapsulated in the states of hybrid automata, where transitions base on decisions made using sensor inputs. This enables the robot to react to ”unexpected” events not foreseen by the programmer. These events are induced by the human behavior, which cannot be completely modeled analytically. Apart from robust behavior, safety is of fundamental concern [5, 6, 7] if human-robot cooperation shall ever be realized beyond the prototype phase. In terms of mechanical design it is not effective to attempt to use large robots and try to make them sufficiently safe [8, 9]. We showed in recent work that in physical human-robot interaction (pHRI) slight collisions with the robot are not fatal, if robots with suitable mechanical design are used and the proper sensor-based reaction strategies are implemented [10]. Furthermore, the human is encouraged in our setup to physically interact with the robot as a modality to ”communicate” with it and provide task-relevant information. This also improves the fault tolerance level of the task since only absolutely worst-case contacts are solved by a complete emergency stop in contrast to approaches for current robots.

Apart from the described approach, the presented concept for the robotic co-worker is fundamentally different from classical industrial ones. None of the components are supposed to be intrinsically fail safe, but the appropriate combination of all components makes the system more safe, robust, and reliable. We use multiple sensor information of the robot and external sensing for increasing the error tolerance and fault recovery rate. The work we present is an attempt to merge our results for safe Human-Robot Interaction with robust exteroceptive and external sensing to achieve the robotic co-worker. We will discuss how to extend our schemes and the available solutions for particular problems and finally reach the stage of a highly flexible state-based programming concept for various applications. This task description allows for novel switching strategies between control modes, sensory reaction strategies, and error handling.

In this overview paper we discuss mostly the general concept. For further readings and details on the methodologies the interested reader is referred to the cited literature.

[1] Please note the difference of complementary from competitive sensor fusion.

The remainder of this paper is organized as follows. First, the general functional modes required for a robotic co-worker are described. Then, the interaction concept is outlined in detail. Furthermore, the task description performed autonomously by the robot is elaborated. Finally, the developed concepts are applied to a robotic bin picking scenario with user interaction as a case study in order to present their practical relevance and implementation.

2 Functional Modes

Currently, industrial settings incorporate, in most cases, simple sequences of tasks whose execution orders are static, allowing sometimes some binary branching. Fault tolerance during task execution is, apart from certain counterexamples[2], usually not an issue due to the well designed environment. Furthermore, human-robot interaction is not yet safely and effectively implemented and the legal foundation for it is, to a large extent, non existent at the current stage. In industrial settings a fault immediately leads to a complete stop of the manufacturing process, i.e. robust behavior in an unstructured environment has not been addressed until now. We propose an integrative and flexible approach to carry out the desired task in a very robust yet efficient way. At the same time, this approach is able to distinguish between different fault stages, which stop the entire process and lower the efficiency only in the worst case. Flexible jumps within execution steps are part of the concept and do not require special treatment. In order to optimally combine human and robot capabilities, the robot must be able to quickly adapt to the human intention during task execution for both safe interaction and high productivity. Thus, the measured human state is the dominant transition between the proposed functional modes. Estimating the human state is a broad topic of research and has been addressed in recent work [11]. The focus is often on estimating the affective state of humans, which is of secondary interest during an industrial process. The more relevant information is the physical state that the human currently occupies, and the estimation of the human attention, so that a clear set of sufficient behaviors can be selected and activated, which leads to robust and reliable overall performance. This paper does not consider attention estimation, instead focusing on the human state.

We compiled the following selection of physical states to provide sufficient coverage for cases relevant to our study, also shown in Fig. 2 (left).

- **oP:** out of perception
- **iP:** in perception
- **iCM:** in collaborative mode
- **iHF:** in human-friendly zone

oP denotes that the human is out of the perceptional ranges of the robot and therefore not part of the running application. **iP** indicates that the human

[2] Checking for a successful grasp is e.g. commonly used.

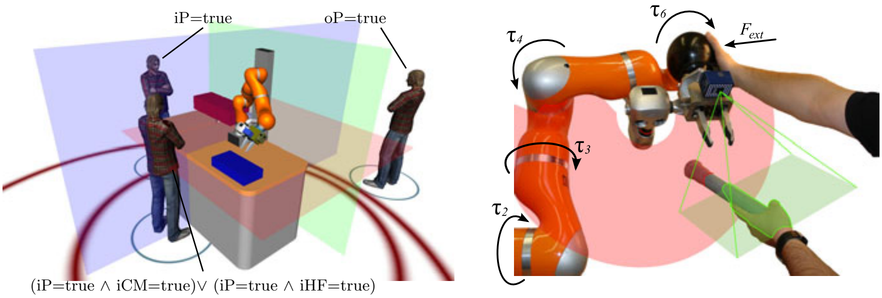


Fig. 2 Proximity and task partition (left) and modalities for multi sensor human robot interaction in the DLR Co-Worker (right).

is in the measurement range of the robot, and thus its presence has to be part of the robot control. **iCM** and **iHF** indicate whether a collaborative or human-friendly behavior must be ensured. Each physical state is subdivided, depending on the task. However, only when **iCM** = *true*, the collaborative intention should be taken into account: This leads to a complex physical interaction task. In this paper we will use the "hand-over and receive" process as an example, see Fig. 2 (right).

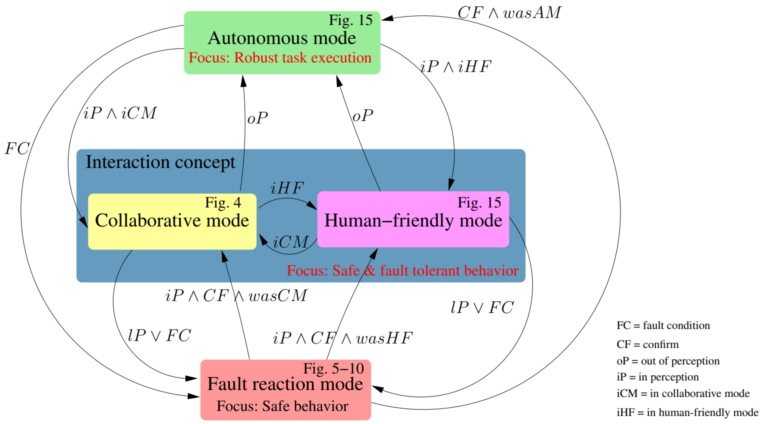


Fig. 3 Functional modes for the DLR Co-Worker.

The human state is primarily used to switch between different functional modes of the robot which in turn are associated with fault behavior. As shown in Fig. 3 we distinguish between four major functional modes of the robot in a co-worker scenario:

1. **Autonomous task execution:** autonomous mode in human absence
2. **Human-friendly behavior:** autonomous mode in human presence

3. **Co-Worker behavior:** cooperation with human in the loop
4. **Fault reaction behavior:** safe fault behavior with and without human in the loop

Their interrelation and transition conditions provide high flexibility in the application design. In the **first** functional mode the robot is autonomously fulfilling its given task without considering the human presence. The task is carried out under certain optimality criteria, such as cycle time, in order to maximize the productivity. In the **second** and **third** modes, we need a meaningful partition of the task space which subdivides the given workspace of the robot into regions of interaction. These incorporate the "hand over" schemes as described in Sec. 3.2 and human-friendly behavior, whose core elements are reactive collision avoidance and self-collision avoidance schemes. In the **third** mode interaction tasks are carried out that have to be specified or generated for fulfilling a common desired goal, involving a synergy of human and robot capabilities in an efficient manner. These two modes form an integrative interaction concept, allowing seamless switching between each other. The **fourth** mode defines the fault reaction behavior, addressing the appropriate and safe state dependent fault reaction of the robot. It incorporates both the robustness concepts during autonomous reaction, as well as human-safe behavior. Since each mode possesses an underlying safety concept, it will be described later in more detail.

3 Interaction Concept

In this section we describe the developed interaction schemes. First, the proposed task space partition is outlined, followed by the interaction layer, different collision avoidance techniques, as well as physical collision detection and reaction for safe pHRI. Finally, the resulting safety architecture, which unifies the different schemes, is presented.

3.1 Proximity and Task Partition

In case humans are in close proximity to robots in current industrial installations, the robots reside inside safety cages in order to prevent any physical contact and thus minimize the risk for humans. However, when humans and robots shall collaborate, such a plant design is no longer an option. The human location has to be taken into account in the control scheme and in higher level control of the robot as an integral part of the system design. The previously introduced physical human states have to be mapped into a meaningful topology shown in Fig. 2 (left), where the four distinct classes are indicated. They should be established with respect to the task and the robot workspace for assessing, whether the human does not have to be taken into account and therefore, the robot still behaves autonomously regardless of the iP state. In case the human does not enter the robot workspace, it is not necessary to degrade the productivity of the robot. In this sense the functional mode of the robot changes only, if the human clearly enters the workspace of the robot (indicated by the inner circle). If the human has entered the robot workspace a distinction between human friendly behavior (on the right side of the table in Fig. 2 (left)) and the cooperative mode (and their respective submodes) is required (on the left side of the table in Fig. 2 (left)). If perception is lost while $iP =$ true, the robot assumes a severe error condition, stops and waits for further instructions. If the presence of the human was not detected at all, i.e. a worst case from a safety point of view, various safe control schemes ensure the safety of the human during possible unforeseen collisions.

Defining these regions is part of the application design and definition phase. Furthermore, we introduce switching zones, which are boundary volumina of pre-defined thickness between task partitions (see Sec. 3.3 for details).

3.2 Interaction Layer

Interaction between robot and human is a delicate task, which needs multi-sensor information. Furthermore, robust as well as safe control schemes are called for to enable intuitive behavior. The main physical collaboration schemes are "joint manipulation" and "hand over and receive". "Parallel execution" may be part of a task, but usually without physical interaction. Some work has been carried out on exchanging objects between human and robot based on reaching gestures [12]. In [13] the concept of interaction history was used to achieve cooperative assembly.

Figure 2 (right) shows the "hand-over" and "receive" for the DLR Co-Worker Central entity is the LWR-III with its soft robotics features. As a default we utilize its high-performance impedance control, and only switch to other schemes, such as position control, if necessary. The robot is equipped with joint torque sensors in every joint. It is well suited for realizing various important features such as load loss detection and online load identification without additional force sensing in the wrist. Collision detection and reaction, depending on the potential physical severity of the impact and current state, is a central feature used for detection and isolating contacts of different intensity along the entire robot structure. By being able to distinguish different contact types, fault tolerant and situation suited behavior is possible.

We utilize virtual walls for avoiding collisions with the environment through control schemes. In order to realize an effective reactive behavior, it is important to change stiffness, velocity, disturbance residuals (see Sec. 3.3), trajectory generators, collision severity reaction strategies, and robot control parameters on the fly within the lower level control rates (here 1 ms), holding also during motion or state execution. With the combination of exteroceptive sensing, capabilities of object recognition, tool surveillance, and human proximity detection (shown in Fig. 2 (right)), we can achieve such aforementioned

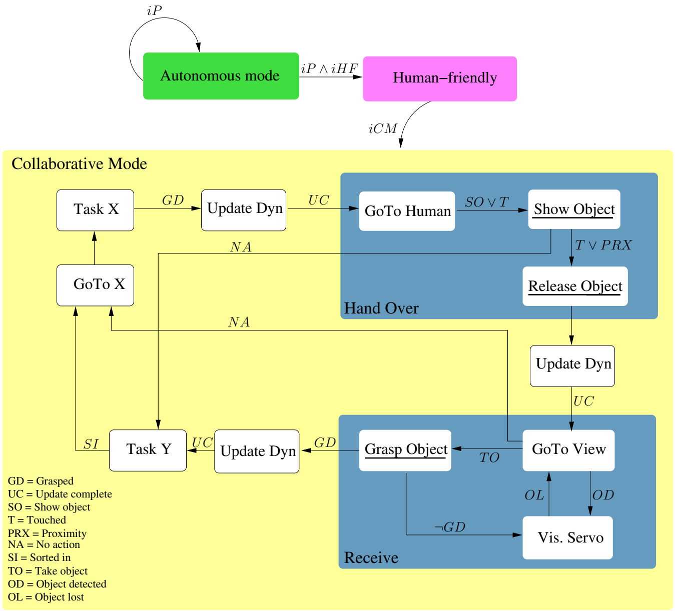


Fig. 4 Example for "hand-over" and "receive". Underlined states incorporate explicit physical interaction.

complex processes as "hand-over" and "receive", shown in Fig. 4. "Receiving" or "handing over" the object is simply triggered by touching the robot at any location along its entire structure or by using the proximity information from the mounted exteroceptive sensors.

3.3 Absolute Task Preserving Reaction: Time Scaling

While the robot is in human-friendly mode, its intention is to fulfill the desired task time efficiently, despite human presence. In order to accomplish this, it is necessary to equip the robot the robot with reactive motion generators that take into account the human proximity and thus prevent inefficient task abortion.

Trajectory scaling preserves the original motion path and at the same time provides compliant behavior by influencing the time generator of the desired trajectory, see [10]. This scheme can even be used to enable a position controlled robot to react compliantly in such a way that it remains on the nominal path, albeit with limited maximum forces in case of physical external disturbances.

A desired trajectory is usually parameterized with respect to time. If the discrete sampling time Δt is modified in such a way that it is used to respond to such external forces, it can be used to step back and forth along the

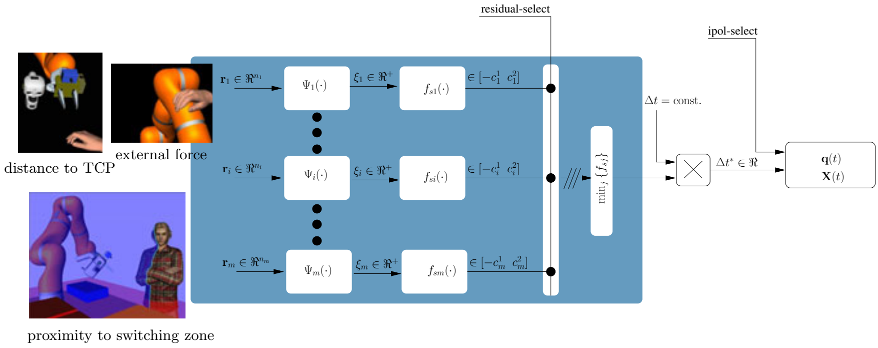


Fig. 5 Residual fusion for integrated trajectory scaling. Ψ_i is a normalization function and f_{si} a sigmoid function for time scaling, [10, 14].

desired path, by "scaling the trajectory in time". In our approach we use physical contact residuals such as the estimated external joint torque, or the external contact wrench, together with proximity based residual signals such as the human-robot proximity, the human-switching zones proximity, and the human-workspace proximity, see Fig. 5. The usefulness of the approach becomes also apparent when considering cases where humans are moving close to switching zones. If the robot would simply use binary switching information about the current state of the human, undesired oscillating behavior would occur due to the imprecise motions and decisions of the human. By using the human proximity to this border as a residual the robot always slows down and stops until the human clearly decides his next action. This way, the user receives intuitive visual feedback, indicating that the robot is aware of his presence and waits for further action.

The fusion of the different residuals is shown in Fig. 5 for several aforementioned signals. This concept allows us to bring quantities of different physical interpretation together and use them in a unified way for trajectory scaling. Each residual is normalized[3] and then nonlinearly shaped to be an intuitive time scale. Depending on the current state, the user can choose suitable residuals accordingly during application design.

3.4 Task Relaxing Reaction: Reactive Path Deformation

Apart form task preserving reaction as described in the previous subsection, reactive real-time reaction with task relaxation is an important element for dealing with dynamic environments as well. A well known technique in this respect is the elastic strips framework [15].

[3] Please note that we refer to an appropriate handling as e.g. projecting external forces to the velocity direction of the robot or similar transformations.

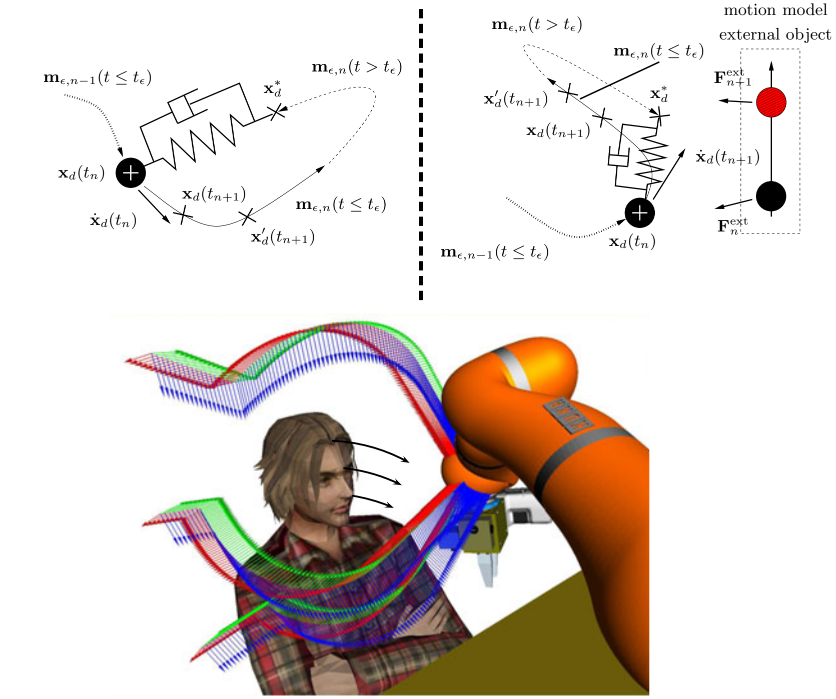

$\mathbf{m}_{\epsilon,n-1}(t \leq t_\epsilon)$
$\mathbf{x}_d^*$
$\mathbf{m}_{\epsilon,n}(t > t_\epsilon)$
$\mathbf{x}_d(t_n)$
$\mathbf{x}_d(t_{n+1})$
$\dot{\mathbf{x}}_d(t_n)$
$\mathbf{m}_{\epsilon,n}(t \leq t_\epsilon)$
$\mathbf{x}'_d(t_{n+1})$
motion model
external object
$\mathbf{m}_{\epsilon,n}(t > t_\epsilon)$
$\mathbf{m}_{\epsilon,n}(t \leq t_\epsilon)$
$\mathbf{F}^{\mathrm{ext}}_{n+1}$
$\mathbf{x}'_d(t_{n+1})$
$\mathbf{x}_d^*$
$\mathbf{x}_d(t_{n+1})$
$\dot{\mathbf{x}}_d(t_{n+1})$
$\mathbf{F}^{\mathrm{ext}}_{n}$
$\mathbf{m}_{\epsilon,n-1}(t \leq t_\epsilon)$
$\mathbf{x}_d(t_n)$

Fig. 6 Schematic views of the collision avoidance for two consecutive iteration steps. The upper left figure denotes free motion, whereas the upper right one takes into account a motion model of an external virtual object. The lower figure shows the collision avoidance for a full dynamic simulation.

The subsequent method uses a decoupled second order impedance system (mass-spring-damper) as a starting point and describes the (dynamic) environment by means of virtual forces, which are generated with a motion model of the according object, see Fig. 6 (upper). The virtual mass associated with the robot is placed at the starting configuration $\mathbf{x}_d(t_n), \dot{\mathbf{x}}_d(t_n)$ of the robot, whereas the equilibrium of the system is the desired goal configuration $\mathbf{x}_d^*$. The resulting trajectory of the excited system leads to (assuming no local minima) a valid trajectory towards the goal. However, such a simple solution by means of velocity and acceleration of the trajectory leads for most cases to very undesired properties of the generated path. In order to overcome this deficit we calculate the traverse path of the system $\mathbf{m}_{\epsilon,n}(t \leq t_\epsilon)$ every time step (incorporating the dynamic behavior of the environment) within a certain reasonable time interval t_ϵ, but dismiss the time information associated with it. In order to match a desired velocity $\dot{\mathbf{x}}'_d(t_{n+1})$, we search for the configuration $\mathbf{x}'_d(t_{n+1})$ along the path that ensures this velocity. Thus, we keep the smooth properties of the generated local path but the velocity of the robot can be commanded independently. During (virtual) contact the velocity can be additionally scaled similarly to the previously described trajectory scaling method. Thus, due to the collision avoidance, the robot would continously reduce speed, or even retract, and at the same time actively avoid the upcoming collision. Figure 6 (lower) shows the result for different starting points and the common goal configuration of the robot. The nominal trajectory is a straight line from different starting points to a common end point. The avoidance takes place for a dynamic motion of the human towards the robot.

3.5 Dealing with Physical Collisions

In our recent work we concentrated on evaluating the injury severity for the human during worst-case robot-human impacts [8]. Furthermore, we developed various control schemes to provide a high level of safety to the human during desired physical interaction and *unexpected collisions*. Numerous solutions against human injury are available. Crucial features for them are an effective physical collision detection and reaction. Furthermore, after a collision is detected and isolated, an appropriate reaction has to be triggered [10]. One possible solution is to stop the robot as soon as a collision is detected. Another one is to switch from position control to zero-gravity torque control [16] and let the robot react in a convenient compliant manner. These approaches provide the possibility to divide the impact severity into several stages, using a disturbance observer. This method for detecting contacts is also able to give an accurate estimation of the external joint torques $\boldsymbol{\tau}_{\mathbf{ext}}$, which in turn can be used to classify collisions with the environment according to their "severity level". This allows us to react variably to different *collision severity stages* [14], leading to a *collision severity based behavior*. Apart from this nominal contact detection, our algorithms are also able to detect malfunction of the joint torque sensors, based on model inconsistencies interpreted as a collision.

3.6 Safety Architecture

Apart from gaining insight into mechanisms behind safe pHRI and isolated tools, it is critical to determine how to apply the knowledge and methodologies in a consistent and appropriate manner. We have developed schemes to utilize these features appropriately in order to maximize task performance under the constraint of achieving sufficient human-friendly behavior, see Fig. 7. Each feature is shown at the according hierarchical level where it is introduced and made available in the appropriate layer of the process.

Figure 8 outlines how the fault management and emergency components are embedded as underlying components for each task. Every task has the appropriate low-severity-fault tolerant components to make it robust against external disturbances in general and prevent unnecessary task abortion. Each of them activates their distinct safety set S_j which is compatible with the particular goal (see Fig. 10 for details).

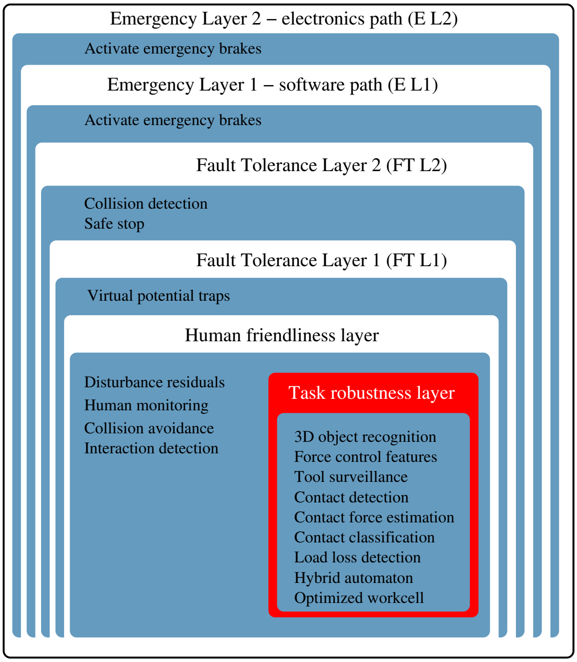


Fig. 7 Safety architecture of the DLR Co-Worker. Only the first two stages are user specific.

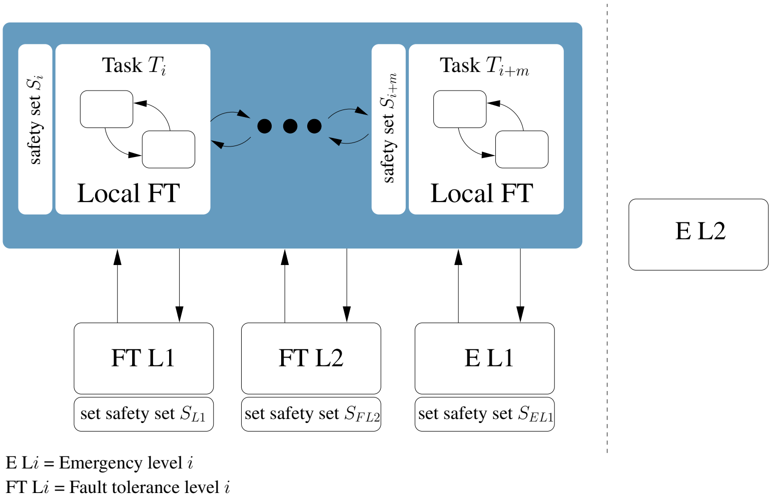
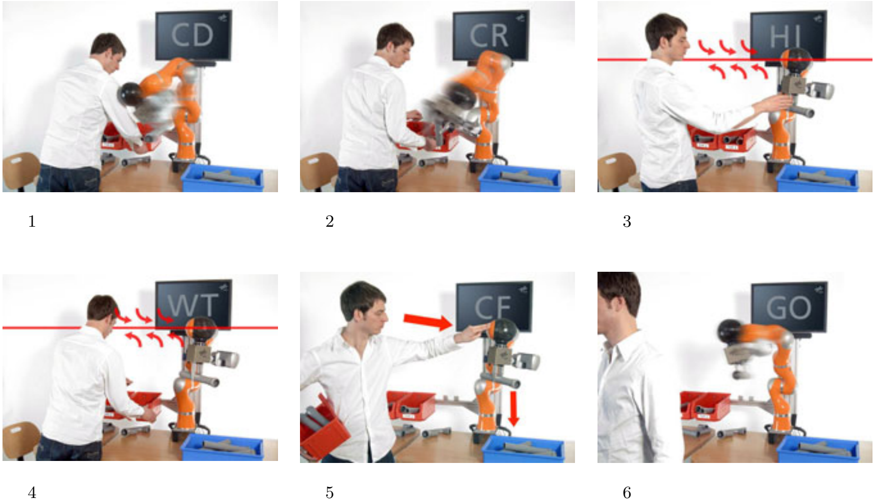


Fig. 8 Safety background of the DLR Co-Worker.



1 2 3

4 5 6

Fig. 9 Safe physical human-robot interaction. Detecting and recovering from a collision in FT L1. It was assumed that the human was not perceived to have entered the workspace.

Figure 9 shows an example of an *unexpected collision* between a worker and a human ①, leading to a collision in layer FT L1. The robot switches to a compliant behavior ② after the collision is detected (CD). Due to the collision reaction (CR), the robot is able to be freely moved in space. This could lead to secondary collisions with the environment. Therefore, we have designed nonlinear virtual walls (Fig. 2) with rigid properties to prevent physical collisions of the robot and secure the sensitive parts as the ToF-camera and the 3DMo. As a result, the human can simply grab the robot anywhere along the structure and hang it like a tool into a predefined arbitrarily shaped virtual potential trap[4] (HI) ③, which smoothly drags it in and keeps it trapped. The human can then complete his task, which he intends to fulfill ④, while the robot waits (WT) for further action. After completion is confirmed (CF) in ⑤ the robot continues ⑥ with the interrupted task (GO). If this was not the case, the robot stays in his constrained passive behavior until either a confirmation for continuation occurs, or a human would drag him out of the hang-in field, depending on a predefined direction of disturbance. Figure 10 shows how such behavior is triggered in a hybrid automaton and the safety sets involved in this process.

[4] This feature “feels” like a magnet.

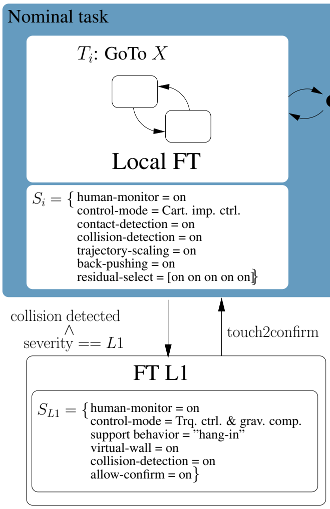

Nominal task
T_i: GoTo X
Local FT
$S_i = \{$ human-monitor = on
control-mode = Cart. imp. ctrl.
contact-detection = on
collision-detection = on
trajectory-scaling = on
back-pushing = on
residual-select = [on on on on on]$\}$
collision detected $\wedge$ severity == $L1$
touch2confirm
FT L1
$S_{L1} = \{$ human-monitor = on
control-mode = Trq. ctrl. & grav. comp.
support behavior = "hang-in"
virtual-wall = on
collision-detection = on
allow-confirm = on$\}$

Fig. 10 Safe reaction to a collision in FT L1 under the assumption that the human was not perceived to have entered the workspace. A simple and convenient behavior is triggered, which can be realized by intuitive use of well designed state dependent control scheme selection.

4 A Sensor Based Approach for Interactive Bin Picking

In this section we focus on describing our solutions to solve an industrially relevant autonomous task by combining computer vision techniques with soft-robotics features and embed it into an interaction scenario with the human. To demonstrate the performance of our system during autonomous task execution, we address the classical bin picking problem, which is well known since the mid-1980s. However, such problems have remained difficult to be solved effectively. This sentiment can be found in different literature, as exemplified below:

> "Even though an abundance of approaches has been presented a cost-effective standard solution has not been established yet."
>
> *Handbook of Robotics 2008* [17]

We have combined environmental modeling, robust and fast object recognition, as well as quick and robust grasping strategies in order to solve the given task. The setup depicted in Fig. 1 (right) serves as our demonstration platform. It is further used for realizing a scenario where the human assembles parts, which are supplied by the robot and, after a "hand over" and "receive" cycle, sorted into a depot by the robot, see Fig. 16. This fully sensor-based concept is entirely embedded in the proposed safe interaction concepts. The intention of this application is to augment human capabilities with the assistance of the robot and achieve seamless cooperation between each other.

4.1 Vision Concept

The LWR-III is equipped with two exteroceptive sensors: the DLR 3D-Modeller and a time-of-flight camera so that different proximity sensors with complementary features can be used within this scenario.

DLR 3D-Modeller

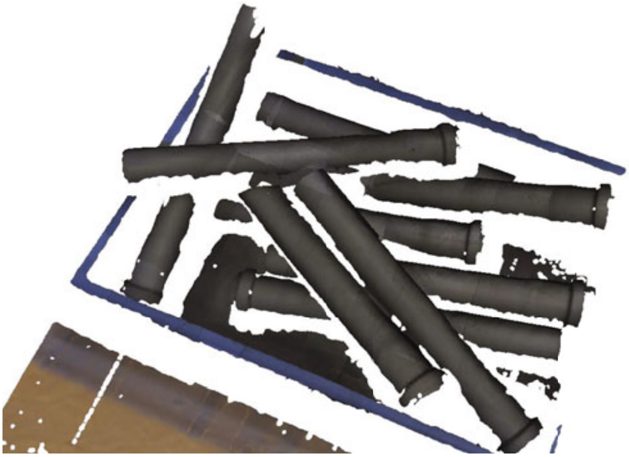


Fig. 11 Generated 3D model from a series of sweep scans over the filled bin.

System: The DLR-3DMo is a multi-purpose vision platform [18], which is equipped with two digital cameras, a miniaturized rotating laser scanner and two line laser modules, see Fig. 1 (right). The DLR-3DMo implements three range sensing techniques:

1. laser-range scanning [19]
2. laser-stripe profiling [20]
3. stereo vision

These techniques are applicable to a number of vision tasks, such as the generation of photo realistic 3D models, object tracking, collision detection, and autonomous exploration [21].

Implementation: The laser-range scanner, used for securely determining obstacles and free regions, provides range data enriched with a confidence value. The proposed application employs the rotating laser range scanner for two tasks. First, the wide scan angle of 270 degrees enables nearly complete surveillance of the working range around the gripper. Secondly, the measured distance data provides information about occupation of the space between the jaws of the gripper and indicates whether a target object is located there.

The laser-stripe profiler is used for modeling the environment and can be used for the localization of the bin or accurate modeling of the entire workcell, see Fig. 11. The shown model was generated with a series of sweep motions of the LWR-III across the scenario. The main purpose of the laser-stripe profiler is to acquire accurate data for model generation, in contrast to the safety functionality of the laser-range scanner.

Time-of-Flight camera

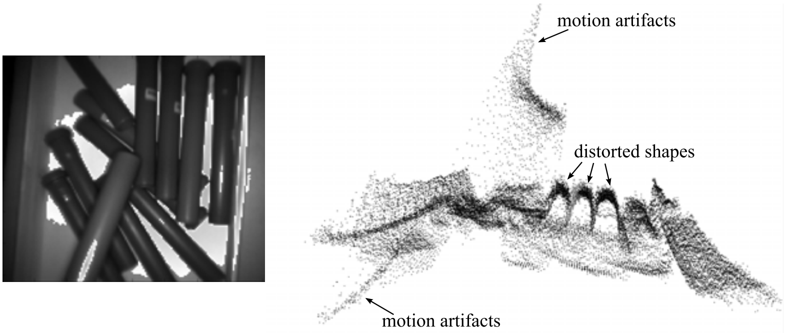


Fig. 12 Amplitude and depth data from view into the bin (left) showing large signal noise (right).

System: The ToF camera Swissranger SR 3000, mounted on the robot, has a resolution of 176 × 144 pixels. An important feature of this device, beneficial for this application, is the ability to capture $2\frac{1}{2}$D depth images at ≈ 25 Hz. Unlike stereo sensors, ToF-cameras can measure untextured surfaces because the measurement principle does not depend on corresponding features. Furthermore, due to the active illumination, ToF-cameras are robust against ambient illuminations changes. These properties enable the recently established use in the robotics domain for tracking, object detection, pose estimation, and collision avoidance. Nonetheless, the performance of distance measurements with ToF-cameras is still limited by a number of systematic and non-systematic error sources, which turn out to be a challenge for further processing.

Figure 12 highlights the non-systematic errors such as noise, artifacts from moving objects, and distorted shapes due to multiple reflections. While noise can be handled by appropriate filtering, the other errors mentioned here are system inherent. The systematic distance-related error can be corrected by a calibration step down to 3 mm, see [22].

Implementation: Generally, the high sampling rate of the ToF-camera guarantees fast object localization and robust object tracking performance based

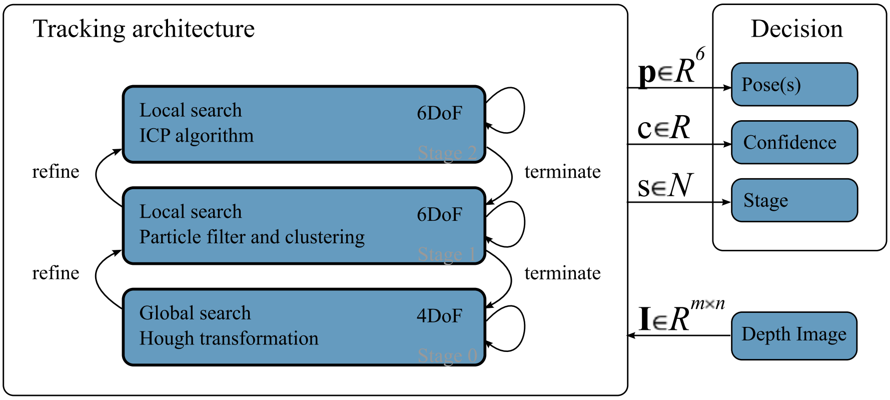


Fig. 13 Multi-stage tracking architecture based on [23].

on a three staged tracking architecture, see Figure 13. In each stage a different algorithm processes an incoming depth image to provide a list of pose hypotheses for the potential object, which is additionally tagged with a confidence value. The stages are continously monitored and executed according to suitable termination criteria or reentered for refinement.

The first stage is a global search, consisting of edge filtering and a Hough transformation for identifying lines as initial hypotheses for the tube location. In the second stage these hypotheses are locally consolidated and clustered by a particle filter. Third, an Iterative Closest Point algorithm (ICP) provides an accurate pose estimation of the target object at a frame rate of ≈ 25 Hz. Both ICP, and particle filter directly process 3D data, and a 3D model of the target. The 3D model is represented by a point set with corresponding normals. This can be either generated from CAD models or surface reconstruction. The object target can be localized and tracked with an accuracy of ≈ 7 mm.

4.2 Soft Robotics Control for Grasping

The soft-robotics features of the LWR-III greatly provide powerful tools to realize such a complex task as bin picking. Cartesian impedance control [16] is used as a key element for robust grasping despite the aforementioned recognition uncertainties. The impedance behavior of the robot is adjusted according to the current situation in order to achieve maximal robustness. Furthermore, the previously introduced strategies for fault detection are used to recognize impossible grasps or unexpected collisions with the environment based on force estimation. Furthermore, there are virtual walls preventing collisions with the static environment. The robustness of grasping against errors in object localization and errors in positioning due to the used impedance control is of great importance for this application. The grasping strategy shown in

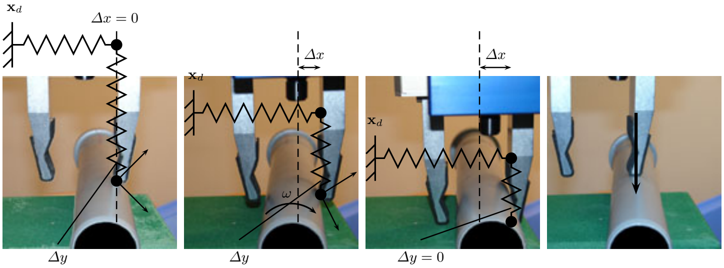


Fig. 14 Compliant grasping strategy.

Fig. 14 successfully copes with possible translational deviations in the range of 55 mm before the grasp fails. Due to the compliant behavior of the robot and gripper-object and object-ground friction, the object is rotated into the firm grasp. The last image shows a case expected to be a failure. However, due to the rotational stiffness we implemented along the axis perpendicular to the image plane grasping can still be achieved.

4.3 Autonomous Task Execution

Figure 15 depicts the autonomous bin picking task automaton, which merges the presented concepts into a high-level view of task description. The application is comprised of object recognition, grasping, and sort-in phases[5]. If the bin is depleted, the robot waits for further supply. Fault tolerant behavior is realized by introducing various branching possibilities for each state

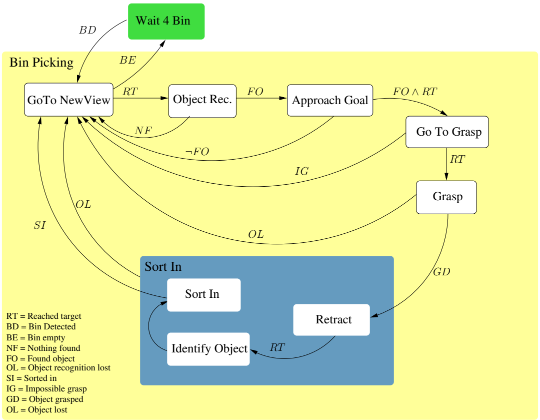


Fig. 15 Automaton for autonomous bin picking.

[5] The initial view and sort-in frames are taught in torque control with gravity compensation. This enables the user to freely move the robot to a desired configuration and save it in the application session.

execution. In case of failure, the robot recovers by monitoring conditions like object recognition dropouts, load losses, or impossibility of grasps.

4.4 Evaluation of Grasping Success

The efficiency and robustness of our approach was tested in a series of autonomous grasps. For this evaluation we replenished the bin (Fig. 11) after each successful grasp in order to have a filled bin and independent trials. On average, the robot needed 6.4 s for one grasping process, which comprises object detection from an arbitrary viewing position, approaching and grasping, unbagging, and moving back to the initial viewing position. The robot was able to grasp an object in every cycle for 80 trials, i.e. the overall cycle success rate was 100%. This result was only achievable due to the fault tolerance capabilities of the system along the entire process, such as the detection of a physical impossibility of a planned grasp, the non-successful grasp (overall 3 times), loosing an object in tracking, or localization without any result. The last fault was mainly caused by the fact that searched objects are often only partially in the field of view, so that the robot had to move to a new view position. All of these failure modes where detected or realized by the system and induced a restart of the grasping process. Consequently, the number of average views to recognize an object was $N_{\text{view}} = 2.2$.

4.5 Extension to Interactive Bin-Picking

Figure 16 describes our implementation of an interactive bin-picking demonstrator, merging the concepts for interaction and the autonomous capabilities of the robot. The initial entrance into the scene by the human is not shown, but is part of the demonstrator, i.e. it is assumed that the human has entered the scene, the "way into interaction" is completed, and the human is part of the process. ① shows the view into the bin and the corresponding object recognition (OR). Then, the robot grasps an object out of the bin ② and identifies it according to its weight, followed by a motion towards the human (GH) in ③. The "hand over" ④ then takes place, after which the robot waits (WT) for the human to complete his process ⑤. As soon as the human has finished, the robot receives the object in a visual servoing loop (VS) in ⑥. Now, the classified object is sorted into (SI) one of the trays ⑦ and the robot goes back to ①. ⑧ and ⑨ show how human-friendly (HF) behavior is an integrative part even in the presence of multiple humans. In ⑧ and ⑨ the tool surveillance and the physical contact during task execution are shown, respectively.

In summary, the system described here presents a versatile and robust solution with standard components for achieving safe and effective human-robot collaboration and a solution for the bin picking problem. Various explicitely non-trained test subjects were able to intuitively use the system.

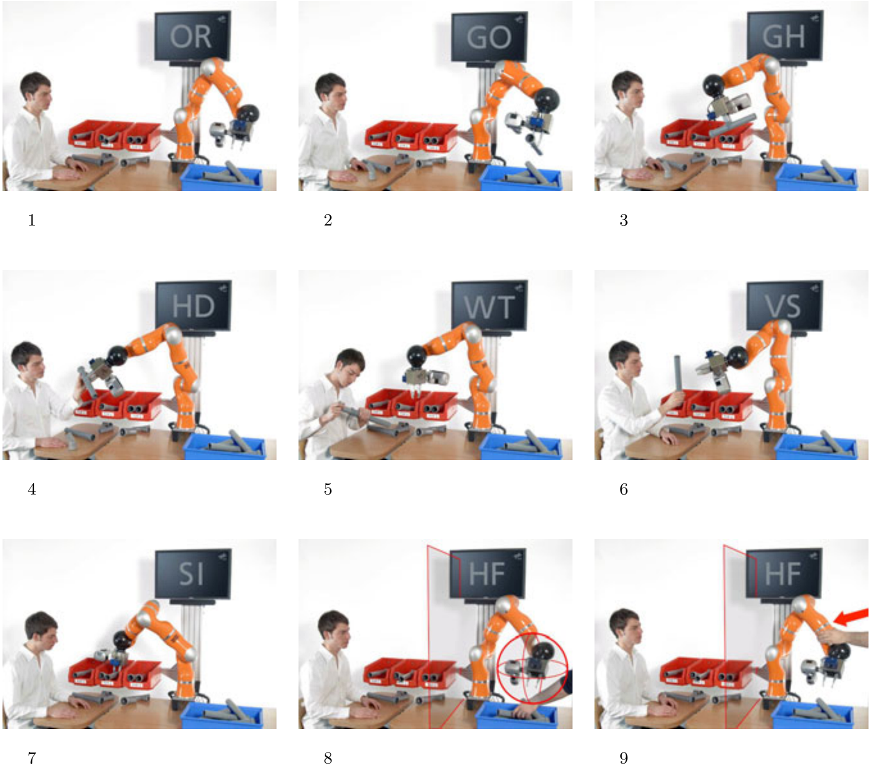


Fig. 16 Interactive bin picking.

5 Conclusion and Outlook

In this paper we proposed a general concept for the robotic co-worker and developed a prototype demonstration for validation based on commercially available technology. We outlined an integrative concept for combining soft-robotics concepts with multi-sensor vision schemes. Flexible hybrid automata can robustly and safely control the modalities of the co-worker in a partially known environment and especially handle the complexity as well as the necessary branching factor during the execution of the tasks. Based on our results in safe physical Human Robot Interaction we were able to effectively combine various control and motion schemes with vision sensing capabilities for the robot to effectively accomplish the task with sufficient safety. Furthermore, exteroceptive sensing is used in combination with compliance control for implementing industrially relevant autonomous tasks. The fusion of these concepts leads to high fault tolerance, supported by the results of the presented bin picking application. The thorough use of multi-sensor information enabled us to combine the proposed interaction and robust autonomy concepts needed for the **robotic co-worker**.

Future work will focus on broadening the interaction scenarios and incorporate cooperative tasks with joint assembly. For this purpose human surveillance and modeling will become a major focus in order to solve complex and dynamic joint processes. Furthermore, the extensive use of sufficiently fast motion planning is contemplated to reduce the "teaching by demonstration" amount.

Videos are provided at www.robotic.dlr.de/Sami.Haddadin/isrr2009.

Acknowledgment

This work has been partially funded by the European Commission's Sixth Framework Programme as part of the project PHRIENDS under grant no. 045359 and by the German Federal Ministry of Education and Research (BMBF) under Contract No. 16SV2298 as part of the project Lynkeus in cooperation with KUKA robotics, Augsburg. We would like to thank Simon Kielhöfer, Holger Urbanek, Xavier Giralt, Ralf Kühnemann, and Sven Parusel for their collaboration and for valuable contribution. Our special thanks go to Tilo Wüsthoff for taking the photographs of this paper.

References

1. Moshner, R.: From handiman to hardiman. Trans. Soc. Autom. Eng. 16, 588–597 (1967)
2. Yamada, Y., Konosu, H., Morizono, T., Umetani, Y.: Proposal of Skill-Assist: a system of assisting human workers by reflecting their skills in positioning tasks. In: IEEE International Conference on Systems, Man, and Cybernetics (SMC 1999), Tokyo, Japan, pp. 11–16 (1999)
3. Hirzinger, G., Heindl, J.: Sensor programming - a new way for teaching a robot paths and forces. In: International Conference on Robot Vision and Sensory Controls (RoViSeC3), Cambridge, Massachusetts, USA (1993)
4. Stemmer, A., Albu-Schäffer, A., Hirzinger, G.: An Analytical Method for the Planning of Robust Assembly Tasks of Complex Shaped Planar Parts. In: Int. Conf. on Robotics and Automation (ICRA 2007), Rome, Italy, pp. 317–323 (2007)
5. Yamada, Y., Hirasawa, Y., Huand, S., Umetani, Y.: Fail-Safe Human/Robot Contact in the Safety Space. In: IEEE Int. Workshop on Robot and Human Communication, pp. 59–64 (1996)
6. Zinn, M., Khatib, O., Roth, B.: A New Actuation Approach for Human Friendly Robot Design. Int. J. of Robotics Research 23, 379–398 (2004)
7. Bicchi, A., Tonietti, G.: Fast and Soft Arm Tactics: Dealing with the Safety-Performance Trade-Off in Robot Arms Design and Control. IEEE Robotics & Automation Mag. 11, 22–33 (2004)
8. Haddadin, S., Albu-Schäffer, A., Hirzinger, G.: Safe Physical Human-Robot Interaction: Measurements, Analysis & New Insights. In: International Symposium on Robotics Research (ISRR 2007), Hiroshima, Japan, pp. 439–450 (2007)

9. Haddadin, S., Albu-Schäffer, A., Hirzinger, G.: Safety Evaluation of Physical Human-Robot Interaction via Crash-Testing. In: Robotics: Science and Systems Conference (RSS 2007), Atlanta, USA, pp. 217–224 (2007)
10. Haddadin, S., Albu-Schäffer, A., De Luca, A., Hirzinger, G.: Collision Detection & Reaction: A Contribution to Safe Physical Human-Robot Interaction. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS 2008), Nice, France, pp. 3356–3363 (2008)
11. Kulic, D., Croft, E.A.: Affective State Estimation for Human-Robot Interaction. IEEE Transactions on Robotics 23(5), 991–1000 (2007)
12. Edsinger, A., Kemp, C.C.: Human-Robot Interaction for Cooperative Manipulation: Handing Objects to One Another. In: IEEE International Symposium on Robot & Human Interactive Communication (RO-MAN 2007), Jeju Island, Korea, pp. 1167–1172 (2007)
13. Dominey, P.F., Metta, G., Natale, L., Nori, F.: Anticipation and Initiative in Dialog and Behavior During Cooperative Human-Humanoid Interaction. In: IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS 2008), Daejeon, Korea, pp. 693–699 (2008)
14. Haddadin, S.: Towards the Human-Friendly Robotic Co-Worker. Master's thesis, Technical University of Munich (TUM) & German Aerospace Center (DLR) (May 2009)
15. Brock, O., Khatib, O.: Elastic Strips: A Framework for Motion Generation in Human Environments. Int. J. Robotics Research 21(12), 1031–1052 (2002)
16. Albu-Schäffer, A., Ott, C., Hirzinger, G.: A Unified Passivity-based Control Framework for Position, Torque and Impedance Control of Flexible Joint Robots. Int. J. of Robotics Research 26, 23–39 (2007)
17. Siciliano, B., Khatib, O. (eds.): Springer Handbook of Robotics. Springer, Heidelberg (2008)
18. Suppa, M., Kielhoefer, S., Langwald, J., Hacker, F., Strobl, K.H., Hirzinger, G.: The 3D-Modeller: A Multi-Purpose Vision Platform. In: Int. Conf. on Robotics and Automation (ICRA), Rome, Italy, pp. 781–787 (2007)
19. Hacker, F., Dietrich, J., Hirzinger, G.: A Laser-Triangulation Based Miniaturized 2-D Range-Scanner as Integral Part of a Multisensory Robot-Gripper. In: EOS Topical Meeting on Optoelectronic Distance/Displacement Measurements and Applications, Nantes, France (1997)
20. Strobl, K.H., Wahl, E., Sepp, W., Bodenmueller, T., Seara, J., Suppa, M., Hirzinger, G.: The DLR Hand-guided Device: The Laser-Stripe Profiler. In: Int. Conf. on Robotics and Automation (ICRA 2004), New Orleans, USA, pp. 1927–1932 (2004)
21. Suppa, M.: Autonomous robot work cell exploration using multisensory eye-in-hand systems. Ph.D. dissertation, Gottfried Wilhelm Leibniz Universität Hannover (2007)
22. Fuchs, S., Hirzinger, G.: Extrinsic and Depth Calibration of ToF-Cameras. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2008), Anchorage, USA, pp. 1–6 (2008)
23. Sepp, W., Fuchs, S., Hirzinger, G.: Hierarchical Featureless Tracking for Position-Based 6-DoF Visual Servoing. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS 2006), Beijing, China, pp. 4310–4315 (2006)

Robotic Manipulation and Humanoids

Cédric Pradalier

The main challenge that researchers interested in humanoid robotics used to have to solve was related to stability and walking. With progress in this field and the development of more complex humanoids, gripping and manipulation need to be addressed as well. On the other hand, research in the field of manipulation used to be a problem for industrial robotics and industrial arms. The convergence of these fields is hopefully illustrated by the collection of papers gathered in this part of the ISRR'09 proceedings.

- In "Cybernetic Human HRP-4C: A humanoid robot with human-like proportions", Kajita et al. provide an insight on the development of the new humanoid – or even human-like – robot HRP-4C. This work represents a tremendous effort not only in terms of engineering but also in terms of design.
- In "Enhanced Mother Environment with Humanoid Specialization in IRT Robot Systems", Inaba et al. present a design framework for the development of humanoid behaviours, from control tasks to perception and interaction. An important aspect of this system is its versatility and adaptability to various kinds of robots.
- In "A Factorization Approach to Manipulation in Unstructured Environments", Katz and Brock propose an approach to first identify the kinematic structure of objects by interacting with them, and then learn how to improve the efficiency of this identification using machine learning techniques. One of the important aspects of this paper is the demonstration of the benefit and need for robotic research to take advantage of all fields of computer science, computer vision, graph theory, control, machine learning, etc...
- In "Reconstruction and Verification of 3D-Object Models for Grasping", Marton et al. address the problem of fitting models to the perception of objects using a 3D laser scanner, with the intention of grasping them with a manipulator. The particularity of the approach is to represent 3D objects by a collection of primitive objects which are easier to detect and parametrize than the complete objects.
- In "Floating Visual Grasp of Unknown Objects Using an Elastic Reconstruction Surface", Vincenzo et al. propose an approach for visually guiding the finger of a robotic hand to the surface of an object to be picked

up. The originality of the approach lies in its iterative combination of a motion planning step and a step of visual estimation of the object's shape.

- In "Generality and Simple Hands", Mason et al. challenge the traditional concept that more complex robotic hands are required for efficient manipulation, in particular in industrial environment. They show that simple three-fingered hands are still very relevant for a wide variety of industrial tasks.
- In "From Sensory-motor Primitives to Manipulation and Imitation Strategies in Humanoid Robots", Asfour et al. summarise the state of the art in development of complex behaviours for humanoid robots in environment designed for humans. Some of the components of these applications are object recognition, grasping, manipulation, but they also include learning sensory-motor skills by imitation.

An underlying challenge reported by all these chapters is the management of complexity. Robotic systems and humanoids in particular are becoming more and more complex, both on the functionality level (hardware, sensors, software primitives) and the behaviour level. Handling uncertainties is already a difficult question, but the robotics community still needs to come up with a generic way to deal with the combinatorial explosion of complexity coming from the combination of more and more components and behaviours.

A Factorization Approach to Manipulation in Unstructured Environments

Dov Katz and Oliver Brock

Abstract. We propose factorization as a concept to analyze and solve manipulation problems in unstructured environments. A factorization is a decomposition of the original problem into factors (sub-problems), each of which can be solved much more easily than the original problem. The appropriate composition of these factors results in a robust and efficient solution to the original problem. Our assumption is that manipulation problems live in lower-dimensional subspaces of the high-dimensional state space associated with unstructured environments. A factorization identifies these subspaces and therefore permits finding simple and robust solutions to the factors. In this paper, we examine the effects of factorization in the context of our recent work on manipulating articulated objects in unstructured environments.

1 Introduction

Mobile manipulation in unstructured environments[1] remains an important challenge in robotics. Even after several decades of research, our ability to endow robotic systems with general manipulation skills remains limited. What is the key to making tangible progress in this domain?

In this paper, we hypothesize that fundamental progress in autonomous manipulation can only be achieved through an understanding of how to adequately compose simple perception, control, planning, and learning skills so that they incrementally realize increasingly complex manipulation behavior.

Dov Katz · Oliver Brock
Robotics and Biology Laboratory, School of Electrical Engineering and Computer Science,
Technische Universität Berlin, Germany

[1] When using the term "unstructured" we refer to environments that have not been modified to accommodate limitations of the robot. We consider providing *a priori* models to the robot as a way to accommodate these limitations.

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 285–300.
springerlink.com

This hypothesis may seem obvious. We believe, however, that it contrasts with some common believes and practices applied in much of the current research in robotics. Should the hypothesis prove to be correct, there would be important implications for how research in autonomous mobile manipulation should be conducted.

Within the manipulation community, some researchers argue that a break-through in manipulation will be triggered by better sensing technologies. Such advances, so the assumption, will lead to the availability of highly accurate models, even in unstructured environments. The problem of acquiring those models is therefore deferred to the sensor community and research proceeds under the assumption that accurate models are available. This view is commonly taken, for example, in motion planning and grasp planning. In contrast, we believe that the perceptual problems associated with obtaining adequate models for task execution will remain very challenging, irrespective of advances in sensor technology.

Another popular view is that the challenges of manipulation in unstructured environments may be addressed using ever-increasing computational power. Considering the recent successes of sampling-based motion planning and POMDP-based approaches to planning, it seems credible that soon we will be able to plan in complex environments while taking sensing and actuation uncertainties into account. In contrast, we believe that the combinatorial explosion associated with problems in unstructured environments will leave general planning for real-world environments out of our computational reach for some time to come.

Why then do we claim that advances in manipulation can be achieved through a suitable composition of perception, control, planning, mechanisms, etc.?

At a high level, our argument is about appropriate decompositions of high-dimensional state spaces. The goal of decomposition is to find sub-problems that can be solved easily and whose composition solves the original, more difficult problem. We refer to such a decomposition as a *factorization*, emphasizing that the decomposition leads to simpler components (factors) that, when combined (multiplied), solve the original problem (equal the product). As an example consider the expression $a^2 - 2ab + b^2$, which can be decomposed as $a^2(1 - \frac{2b}{a} + \frac{b^2}{a^2})$ or as $(a-b)(a-b)$. Clearly, both expressions are equivalent—they compute the same number (or achieve the same functionality)—but the latter one is much simpler and requires less computation. We only refer to the latter decomposition as a factorization.

Solving complex problems by decomposition is hardly a new idea. In fact, the fragmentation of robotics into sub-fields such as vision, control, planning, grasping, and manipulation represents a particular decomposition of the "robotics" problem. However, we believe that factorizations, i.e. "good" decompositions, do not naturally coincide with the boundaries imposed by the traditional academic sub-fields. Instead, we hypothesize that a factorization typically exploits synergies that arise when these very boundaries are crossed.

Factorization will enable progress in manipulation for two reasons. First, they lead to simple, efficient, and robust solution to manipulation problems, because factorizations identify the low-dimensional subspace of the high-dimensional state space within which the solution to the problem lies. Second, factorizations enable

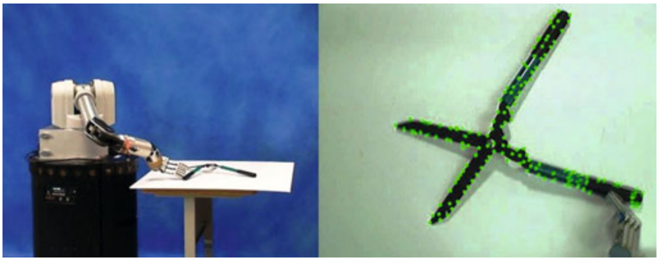


Fig. 1 UMan (UMass Mobile Manipulator) performs a manipulation task without prior knowledge about the manipulated object. The right image shows the scene as seen by the robot through an overhead camera; dots mark tracked visual features.

the *incremental* development of increasingly complex skills. Once the "right" factor has been split off, the remaining product itself becomes easier to factorize. Choosing a poor decomposition, however, may leave us with parts that are as hard to solve as the original problem.

Let us consider for instance the problem of grasping. Grasping is often decomposed into perception, planning, execution, and mechanism design. Planning methods assume accurate models and determine force closure based on this information. They have rarely, if at all, scaled to real-world environments. In contrast, consider the impressive real-world performance of the shape-deposition manufacturing (SDM) hand design by Dollar and Howe [9]. The hand has a single actuated degree of freedom and is able to robustly and repeatably pick up objects of greatly differing geometries, assuming the hand is positioned appropriately relative to the object.

A closer look at the decompositions used by these two approaches reveals why, in our view, one is much more successful than the other. The classical approach decomposes the problem along the boundaries of existing sub-fields into sensing to build an accurate model and planning a grasp. Both of these sub-problems have proved to be very difficult. Dollar and Howe pursue a different approach: they decompose grasp planning into determining a hand placement and closing the hand around the object. We believe that these two factors are much easier to solve than those of the classical decomposition, while achieving the same objective. Dollar and Howe show that the second factor (closing the hand to form a stable grasp) can be achieved easily for a variety of objects. They do so by leveraging compliance in the hand design. Compliance provides resilience to uncertainty and eliminates the necessity for complex perception and accurate models. The solution chosen by Dollar and Howe for the second factor thus has the potential of greatly facilitating the solution to the first factor—a sign of a good factorization. (A similar example of a good factorization is RHex [2], the biologically inspired hexapod robot. Both of these examples can be viewed as instances of the more general concept of morphological computation [27].)

In this paper, we evaluate our hypothesis in light of our recent work on manipulation in unstructured environments [16, 17, 18] (see Figure 1). We show that by translating our metaphor of factorization into a practical method for manipulation, a robot can autonomously obtain general domain knowledge for manipulation. Our work is preliminary and not intended as a conclusive validation of our hypothesis. However, we hope to be able to initiate a discussion about the most suitable way for making progress towards autonomous manipulation capabilities in unstructured environments.

2 Related Work

In our discussion of related work, we do not intend to survey research in manipulation (please refer to [6, 19, 26]). Instead, we examine the relationship of factorization to other areas of research and to prior work in robotics.

In the eighties, the psychologist Gibson [13, 14] questioned the separation of action and perception. He argued that perception is an active process and highly coupled with motor activities. Motor activities are necessary to perform perception—and perception is geared towards detecting opportunities for motor activities. He called these opportunities "affordances." This view of perception stands in contrast with the classical take on computer vision as "inverse optics", proposed by David Marr in 1982.

Gibson's theories continue to be relevant in psychology, cognitive science, and philosophy [24]. In a recent book, the philosopher-turned-cognitive-scientist Alva Noë describes an "enactive" approach to perception. He argues that perception is an embodied activity that cannot be separated from motor activities and that can only succeed if the perceiver possesses an understanding of motor activities and their consequences [23].

Similar "enactive" theories have been proposed for the development of cognitive capabilities [31, 12]. These "enactive" theories, be it in psychology, cognitive science, or philosophy, reject a functional separation of perception, thinking, and acting (as in sense, plan, act). Such a separation is at odds with experimental evidence in psychology and neuroscience and cannot hold up to the theoretical scrutiny of philosophers. This evidence might suggest that the development of advanced manipulation capabilities in the context of robotics will greatly benefit from the reorganization, and possibly the convergence, of existing sub-disciplines.

The trend towards eliminating the separation between perception, action, and cognition has long been present in the robotics community. Brooks' behavior-based robotics [8] exhibits conceptual parallels with the theory of behavioral psychology [25] (behaviorism). Both are, viewed simplistically, reactive paradigms. Based on this paradigm, behavior-based robotics already departs from the sense-plan-act paradigm and replaces it with hierarchies of reactive behavior [7], thereby overcoming the separation between action and perception.

Psychologists have criticized behaviorism as it does not account appropriately for the deliberate actions of an individual. The "enactive" perspective [31, 23] responds to this limitation by emphasizing the role these actions play in perception and cognition. It arrives at the conclusion that action, perception, and the associated cognitive processes cannot be separated from embodiment.

We believe Noë's theories lend support to our view that the simplest solution to manipulation in unstructured environments does not necessarily have to—or maybe even: must not—follow a strict separation between sensing, thinking and acting.

There is, of course, much work in robotics that is consistent with our hypothesis. Active vision [1] and visual servoing [15], for example, tightly couple perception and action (for a special set of skills). And we already discussed the use of compliance in embodiment [2, 9] to replace aspects of traditional computation with morphological computation [27], effectively crossing the boundary between embodiment and action.

The work of Edsinger and Kemp in mobile manipulation [11] also follows similar ideas. They "let the body do the thinking," an idea similar to morphological computation. They also emphasize the importance of task-relevant perception, a consequence of close coupling between action and perception.

There are also a number of ongoing mobile manipulation projects that in our view proceed along a different direction. These projects demonstrate impressively what robotic systems can accomplish today, based on the integration of technologies that has been developed within the boundaries of existing sub-fields. Among them are the STAIR project at Stanford University [3], El-E at Georgia Tech [21, 22], and HERB at Intel/CMU [4, 5]. All of these projects share one goal: they want to develop robots that can perform manipulation tasks in everyday environments. Whether the right path towards that goal will prove to be the integration of existing technologies or the factorization of specific manipulation problems remains an open question.

3 Factorizing a Manipulation Skill

We now present a case study of factorization for manipulation skills in unstructured environments. The specific skill we are interested in concerns the manipulation of articulated objects. To reflect the fact that the robot operates in an unstructured environment, it initially has no specific knowledge about the objects it interacts with. The robot plays with an articulated object until it has understood the object's kinematic structure. Based on the acquired information, the robot then manipulates the object into a given configuration. In the current scenario, illustrated in Figure 1, we restrict the class of objects to planar kinematic chains, as the ones shown in Figure 2.

The ability to manipulate kinematic objects is elementary for a wide range of manipulation tasks (all prehensile manipulation tasks with rigid objects). We therefore believe that the skill discussed here can serve as a sensorimotor foundation for more complex manipulation tasks. We believe this manipulation skill represents a "good" factor and will therefore facilitate the factorization of the other, more complicated factors, i.e. it will be useful for the development of more complex capabilities. This, of course, remains to be demonstrated in future research.

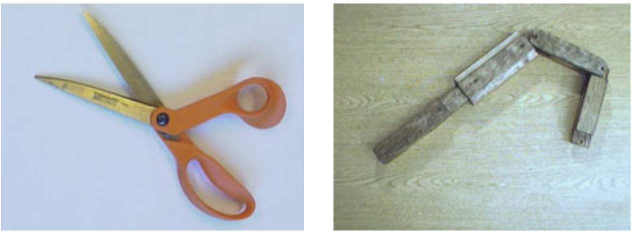


Fig. 2 Two examples of kinematic structures: scissors with a single revolute joint and a wooden toy with a prismatic joint and two revolute joints.

3.1 Action and Perception

Determining the kinematic structure of a planar articulated object is difficult based on visual clues alone. It is equally difficult based on haptic interactions alone. When using visual clues and interactive abilities together, however, the task becomes very simple.

The key idea is simple: we use the embodiment of the robot to create a visual signal that facilitates the identification of the kinematic structure of articulated objects. This means that the robot pushes the object and observes the resulting changes in the scene (Figure 1). The observation consists of tracking the motion of visual features in the scene. The motion of these features can be analyzed to determine the kinematic structure of the articulated object and the approximate extent of the links.

The first step of our algorithm analyzes the motion of features to identify all rigid bodies observed in the scene. The algorithm builds a graph $G(V,E)$ from the feature trajectories obtained throughout the interaction. Every vertex $v \in V$ in the graph represents a tracked image feature. An edge $e \in E$ connects vertices (v_i, v_j) if and only if the distance between the corresponding features remains smaller than some threshold throughout the observed interaction. Features on the same rigid body are expected to maintain approximately constant distance between them throughout the entire observation. In the resulting graph, all features that lie on a single rigid body form a highly connected component (see Figure 3). To separate the graph into these components we use the min-cut algorithm. Identifying the highly connected sub-graphs is analogous to identifying the object's different rigid bodies.

The min-cut algorithm we use has worst case complexity of $O(nm)$, where n represents the number of nodes in the graph and m represents the number of clusters [20]. Most objects possess only few joints, making $m \ll n$. Thus, for practical purposes, we consider clustering to be linear in the number of tracked features.

This procedure of identifying rigid bodies is robust to the noise present in the feature trajectories. Unreliable features randomly change their relative distance to other features. This behavior places such features in small clusters, most often of size one. In our algorithm, we discard connected components with three of fewer features. This is a very simple and effective way to filter out sensor and tracking noise. The remaining connected components consist of features that were tracked reliably throughout the entire interaction. Each of these components corresponds to a rigid body in the scene.

The second step of our algorithm identifies the kinematic relationship among rigid bodies. We will discuss this for revolute joints, prismatic joints and other algorithmic aspects of the method are presented in detail in [16]. To find revolute joints, our algorithm examines all pairs of rigid bodies identified in the previous step. Based on their relative motion, it classifies their kinematic relationship as either revolute, prismatic, or disconnected.

To find revolute joints, we exploit the information captured in the graph *G*. Vertices that belong to one connected component must have maintained constant distance from all vertices in their cluster. This property holds for features on or near revolute joints, connecting two or more rigid bodies. To find all revolute joints, we simply search the entire graph for vertices that belong to two clusters (see Figure 3).

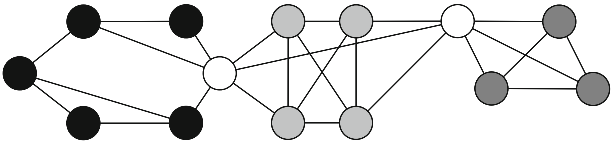


Fig. 3 Graph for an object with two revolute degrees of freedom. Highly-connected components (shades of gray) represent the links. Vertices of the graph that are part of two components represent revolute joints (white vertices).

After all pairs of rigid bodies represented in the graph have been considered, our algorithm has determined appropriate explanations for their relative motions. Using this information, we build a kinematic model of the object using Denavit-Hartenberg parameters. This is illustrated for a real-world object in Figure 4. Note that both the tool and the table have wood texture and color, making the vision problem difficult for any color- or texture-based algorithm.

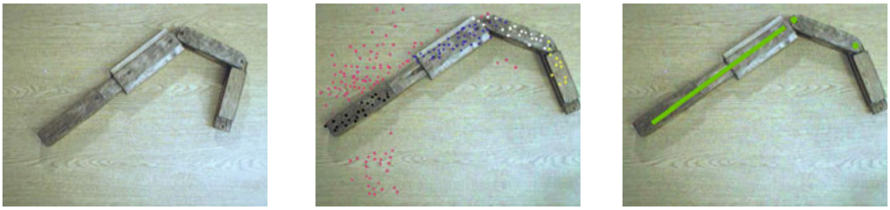


Fig. 4 Experimental results from [16] showing the extraction of the kinematic properties of a wooden toy (length: 90cm) using interactive perception: The left image shows the object in its initial pose. The middle image shows the object after the interaction. The color-coded clusters of visually tracked features correspond to the rigid bodies of the toy. The right image shows the detected kinematic structure (line marks the prismatic joint, dots mark the revolute joints).

The robustness of this skill was demonstrated in dozens of real-world experiments. The algorithm makes no assumptions about the kinematic structure of objects in the scene (except for the kinematic structure being planar); it can handle serial chains as well as planar branching mechanisms and kinematic loops. The algorithm does not require prior knowledge of the objects, is insensitive to lighting conditions and specularities, succeeds irrespective of the texture and color of the object's parts, works reliably even with low-quality video, and is computationally efficient. At the same time, the algorithmic components of this interactive perception skill are very basic (feature tracking, pushing, graph min cut). Nevertheless, the "right" composition of these simple ingredients results in an interactive manipulation skill for unstructured environments that is extremely robust and computationally efficient.

3.2 Learning Effective Interactive Perception

So far we assumed that the robot's interactions with objects in the environment were scripted. Now we show how a robot can learn how to interact with its environments in the most effective manner. In the process of performing a number of such self-observed interactions, the robot gathers domain knowledge and is able to use this knowledge to extract complete kinematic models with fewer and fewer interactions.

To enable learning in the domain of planar articulated objects, we capture the robot's experience in a relational representation. This representation is critical to the success of our learning-based approach to manipulation. Using a finite set of relations, we describe an infinite number of states and actions. It thus becomes feasible to represent and reason about situations that a propositional representation cannot handle. For example, a robot may encounter many types of scissors, varying in color, shape, and size. All scissors, however, have the same kinematic structure. A single relational formula can capture this structure for *all* scissors, irrespective of other physical characteristics. Therefore, a single relational action can be applied to *all* such objects (see Figure 5). Furthermore, experience gathered with one object can be applied to *all* objects that contain the same kinematic substructure. Propositional representations, in contrast, require a proposition for every link in the kinematic structure, and one for every action. The relational representation avoids this combinatorial explosion and makes learning possible.

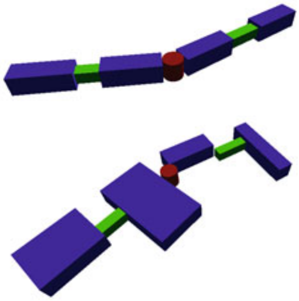


Fig. 5 Two objects with different physical properties but identical kinematic structure: because their relational representation is identical, experience acquired with one can be used directly to interact with the other

Our relational representation for kinematic models of articulated objects captures links, link properties, and kinematic relationships between links. Figure 2 shows two examples of planar kinematic structures. The scissors have a single revolute degree of freedom and the wooden toy is a serial kinematic chain with a prismatic joint (on the left of the figure) and two revolute joints. Our relational representation uses predicates $R(\cdot)$, $P(\cdot)$, and $D(\cdot)$ to describe that rigid bodies are connected by a revolute joint, a prismatic joint, or are disconnected, respectively.

The predicates are n-ary, with $n \geq 2$, to capture branching kinematic structures. The rigid body passed as the first argument to the relation is the one in relationship with all other arguments. For example, $R(x,y,z)$ is equivalent to $R(x,y) \wedge R(x,z)$. Using these relations, we can represent the kinematic structure of the scissors as $D(l_b, R(l_1,l_2))$, where l_1 and l_2 represent the two links of the scissors and l_b is a disconnected background link. The kinematic structure of the wooden toy if Figure 2 can be represented as $D(l_b, R(l_4, R(l_3, P(l_1,l_2))))$. Note that this representation is not unique. The wooden toy could also be represented as

$$D(P(l_4, R(R(l_1,l_2),l_3)), l_b).$$

Which of these representations is used by the robot depends on the order of discovery of the links. The most deeply nested relation is discovered first.

By extending our atomic representation of links to m-ary relations $L(\cdot)$, $m \geq 1$, we can include link properties in our description of kinematic chains. We will limit ourselves to a single property, the size of the link. The wooden toy can now be represented as

$$D(l_b, R(L(s,f_4), R(L(s,f_3), P(L(s,f_1), L(s,f_2))))),$$

where s stands for the property *small* and the sets of visual features f_i spatially identify links in the physical world. The extension to an arbitrary number of link properties is straightforward.

We also use a relational representation for the actions performed by the robot. Actions apply pushing or pulling forces to one of the links. The forces can be applied along the major axes of the link or along a forty-five degree angle to the major axes. An action is represented as $A(L(\cdot), \alpha)$, where $L(\cdot)$ represents a link and $alpha$ is an atom describing one of the possible six pushing/pulling directions relative to the link.

Based on this relational representation, we cast the incremental acquisition of kinematic representations of objects as a relational reinforcement learning [10, 28, 30] problem. We define a Relational Markov Decision Process (RMDP) [30] and then apply Q-learning [32] to find an optimal policy.

A Markov Decision Process (MDP) is a tuple $M = (S, A, T, R)$, where S designates the set of possible states, A is the set of actions available to the robot, $T : S \times A \rightarrow \Pi(S)$ specifies a state transition function to determine a probability distribution $\Pi(S)$ over S, indicating the probability of attaining a successor state when an action is performed in an initial state, and $R : S \times A \rightarrow \mathbb{R}$ is a function to determine the reward obtained by taking a particular action in a particular state. In our case, the description of states and actions is relational and therefore we have a relational MDP. The details of this MDP and the learning algorithm are given in reference [17].

The robot remembers each of its experiences from interactions with the world by storing a tuple $E(s,a,r)$ of state s, action a, and the Q-value, or reward, r obtained when performing the action in that state. Because states and actions are relational and stored un-instantiated, every stored experience describes a possibly infinite number of experiences. These experiences serve as an instance-based representation of the Q-value function.

Our relational representation of experiences permits the robot to leverage past experience, even if it has not previously visited the exact same state. Given the current state, the robot retrieves the best action based on its experience of the most similar previously encountered state. Similarity between states is determined by considering the state's kinematic structure and the properties of the links in that structure. Neither of these aspects have to match perfectly for the robot to retrieve relevant experience.

We define a similarity measure using unification for approximately matching link properties and structure, specifically sub-graph mono-morphism [29], for identifying partial matches in kinematic structure between the current state and the robot's prior experience. The details of the similarity measure are given in reference [17].

By combining the interactive perception skill described in the previous section with this learning framework, the robot can learn to extract kinematic models with increasing effectiveness. It continuously interacts with articulated objects in the environment, stores its experiences, and remembers which actions lead to the discovery of new rigid bodies and their kinematic relationships. This experience, in turn, is used to guide further interactions.

To demonstrate the effectiveness of our learning-based approach to manipulation in unstructured environments, we perform two types of experiments. First, we show that our approach permits the learning of manipulation knowledge from experience. Second, we show that the acquired experiences transfer to previously unseen objects.

Our experimental evaluation requires a large number of experiments. For practical reasons, we performed these experiments in a simulated environment. Due to the robustness of the perceptual skill described in Section 3.1 and due to the simplicity of force guided pushing required for our experiments, we argue that our results remain valid in real-world experiments. Our simulation environment is based on the Open Dynamics Engine (ODE), a dynamics simulator. The simulation includes gravity, friction, and non-determinacy.

In each experiment, the robot interacts with an articulated object to extract its kinematic structure. Example objects are given in Figures 5, 6, and 7. Revolute joints are shown as red cylinders, prismatic joints are represented by green boxes, and links are shown in blue. We only report on experiments with serial chains, even though we have successfully experimented with branching mechanisms and kinematic loops. Perceptual information about the manipulated objects is obtained from a simulation of the perceptual skill described in Section 3.1 [16]. We do not use the simulator's internal object representation to obtain information about the object.

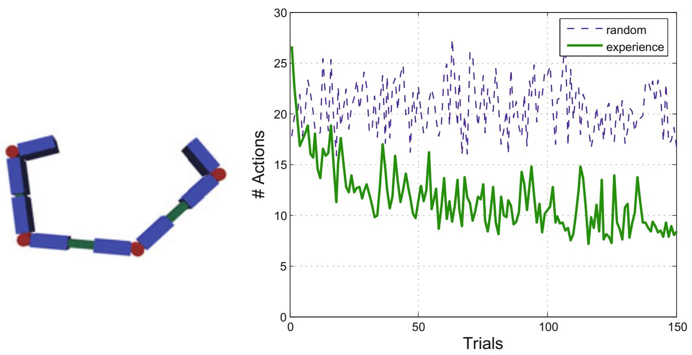
Each experiment consists of a sequence of trials. For each trial we report the average over 10 independent experiments. A trial consists of a number of steps; in each step, the robot applies a pushing action to the articulated object. The trial ends when an external observer signals that the obtained model accurately reflects the kinematic structure of the articulated object. The number of steps required to uncover the correct kinematic structure measures the effectiveness with which the robot accomplishes the task.

Each step of a trial can be divided into three phases. In the first phase, the robot selects an action and a link with which it wants to interact. The action is instantiated using the current state and the experience stored in the representation of the Q-value function. In the second phase, the selected action is applied to the link, and the ODE simulator generates the resulting object motion. The trajectories of the visual features tracked by the perception skill are reported to the robot. In the last phase, the robot analyzes the motion of visual features and determines the kinematic properties of the rigid bodies observed so far. These properties are then incorporated into the robot's current state representation. With each step, the robot accumulates manipulation experiences that improves its performance over time.

A trial ends when the kinematic model obtained by the robot corresponds to the structure of the articulated object. In our simulation experiments, an external supervisor issues a special reward signal to end the particular trial. Note that such a supervisor is not required for real-world experiments. The robot can decide to perform manipulation based on incomplete information. If new kinematic information is discovered during manipulation, the robot simply updates its kinematic model and revises its manipulation strategy.

To demonstrate the ability of the proposed learning framework to acquire relevant manipulation knowledge, we observe the number of actions required to discover a kinematic structure. We compare the performance of the proposed grounded relational reinforcement learning approach to a random action selection strategy, using an object with seven degrees of freedom and eight links (Fig. 6(a)). The resulting learning curve is shown in Figure 6(b). Random action selection, as to be expected, does not improve its performance with additional trials. In contrast, action selection based on the proposed relational reinforcement learning approach results in a substantial reduction in the number of actions required to correctly identify the kinematic structure. This improvement already becomes apparent after about 20 trials. Using the learning-based strategy, an average of 8 pushing actions is required to extract the complete kinematic model, compared to the approximately 20 pushing actions required with random action selection. This corresponds to an improvement of about 60%.

This first experiment demonstrates that our approach to manipulation enables robots to acquire manipulation knowledge and to apply this knowledge to improve manipulation performance. To demonstrate that the manipulation experience acquired with one object transfers to other objects, we perform two additional experiments in which we observe the number of actions required to discover a kinematic structure with and without prior experience.



(a) Articulated object (b) Performance compared to random strategy

Fig. 6 Experiments with a planar kinematic structure with seven degrees of freedom (RPRPRPR, R = revolute, P = prismatic). The learning curve for our learning-based approach to manipulation (green solid line) converges to eight required actions with a decreasing variance, representing an improvement of 60% over the random strategy (blue dashed line).

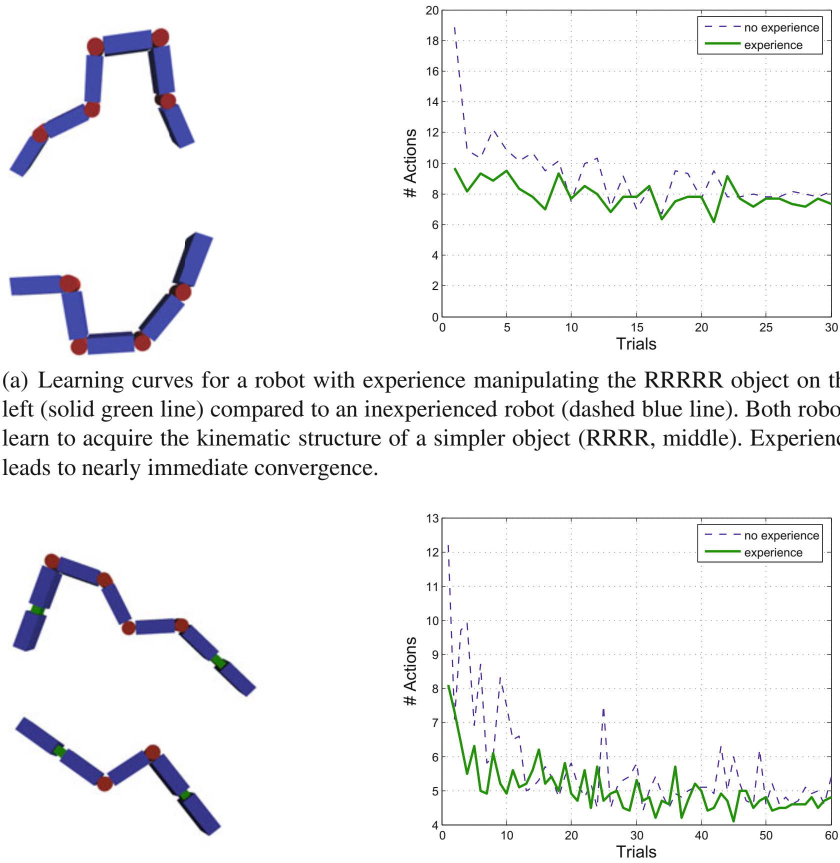
In the first experiment, the robot learns to manipulate a complex articulated object with 5 revolute joints. After 50 trials, the robot is given a slightly simpler structure that only possesses four revolute joints. The simpler structure is a sub-structure of the more complex one. We compare the robot's performance after these initial 50 trials to another robot's performance without prior experience (see Fig. 7(a)). Given prior experience, the robot achieves convergence almost immediately. This corresponds to a performance improvement of about 50% in the first trial, compared to the robot without experience. After about ten trials, both robots achieve similar performance, which is to be expected for simple structures that exclusively consist of revolute joints.

In the second experiment, the robot learns to manipulate an articulated object with 6 degrees of freedom (see Fig. 7(b)). After 50 trials, the robot is given a different structure that is not a substructure of the other. We compare the robot's performance after these initial 50 trials to another robot's performance without prior experience (see Fig. 7(b)). Again, experience results in a much faster convergence (after only five trials) towards about five required interactions. In addition, the variance of successive trials is reduced. After about 15 trials, both robots converge towards the same number of interactions.

4 Effects of Factorization

How does our approach for extracting planar kinematic models from articulated objects in unstructured environments relate to the concept of factorization?

The interactive perception skill described in Section 3.1 fuses action and perception into a single framework. In this framework, the perception of kinematic



(a) Learning curves for a robot with experience manipulating the RRRRR object on the left (solid green line) compared to an inexperienced robot (dashed blue line). Both robots learn to acquire the kinematic structure of a simpler object (RRRR, middle). Experience leads to nearly immediate convergence.

(b) Learning curves for a robot with experience manipulating the PRRRRP object on the left (solid green line) compared to an inexperienced robot (dashed blue line). Both robots learn to acquire the kinematic structure of a simpler object (PRRP, middle). The simpler object is **not** a sub-structure of the complex object. With experience, convergence is achieved in about five trials.

Fig. 7 Experimental validation of transfer of manipulation experience between different articulated objects.

structures (model acquisition) is enabled through manipulation of the world. At the same time, the successful manipulation of kinematic chains depends on these very perceptual capabilities. As the robot manipulates kinematic structures, it continuously observes the joint angles of the object. It can use this information to complete the manipulation task, i.e. to move the object from its current configuration into a goal configuration, even in the presence of uncertainty. The synergistic combination of action and perception reflects a factorization that leads to a simple and robust skill for unstructured environments.

This stands in contrast with the traditional view of robotics, in which the problem of manipulating articulated bodies in unstructured environments would be decomposed into model acquisition and manipulation. We believe that this is an inappropriate decomposition, resulting in an overly difficult perceptual problem that to our knowledge has not been solved yet.

In our approach, the effects of factorization go beyond action and perception. They extend to the robot's world model, dividing it into an internal and an external part (the world).[2] Traditionally, the task of acquiring information about the world and acting on that information are separated. Here, model acquisition and manipulation are fused into a single framework. The robot can act on its incomplete and possibly inaccurate internal model. Through deliberate interactions, additional information can be obtained from the world. The robot continuously incorporates this new information into its internal model. This integration of action, perception, and model acquisition in our factorization thus contributes to the robustness of our manipulation skill in unstructured environments.

Furthermore, the chosen factorization enables the robot to learn and subsequently apply domain-specific manipulation knowledge. The effectiveness of learning in a symbolic, relational domain directly depends on the relationship between the symbols and the sensorimotor capabilities of the robot. For example, one could pick very low-level symbols and perform learning using basic visual features. This would result in a simple perceptual task, putting most of the complexity into the learning task. We believe that this is an inappropriate decomposition. In contrast, we choose symbols that are directly grounded in *task-specific* sensorimotor capabilities of the robot. The perceptual problem now consists of identifying kinematic degrees of freedom. The learning problem derives its simplicity from the resulting relational description of the physical world. This factorization appropriately distributes the complexity of the overall problem. The result is a robust and efficient approach to the manipulation of articulated objects in unstructured environments.

5 Conclusion

We examined the hypothesis that manipulation problems in unstructured environments must be addressed by a suitable composition of capabilities in the areas of perception, action, learning, and model acquisition. We argued that such a composition can only be found if the boundaries between the traditionally established sub-fields in robotics are ignored. Ignoring these boundaries makes it possible to decompose manipulation problems into sub-problems that can be solved effectively and re-composed to robustly solve the original problem. We refer to decompositions that satisfy this requirements as factorizations.

To support our hypothesis, we presented and analyzed a manipulation skill for planar articulated objects. We argued that the decomposition of the manipulation problem reflected in this skill tightly integrates perception, action, learning, and model acquisition. We view the robustness and effectiveness of this skill in unstructured environments as initial evidence that factorization is a good conceptual framework to guide research in this area. We hope that our arguments will initiate

[2] Experimental evidence shows that the humans perceptual system greatly relies on the physical world as part of its world model [23].

a discussion about the most appropriate approach to manipulation in unstructured environments. We are convinced that the reasoning presented in this paper in the context of a single task will extend to other tasks and aspects of robotics in unstructured environments.

Should the concept of factorization prove to be indeed an important enabler of progress for manipulation in unstructured environments, we believe there might be interesting implications. For one, it would seem appropriate to shift emphasis in robotics research from developing narrow, high-performance systems to building robust, versatile, and integrated systems with lower-level capabilities that can be brought to bear in a variety of problem domains. Furthermore, it would indicate that progress towards truly autonomous robots can most effectively be made by focusing on building up the competency of these integrated systems incrementally, starting with very basic skills, such as the one presented here, rather than by integrating best-of-breed approaches to individual facets of a real-world problem.

Acknowledgments

We gratefully acknowledge the generous support of the Alexander von Humboldt Foundation through the Alexander von Humboldt Professorship, financed by the German Federal Ministry of Education and Research (BMBF). We are equally grateful for support by the National Science Foundation (NSF) under grants CNS-0454074 and IIS-0545934.

References

1. Aloimonos, J., Weiss, I., Bandyopadhyay, A.: Active vision. International Journal of Computer Vision 1, 333–356 (1988)
2. Altendorfer, R., Moore, N., Komsuoglu, H., Buehler, M., Brown Jr., H.B., McMordie, D., Saranli, U., Full, R., Koditschek, D.E.: RHex: A Biologically Inspired Hexapod Runner. Autonomous Robots 11(3), 207–213 (2001)
3. Ashutosh Saxena, A.Y.N., Driemeyer, J.: Robotic grasping of novel objects using vision. International Journal of Robotics Research 27(2), 157–173 (2008)
4. Berenson, D., Srinivasa, S.: Grasp synthesis in cluttered environments for dexterous hands. In: Proceedings of the IEEE International Conference on Humanoid Robots (December 2008)
5. Berenson, D., Srinivasa, S., Ferguson, D., Kuffner, J.: Manipulation planning on constraint manifolds. In: Proceedings of the IEEE International Conference on Robotics and Automation, ICRA (May 2009)
6. Bicchi, A., Kumar, V.: Robotics grasping and contact: A review. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), San Francisco, CA (2000)
7. Brooks, R.A.: A robust layered control system for a mobile robot. International Journal of Robotics and Automation RA-2(1), 14–23 (1986)
8. Brooks, R.A.: Intelligence without reason. Proceedings of the International Joint Conference on Artificial Intelligence 1, 569–595 (1991)
9. Dollar, A.M., Howe, R.D.: The sdm hand as a prosthetic terminal device: A feasibility study. In: Proceedings of the 2007 IEEE International Conference on Rehabilitation Robotics (ICORR), Noordwijk, Netherlands (2007)

10. Džeroski, S., de Raedt, L., Driessens, K.: Relational reinforcement learning. Machine Learning 43(1-3), 7–52 (2001)
11. Edsinger, A., Kemp, C.C.: Manipulation in human environments. In: Proceedings of the IEEE International Conference on Humanoid Robots (2006)
12. Gallagher, S.: How the Body Shapes the Mind. Oxford University Press, Oxford (2005)
13. Gibson, J.J.: The Senses Considered as Perceptual Systems. Greenwood Press, Westport (1983) (reprint)
14. Gibson, J.J.: The Ecological Approach to Visual Perception. Lawrence Erlbaum, Mahwah (1986)
15. Hutchinson, S., Hager, G., Corke, P.: A tutorial on visual control. IEEE Transactions on Robotics and Automation 12, 651–670 (1996)
16. Katz, D., Brock, O.: Manipulating articulated objects with interactive perception. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Pasadena, USA (2008)
17. Katz, D., Pyuro, Y., Brock, O.: Learning to manipulate articulated objects in unstructured environments using a grounded relational representation. In: Proceedings of Robotics: Science and Systems IV, Zurich, Switzerland (June 2008)
18. Kenney, J., Buckley, T., Brock, O.: Interactive segmentation for manipulation in unstructured environments. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Kobe, Japan (2009)
19. Mason, M.T.: Mechanics of Robotic Manipulation. MIT Press, Cambridge (2001)
20. Matula, D.W.: Determining edge connectivity in O(mn). In: Proceedings of the 28th Symp. on Foundations of Computer Science, pp. 249–251 (1987)
21. Nguyen, H., Anderson, C., Trevor, A., Jain, A., Xu, Z., Kemp, C.C.: El-E: An assistive robot that fetches objects from flat surfaces. In: HRI Workshop on Robotic Helpers: User Interaction Interfaces and Companions in Assistive and Therapy Robots (2008)
22. Nguyen, H., Jain, A., Anderson, C., Kemp, C.C.: A clickable world: Behavior selection through pointing and context for mobile manipulation. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS (2008)
23. Noë, A.: Action in Perception. MIT Press, Cambridge (2004)
24. Noë, A., Thompson, E. (eds.): Vision and Mind: Selected Readings in the Philosophy of Perception. MIT Press, Cambridge (2002)
25. O'Donohue, W., Kitchener, R. (eds.): Handbook of Behaviorism. Academic Press, London (1998)
26. Okamura, A.M., Smaby, N., Cutkosky, M.R.: An overview of dexterous manipulation. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), San Franscisco, USA, vol. 1, pp. 255–262 (2000)
27. Pfeifer, R., Iida, F.: Morphological computation: Connecting body, brain and environment. Japanese Scientific Monthly 58(2), 48–54 (2005)
28. Tadepalli, P., Givan, R., Driessens, K.: Relational reinforcement learning: An overview. In: Proceedings of the Workshop on Relational Reinfocement Learning at ICML 2004, Banff, Canada (2004)
29. Ullmann, J.R.: An algorithm for subgraph isomorphism. Journal of the ACM 23(1), 31–42 (1976)
30. van Otterlo, M.: A survey of reinforcement learning in relational domains. Technical Report TR-CTIT-05-31, Department of Computer Science, University of Twente, The Netherlands (July 2005)
31. Varela, F.J., Thompson, E., Rosch, E.: The Embodied Mind. MIT Press, Cambridge (2003)
32. Watkins, C.J.C.H., Dayan, P.: *Q*-learning. Machine Learning 8(3-4), 279–292 (1992)

Cybernetic Human HRP-4C: A Humanoid Robot with Human-Like Proportions

Shuuji Kajita, Kenji Kaneko, Fumio Kaneiro, Kensuke Harada,
Mitsuharu Morisawa, Shin'ichiro Nakaoka, Kanako Miura, Kiyoshi Fujiwara,
Ee Sian Neo, Isao Hara, Kazuhito Yokoi, and Hirohisa Hirukawa

Abstract. Cybernetic human HRP-4C is a humanoid robot whose body dimensions were designed to match the average Japanese young female. In this paper, we explain the aim of the development, realization of human-like shape and dimensions, research to realize human-like motion and interactions using speech recognition.

1 Introduction

Cybernetics studies the dynamics of information as a common principle of complex systems which have goals or purposes. The systems can be machines, animals or a social systems, therefore, cybernetics is multidiciplinary from its nature. Since Norbert Wiener advocated the concept in his book in 1948[1], the term has widely spreaded into academic and pop culture. At present, cybernetics has diverged into robotics, control theory, artificial intelligence and many other research fields, however, the original unified concept has not yet lost its glory.

Robotics is one of the biggest streams that branched out from cybernetics, and its goal is to create a useful system by combining mechanical devices with information technology. From a practical point of view, a robot does not have to be humanoid; nevertheless we believe the concept of cybernetics can justify the research of humanoid robots for it can be an effective hub of multidiciplinary research.

WABOT-1, the world first humanoid robot developed by Kato and his colleagues in 1973[2], was built as an integrated system having two arms, two legs, a voice

Shuuji Kajita · Kenji Kaneko · Fumio Kanehiro · Kensuke Harada · Mitsuharu Morisawa ·
Shin'ichiro Nakaoka · Kanako Miura · Kiyoshi Fujiwara · Ee Sian Neo · Isao Hara ·
Kazuhito Yokoi · Hirohisa Hirukawa
AIST, 1-1-1 Umezono, Tukuba, Ibaraki, Japan
e-mail: {s.kajita,k.kaneko,f-kanehiro,kensuke.harada,m.morisawa,
s.nakaoka,kanako.miura,k-fujiwara,rio.neo,isao-hara,
Kazuhito.Yokoi,hiro.hirukawa}@aist.go.jp

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 301–314.
springerlink.com

recognition system etc. Later, their group developed a piano playing humanoid, WABOT-2 in 1985[3].

In the early 1990s, Brooks and his colleague started to build humanoid robots as physical embodiment of artificial intelligence[4]. Research activities in ATR[5] and the RobotCub project[6] can be considered to have the same intention, namely humanoid robotics for cognitive science.

Since the breakthrough on biped walking done by Hirai et al.[7], many research projects on biped humanoid robots were conducted. Research was carried on generation of walking motion[8], jogging and running [9, 10], efficient walking[11], human-like walking[12], dynamic whole body balance control[13] and so forth.

There also exists research working on facial expression of humanoid robots [14, 15, 16, 17]. These works target social communication and interaction, which is another aspect of cybernetics.

In AIST, we have been developing a series of humanoid robots[19, 20]. Cybernetic human HRP-4C is our latest development, which is a humanoid robot designed to have body dimensions close to average Japanese young female(Fig.1, Table 1). This paper explains the goal of our project and introduce the developed robot system.

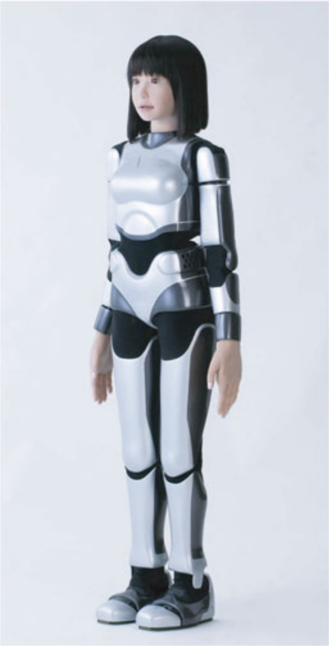


Fig. 1 Cybernetic human HRP-4C

Table 1 Principal specifications of HRP-4C

Height		1,580 [mm]
Weight(with batteries)		43 [kg]
Total DOF		42 DOF
	Face	8 DOF
	Neck	3 DOF
	Arm	6 DOF × 2
	Hand	2 DOF × 2
	Waist	3 DOF
	Leg	6 DOF × 2
CPUs	Motion controller	Intel Pentium M 1.6GHz
	Speech recognition	VIA C7 1.0GHz
Sensors	Body	Posture sensor
	Sole	6-axies force sensor × 2
Batteries		NiMH

2 Aim of the Development

2.1 User Centered Robot Open Architecture

HRP-4C was developed in the User Centered Robot Open Architecture (UCROA) Project which is one of the projects under the AIST Industrial Transformation Research Initiative, a 3-year industry-academia joint project implemented by AIST

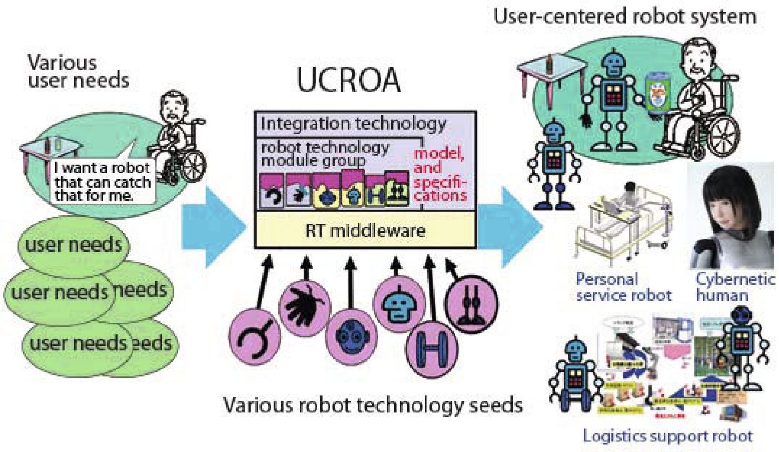


Fig. 2 User Centered Robot Open Architecture (UCROA)

from the fiscal year of 2006. The goal of the UCROA is to show society that it is possible to develop robot products that meet the specifications required by users combining reusable core technologies and developing prototype next-generation robots which can be used in practice by 2010 and for which the market is expected to be large. Three prototype robots were developed in the project (Fig.2).

1. Logistics support robot
2. Personal service robot
3. Cybernetic human

For the logistics support robot, we developed a robotic system for warehouses. For the personal service robot, a small manipulator system to assist daily life of handicapped people was developed. The third subject of the UCROA is the cybernetic human, whose concept is explained in the next subsection.

2.2 Concept of Cybernetic Human

A humanoid robots attract many people, because of its human-like appearance and behavior. Although a robot with very human-like appearance is called an "android" in general, we coined a new term *Cybernetic human* to define a humanoid robot with the following features.

1. Have the appearance and shape of a human being
2. Can walk and move like a human being
3. Can interact with humans using speech recognition and so forth

Such robots can be used in the entertainment industry, for example, exhibitions and fashion shows. It can be also used as a human simulator to evaluate devices for humans.

As the successor of our previous humanoid robots HRP-2 and HRP-3[19, 20], we call our new humanoid robot HRP-4C, "C" stands for cybernetic human.

3 Realization of Human-Like Shape and Dimensions

3.1 Target Specifications

To determine the target shape and dimensions of HRP-4C, we used the anthropometric database for Japanese population, which was measured and compiled by Kouchi et al.[18]. The database provides the dimensions of the following four different Japanese groups.

1. Young male: aged 19-27, average 20.5 years old, 110 samples
2. Young female: aged 19-27, average 20.2 years old, 107 samples
3. Aged male: aged 60-82, average 68.6 years old, 51 samples
4. Aged female: aged 60-80, average 66.9 years old, 50 samples

Considering the entertainment applications like fashion show, we picked the average young female data. Figure 3 shows part of the dimensions provided by the database.

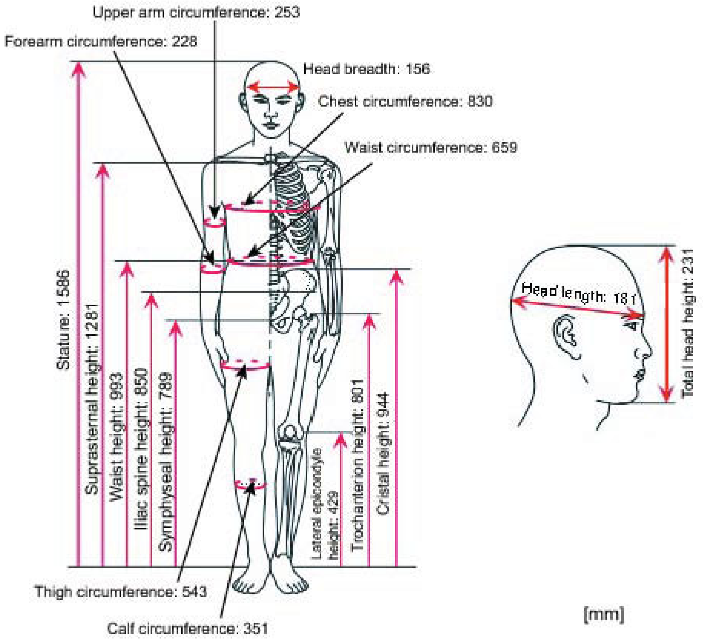


Fig. 3 Anthropometric data of average young Japanese female [18]

In the early stage of design, we explored the possible choice and arrangement of mechanical and electric devices by scaling up and down Fig.3 to have the stature between 145cm and 160cm. In our final design, HRP-4C is 158cm tall which is very close to the average of the young female, 158.6cm.

3.2 Joint Configuration

To obtain graceful motion of females, we asked a professional walking model to perform walking, turning, sitting on chair, and other motions. The positions of 86 markers attached to her body were captured by Vicon Motion Systems, a 3D optical motion capture device (Fig.4). This data was used to evaluate the different joint configurations proposed for HRP-4C structure. By calculating joint angles to realize the captured motion, the necessary movable range was estimated. In addition, we

Fig. 4 Motion capturing of professional walking model (Walking Studio Rei)

estimated the motor power during biped walking to determine the appropriate leg joint configuration.

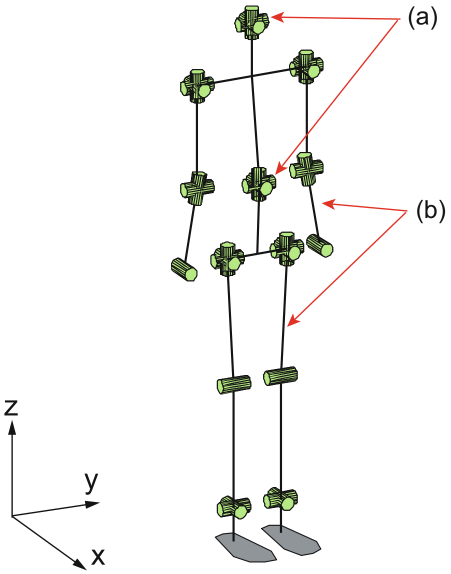


Fig. 5 Joint configuration of HRP-4C body (The head and hands are omitted)

Figure 5 shows the joint configuration we finally decided on for HRP-4C. In this drawing, the joints for head and hands are omitted. It has the following characteristic compared with our former humanoid robots, HRP-2 and HRP-3.

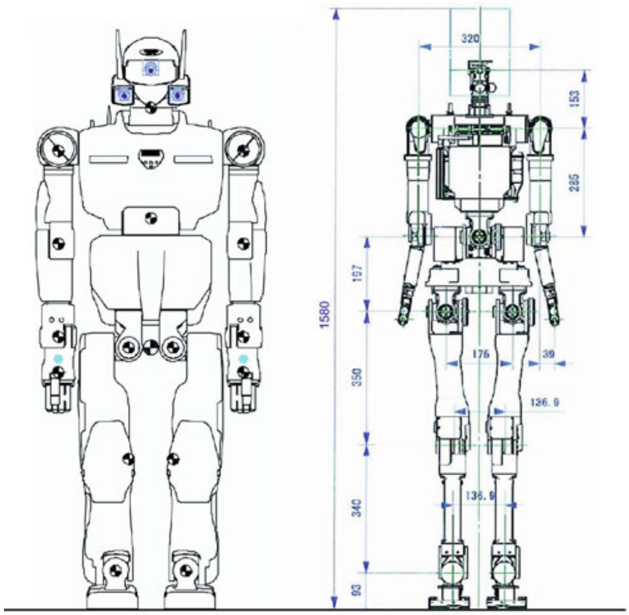
- Roll axes of waist and neck were added to realize human-like behavior (Fig.5 arrow (a))
- Slanted links of forearm and thigh to mimic human (Fig.5 arrow (b))
- Use of standard hip joint structure to realize natural waist line (For HRP-2 and HRP-3, we used cantilever type hip joint[19, 20]).

3.3 Design of the Body Mechanism

Figure 6 shows the designed body mechanism of HRP-4C (right) and HRP-2 (left). By comparison, we see that HRP-4C realized much smaller chest and hip as well as slender extremities. To realize this body mechanism, we adopted the following technologies.

- PCI-104 single board computer and peripheral boards for the whole body motion control
- Distributed network motor drivers
- Development of slim ankle mechanism

For further details of the body mechanism and electronics, see our next report[21].



Design: Kawada Industries. Inc.

Fig. 6 HRP-2 and mechanism of HRP-4C

3.4 Design of Appearance

After the mechanical design of the leg part of HRP-4C was completed, we started to consider the design of appearance. Figure 7 shows the proposed appearances of HRP-4C. Fig.7(a) is a metallic robot with womanly form, (b) is a human-like design with artificial skin and a dress to cover the mechanical parts, (c) is a conventional humanoid robot design and (d) is a design as a mannequin.

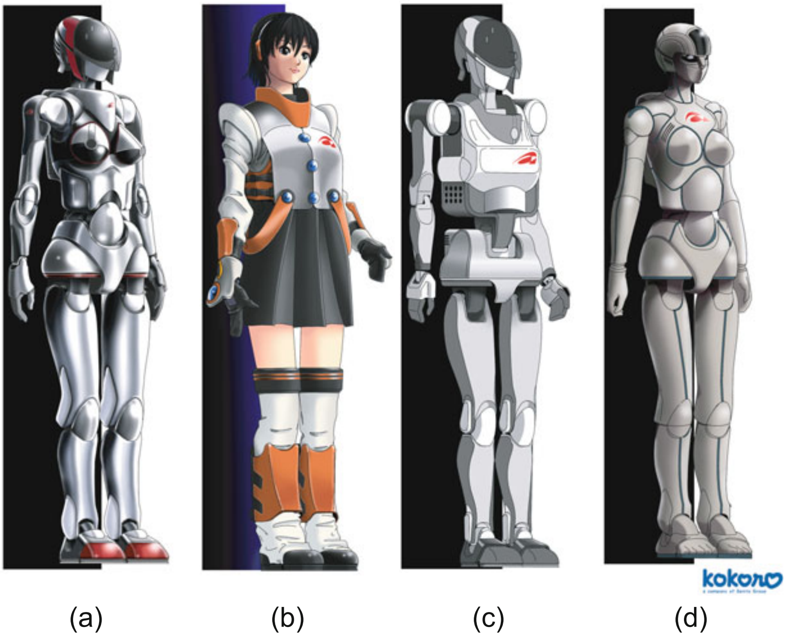


Fig. 7 Proposed designs for HRP-4C

One of the problems of the design like Fig.7(b) is when the degree of similarity passes a certain level, it can make people feel strange or even fearful. This effect was named the *uncanny valley* by Mori[22]. He also pointed out that it can be amplified by the difference of motion pattern between the robot and human. This is serious for an entertainment robot.

Considering the uncanny valley and the impact as the entertainment robot, the final design was decided to be between (a) and (b), that is metallic robot body with a human-like face (Fig.8). We also decided to make the robot face not so real.

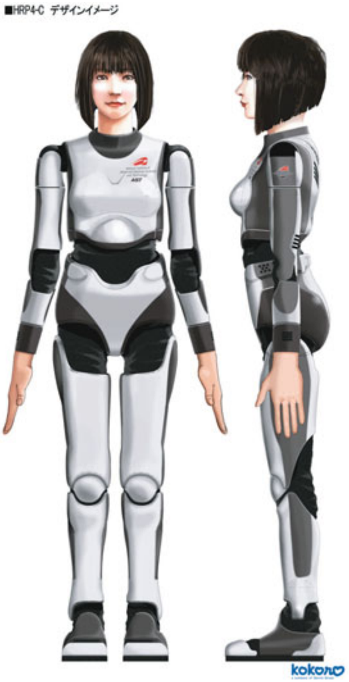


Fig. 8 Final design for HRP-4C

3.5 *Head and Hands*

To realize a head size close to the average young female (Fig.3) and achieve weight reduction, we limited the numbers of degrees of freedom (DOF) of the head. As the minimum DOF to create facial expression we chose the movements shown in Table 2. Since each pair of the eyebrows, eyelids, eyeballs pan and eyeballs tilt are actuated by one servomotor, HRP-4C cannot perform some facial expressions like winking. Figure 9 shows the face of HRP-4C. Despite the above noted limitations, HRP-4C can perform effective facial expressions like smile, surprise, anger etc.

Table 2 Head joints of HRP-4C

Joint name	DOF	Joint name	DOF
Eyebrows	1	Mouth	1
Eyelids	1	Upper lip	1
Eyeballs pan	1	Lower lip	1
Eyeballs tilt	1	Cheek	1
Total			8

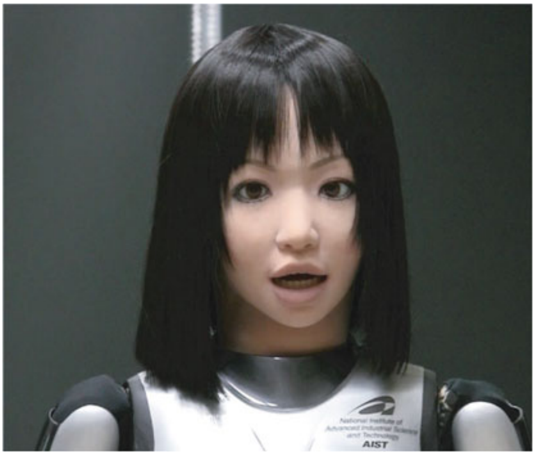


Fig. 9 Face of HRP-4C

We designed the hand of HRP-4C to create motion which is used in dance performance. Its DOF was again limited to the minimum (Table 3). The four fingers from the index to the little fingers are driven by one servomotor and the thumb is driven by another. The size of the hand became bigger than the average young female, due to the selected servomotors.

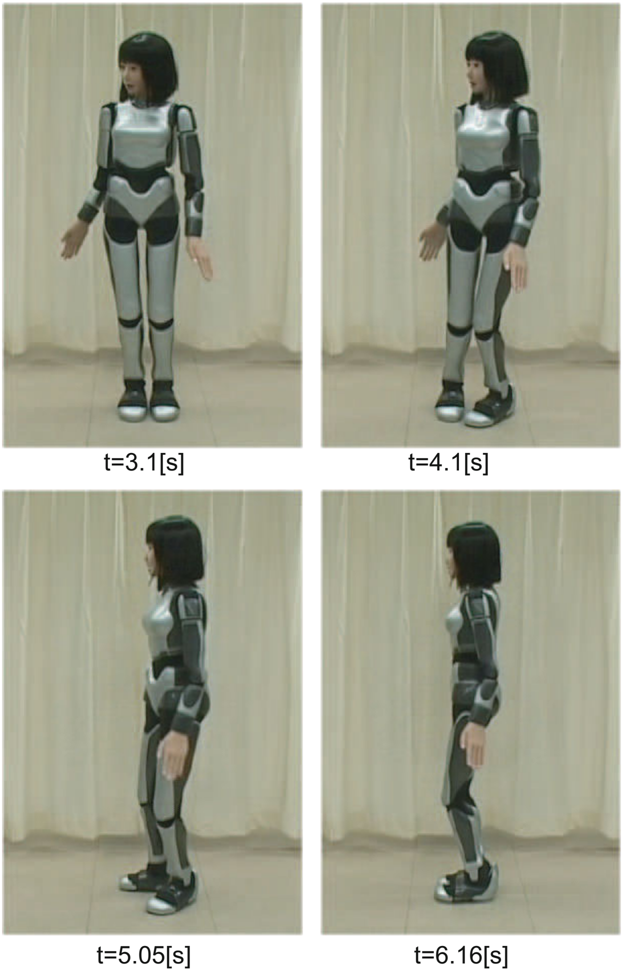
Table 3 Hand joints of HRP-4C

Joint name	DOF
Index, middle, third and little fingers	1
Thumb	1
Total	2

4 Towards Human-Like Motion and Walking

We developed a couple of algorithms to generate human-like motion and walking from the mocap data obtained in 3.2[23, 24]. Figure 10 shows HRP-4C performing a 90 degree turn which was created from the captured human motion. We developed a new control software to stabilize a humanoid robot motion with stretched knee. Currently, the reliability of the controller is not enough, and we are improving it.

We also have developed a software tool to manually program HRP-4C. It can be used to quickly create the motion of HRP-4C through an interactive user interface [25].



t=3.1[s]

t=4.1[s]

t=5.05[s]

t=6.16[s]

Fig. 10 90 degree turn based on captured motion

5 Interaction Using Speech Recognition

We used the open source speech recognition engine Julian[26]. It runs on the CPU embedded in the head of HRP-4C. To realize robust recognition against ambient noise, an operator uses a wireless Bluetooth microphone to send voice commands.

The recognition result is transmitted to the motion control software running on the other CPU in the body of HRP-4C via RT middleware[27]. Figure 11 shows an operator speaking the voice command “Look surprised!” and HRP-4C demonstrating the surprise motion.

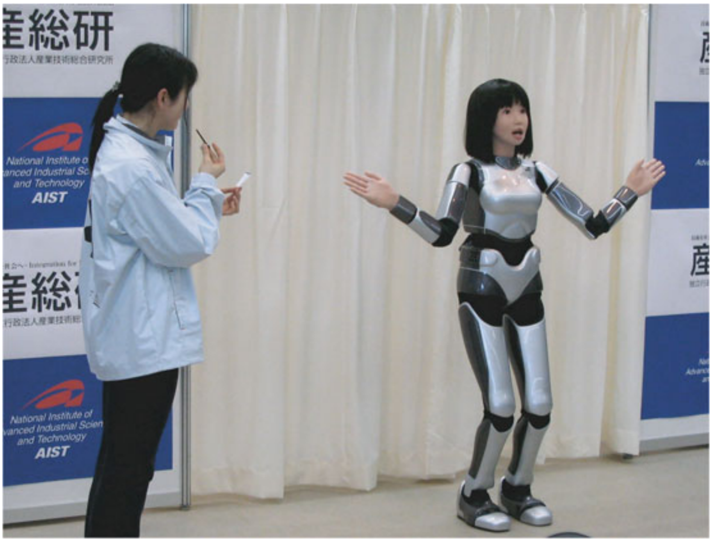


Fig. 11 HRP-4C responding to the voice command ”Look surprised!”

6 Conclusions and Future Work

In this paper, we gave an overview of the development of our new humanoid robot, cybernetic human HRP-4C. The robot has the appearance and shape of a human being, specifically, an average young Japanese female. It can perform biped walk using its own battery and it can interact with humans using speech recognition.

The software of HRP-4C is still under development for the project was carried out with a tight schedule. One of the urgent goal is to realize a reliable human-like biped walking with stretched knees.

Acknowledgments

We would like to thank Dr. Junji Ito, director of AIST, for his generous help obtaining the financial support to the project. Moreover, we also thank Dr. Masaahi Mochimaru, Dr. Kei Aoki and other members of the Digital Human Research Center, who kindly helped our motion capturing experiment. Without the help of above mentioned people, the development of HRP-4C was not possible.

References

1. Wiener, N.: Cybernetics: or Control and Communication in the Animal and the Machine. MIT Press, Cambridge (1948)
2. Kato, I.: Development of biped walking robot WABOT-1. Biomechanism 2, 173–214 (1973) (in Japanese)

3. Sugano, S., Kato, I.: WABOT-2: Autonomous robot with Dexterous Finger-Arm Finger-Arm Coordination Control in Keyboard Performance. In: Proc. of IEEE Int. Conference on Robotics and Automation, vol. 4, pp. 90–97 (1987)
4. Brooks, R.A., Breazeal, C., Marjanovic, M., Scassellati, B., Williamson, M.: The Cog Project: Building a Humanoid Robot. In: Nehaniv, C.L. (ed.) CMAA 1998. LNCS (LNAI), vol. 1562, p. 52. Springer, Heidelberg (1999)
5. Peters, J., Vijayakumar, S., Schaal, S.: Reinforcement Learning for Humanoid Robotics. In: Proc. on IEEE-RAS Int. Conference on Humanoid Robots (2003)
6. Metta, G., Sandini, G., Vernon, D., Natale, L., Nori, F.: The iCub humanoid robot: an open platform for research in embodied cognition. In: PerMIS: Performance Metrics for Intelligent Systems Workshop (2008)
7. Hirai, K., Hirose, M., Haikawa, Y., Takenaka, T.: The Development of Honda Humanoid Robot. In: Proc. IEEE Int. Conference on Robotics and Automation, pp. 1321–1326 (1998)
8. Kagami, S., Nishiwaki, K., Kitagawa, T., Sugihara, T., Inaba, M., Inoue, H.: A Fast Generation Method of a Dynamically Stable Humanoid Robot Trajectory with Enhanced ZMP Constraint. In: Proc. of IEEE Int. Conf. on Humanoid Robotics (2000)
9. Lohmeier, S., Bushmann, T., Ulbrich, H.: Humanoid Robot LOLA. In: Proc. IEEE Int. Conference on Robotics and Automation, pp. 775–780 (2009)
10. Tajima, R., Honda, D., Suga, K.: Fast Running Experiments Involving a Humanoid Robot. In: Proc. of IEEE Int. Conference on Robotics and Automation, pp. 1571–1576 (2009)
11. Collins, S., Ruina, A., Tedrake, R., Wisse, M.: Efficient Bipedal Robots Based on Passive-Dynaic Walkers. Science 307, 1082–1085 (2005)
12. Ogura, Y., Aikawa, H., Shimomura, K., Kondo, H., Morishima, A., Lim, H., Takanishi, A.: Development of A Humanoid Robot WABIAN-2. In: Proc. of IEEE Int. Conference on Robotics and Automation, pp. 76–81 (2006)
13. Hyon, S.H., Osu, R., Otaka, Y.: Integration of multi-level postural balancing on humanoid robots. In: Proc. IEEE Int. Conference on Robotics and Automation, pp. 1549–1556 (2009)
14. Breazeal, C., Scasselati, B.: How to build robots that make friends and influence people. In: Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp. 858–863 (1999)
15. Ishiguro, H.: Android science: conscious and subconscious recognition. Connection Science 18(4), 319–332 (2006)
16. Oh, J.H., Hanson, D., Kim, W.S., Han, I.Y., Kima, J.Y., Park, I.W.: Design of android type humanoid robot albert HUBO. In: Proc. of the IEEE/RSJ Inter. Conf. on Intelligent Robots and systems, pp. 1428–1433 (2006)
17. Hashimoto, T., Hiramatsu, S., Kobayashi, H.: Dynamic display of facial expressions on the face robot made by using a life mask. In: Proc. IEEE-RAS Inter. Conf. on Humanoid Robots, pp. 521–526 (2008)
18. Kouchi, M., Mochimaru, M., Iwasawa, H., Mitani, S.: Anthropometric database for Japanese Population 1997-1998. Japanese Industrial Standards Center (AIST, MITI) (2000), http://riodb.ibase.aist.go.jp/dhbodydb/
19. Kaneko, K., Kanehiro, F., Kajita, S., Hirukawa, H., Kawasaki, T., Hirata, M., Akachi, K., Isozumi, T.: Humanoid Robot HRP-2. In: Proc. IEEE Int. Conference on Robotics and Automation, pp. 1083–1090 (2004)
20. Kaneko, K., Harada, K., Kanehiro, F., Miyamori, G., Akachi, K.: Humanoid Robot HRP-3. In: Proc. IEEE/RSJ Int. Conference on Intelligent Robots and Systems, pp. 2471–2478 (2008)

21. Kaneko, K., Kanehiro, F., Morisawa, M., Miura, K., Nakaoka, S., Kajita, S.: Cybernetic Human HRP-4C. In: 9th IEEE-RAS Int. Conference on Humanoid Robots (2009) (to appear)
22. Mori, M.: The uncanny valley. Energy 7(4), 33–35 (1970), http://www.androidscience.com/theuncannyvalley/proceedings2005/uncannyvalley.html
23. Harada, K., Miura, K., Morisawa, M., Kaneko, K., Nakaoka, S., Kanehiro, F., Tsuji, T., Kajita, S.: Toward Human-Like Walking Pattern Generator. In: Proc. IEEE/RSJ Int. Conference on Intelligent Robots and Systems, pp. 1071–1077 (2009)
24. Miura, K., Moisawa, M., Nakaoka, S., Kanehiro, F., Harada, K., Kaneko, K., Kajita, S.: Towards Human-like Locomotion of Humanoid Robots Walking and Turning based on Motion Capture Data. In: 9th IEEE-RAS Int. Conference on Humanoid Robots (2009)
25. Nakaoka, K., Miura, M., et al.: Creating Facial Motions of Cybernetic Human HRP-4C. In: 9th IEEE-RAS Inter. Conf. on Humanoid Robots (2009) (to appear)
26. Lee, A., Kawahara, T., Shikano, K.: Julius — an open source real-time large vocabulary recognition engine. In: Proc. European Conference on Speech Communication and Technology (EUROSPEECH), pp. 1691–1694 (2001)
27. OpenRTM-aist (2007), http://www.is.aist.go.jp/rt/OpenRTM-aist

Reconstruction and Verification of 3D Object Models for Grasping

Zoltan-Csaba Marton, Lucian Goron, Radu Bogdan Rusu, and Michel Beetz

Abstract. In this paper we present a method for approximating complete models of objects with 3D shape primitives, by exploiting common symmetries in objects of daily use. Our proposed approach reconstructs boxes and cylindrical parts of objects from sampled point cloud data, and produces CAD-like surface models needed for generating grasping strategies. To verify the results, we present a set of experimental results using real-world data-sets containing a large number of objects from different views.

1 Introduction

In this paper we discuss a method on how to obtain suitable complete 3D representations for typical objects in a kitchen table cleaning scenario. Our approach differentiates from similar research initiatives in the sense that we do not use databases containing predefined object models in combination with machine learning classifiers to address the object reconstruction problem. Instead we generate CAD-like quality surface models that are extremely smooth and can directly be used to infer grasping points by exploiting shape symmetries. To show the applicability of our approach, we make use of a mobile manipulation platform (see Figure 1) to acquire 3D point cloud datasets, and show segmentation results for table planes together with sets of unseen objects located on them. The result of our mapping pipeline includes a complete set of 3D representations that can be used to compute grasping points.

Most grasping paradigms, like [11] and [5], require a CAD-like representation of the objects, which is difficult to obtain from sensed data. The two main approaches to produce these representations rely on image or depth information. In the first case,

Zoltan-Csaba Marton · Lucian Goron · Radu Bogdan Rusu · Michael Beetz
Technische Universitaet Muenchen, Boltzmannstr. 3, 85748 Garching b. Muenchen
e-mail: marton@cs.tum.edu, goron@cs.tum.edu, rusu@cs.tum.edu,
beetz@cs.tum.edu

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 315–328.
springerlink.com

usually a model from a database is matched to the image as in [3, 6], while in the latter, a more flexible combination of shape primitives [17], superquadrics [1, 21] are fit, or a triangulation of the surface [13, 9] is performed.

The most accurate 3D sensing devices usable by robots are laser scanners, which can have an accuracy in the millimeter range for some surface types, but they can provide only partial information about the object (one side, from a single viewpoint), and they have problems when dealing with shiny (eg. metal or ceramic) objects. Unfortunately, these are quite common in every day manipulation tasks, like those a personal assistant robot would encounter in a kitchen, for example. These objects need to be segmented into distinct clusters for grasping, but in some cases an object appears in two separate clusters because of the previously mentioned problems, or occlusions. Thus a mug is typically represented by point clouds in two semi-circular parts and with only a few points on the handle (see Figure 2).

Some simplifications have to be made in order to be able to approximate the occluded parts of objects. Our assumption is that most objects have a vertical plane or axis of symmetry (eg. mugs, boxes, bottles, jars, plates, pans, bowl, silverware, etc.), are composed of planar and cylindrical parts, or can be roughly approximated by such, and are representable as groupings of planar patches and cylindrical parts. Because of their vertical symmetries, these models can be obtained by analyzing the footprint of the objects on their supporting plane, and detecting linear and circular segments in it (see Figure 3). We call these objects with sides that are perpendicular

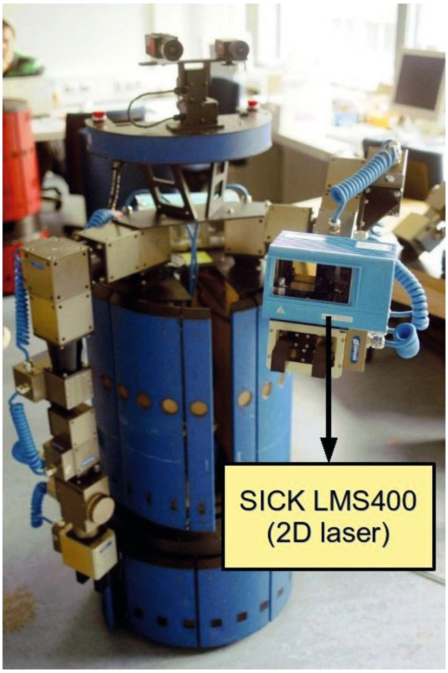


Fig. 1 The mobile manipulation platform used for the experiments presented herein.

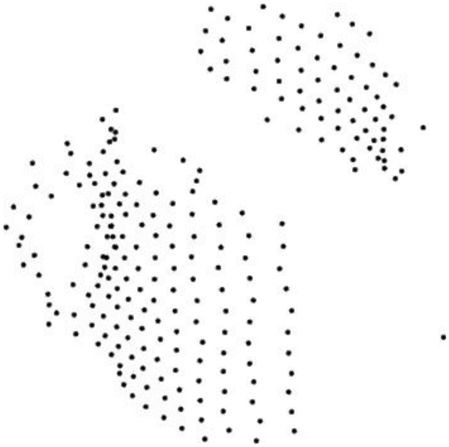


Fig. 2 Typical one-side scan of a mug, where the rim and handle didn't return enough measurement points, thus the mug will be over-segmented into two separate connected components, and the handle is hard to detect.

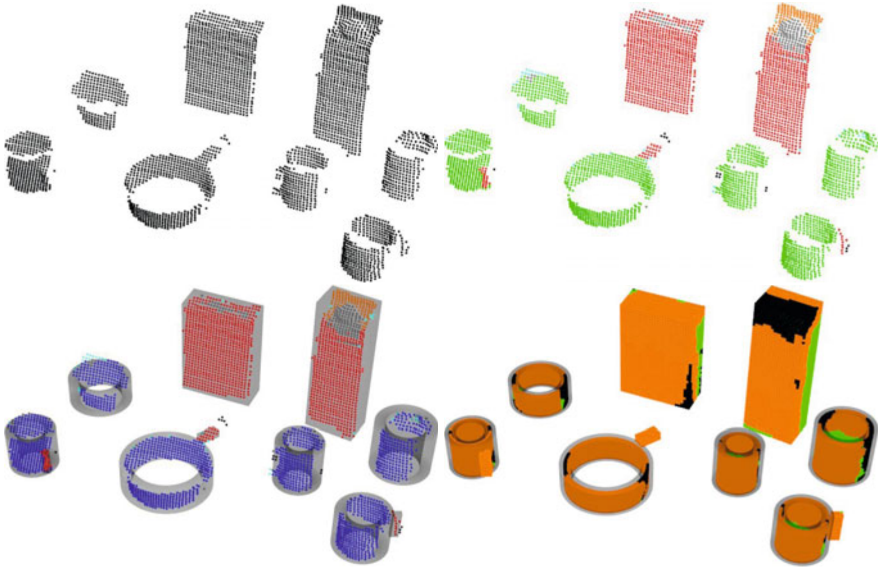


Fig. 3 The most important processing steps of our method are illustrated in the pictures, from left to right, top to bottom: a) measurement points returned from objects on the table are segmented from the complete scan, and sparse outliers are removed; b) connected components are identified and shape primitives are fit to its footprint silhouettes (points belonging to a vertical plane are marked with red and orange, while those belonging to cylinders are green); c) the 3D shapes (boxes and thick cylinders) are obtained from the corrected shape primitives; d) points are generated on the shapes and verified for correspondence to measurements and visibility (green points are invisible from the current viewpoint, orange points are verified by measurements while black points are in visible free space, meaning that the model of the object should not include those parts of the shapes.

to their bottom *standing objects*. Spherical, toroidal and conical parts will be approximated with cylinders, but this would explain the data only partially, so we can recognize the cases when these problems are encountered. This approach provides straightforward ways of correction (like merging two components if they belong together) and verification of the correctness of the fitted model at each surface area unit.

The main contributions of our approach are:

- exploiting common symmetries to produce suitable completed models for grasping applications from single view scans;
- creating a complete model from a single view for *standing objects*, together with a measure for verifying the approximated reconstruction;
- a way to deal with 3D over-segmentation of objects produced by occlusions and measurement errors.

The remaining of the paper is organized as follows. In the next section an overview is given on the current approaches, followed by the presentation of our method in Section 3. The experimental results are analyzed in Section 4, followed by our conclusions and a discussion on future work in Section 5.

2 Related Work

A computer vision and machine learning based method is used in [16] to train classifiers that can predict the grasping points in an image. This is then applied to images of unseen objects. To obtain 3D positions of grasping points, the authors use stereo cameras, but their approach works reliably only to the extent provided by the training data. Another issue is the segmentation of objects, since only grasp points are provided with no information about what objects are in the scene and to which of them do the identified points correspond. In [2] an accurate line laser and a camera builds models and identifies grasping points for novel objects with very encouraging results. However the system was tested only on two objects, thus its scalability is not clear. Available models of complex objects are decomposed into superquadric parts in [1, 21], and these models are fit to a point cloud. This however needs a database of models, and moreover, their decomposition into superquadric components, which is often difficult to obtain.

In purely computer vision based approaches either features like [8] or [7] are used to find matches between parts of a scene and a database of object images. The problem with these kinds of approaches is that they only work for objects that are in the database, and since no knowledge about the 3D information is known, the system can easily make mistakes and return false positives (e.g., a cereal box containing a picture of a beer bottle printed on it might get recognized as a bottle of beer). Another approach to obtain 3D information directly form camera images, is to project CAD models from a database to the image and search for good fits in the edges domain, for example like in [19]. While this is a more direct method, it's still dependent on a database of different CAD models. Acquiring these automatically from the Internet has been explored in [6], but obtaining the models of all the objects in a scene requires selecting all the possible good fits of the models to the image, which takes considerable amounts of time. In our approach, we do not have these problems, since we have access to the 3D information directly, so we can use a bottom up approach for reconstructing the object model from the geometry data, with lower computational constraints than the initiatives mentioned previously.

Hough transforms are sometimes used to detected geometric shapes like cylinders in [12], or planar patches in [20] in point clouds. However they are not as popular as sample consensus based methods like RANSAC [4], since a parameter space has to be constructed for each shape type separately, which complicates things for more complex models. In the sample consensus paradigm, the data is used directly to compute best-fit models. We are using RANSAC because it allows the definition of different models for more complex geometric shapes.

A similar sample consensus based approach for model decomposition is presented in [17], where a set of 3D geometric primitives (planes, spheres, cylinders, cones and tori) are fit to noisy point clouds. Since the point clouds presented there are complete, the authors don't need to reconstruct the missing parts. To solve this problems in our case, we are fitting planar and cylindrical shapes, and exploit the vertical symmetries present in most objects to reconstruct their occluded parts. In [18] the authors describe a method for detecting and verifying symmetries in point clouds obtained from a single viewpoint which works very well for nicely segmented objects, however the problem of under- or over-segmented objects remains.

3 Object Model Reconstruction

Our system takes a single view of a table scene as input, and extracts the table together with the points returned from the objects that are on it. The data is cleaned using a statistical analysis of point densities, and clustering is performed to segment the different objects into separate regions.

These regions are then reconstructed, and the fitted models are corrected, evaluated, and used for detecting cases of over-segmentation (i.e., when objects are split into multiple regions).

In the next subsections we present the aforementioned steps in more detail.

3.1 Table Detection

The initial step of our method is to locate the table and the objects on it. This problem falls outside the scope of this paper, and our previous work on Object Maps [15] have already presented the robust localization of important furniture parts in a kitchen in more detail. After the location of the table is obtained, a scan can be made of the area, and a restricted planar search can be performed in order to obtain the model of the table as presented in Figure 4 (see [15] for more details).

3.2 Object Footprints

After the points that are above the table are obtained, the sparse outliers are removed using a statistical analysis of the point densities in each points neighborhood as detailed in [14]. The remaining points are projected along the normal of table plane to obtain 2D clusters, which are then grouped based on a connectivity criterion.

In each cluster a set of boundary points are identified as those which have a maximum angle between the vectors pointing towards their neighbors (from the same region) that matches or exceeds the opening of a what would be a straight line (that is 180°). The neighbors of these points are also marked as boundary points in order to provide a contingent set of boundary points around each cluster. These "footprint" points are then used to match shape primitives to them as it can be seen in Figure 5.

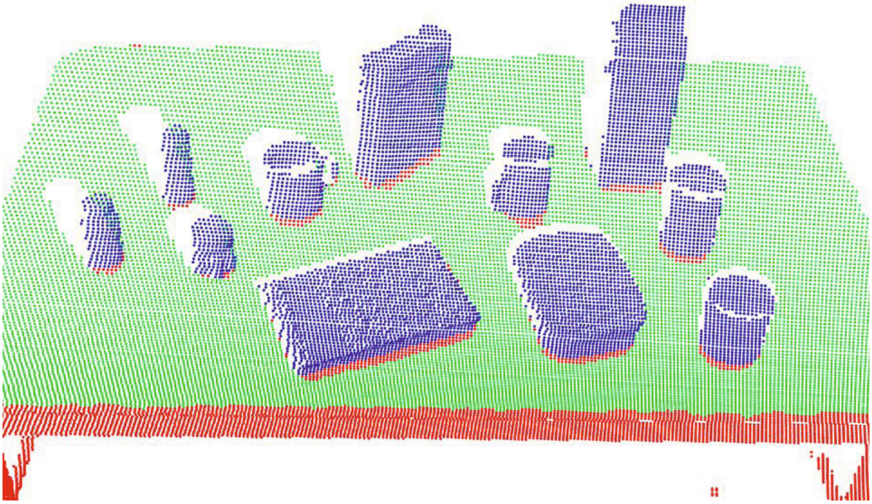


Fig. 4 Detected table and objects on top of it in a partial scan.

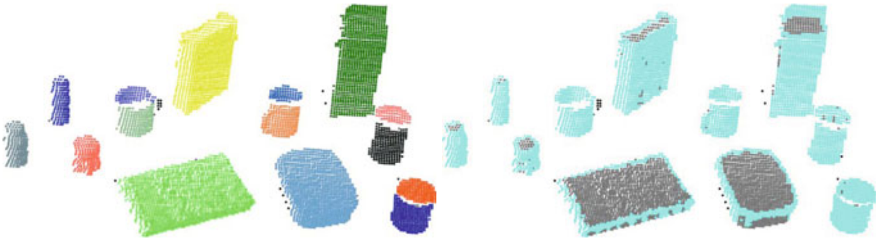


Fig. 5 Left: the identified objects on the table based on the clustering of their projections shown in random colors. Right: points that lie on the boundaries of the clusters are highlighted.

The search radius used for identifying the neighbors was chosen so that it compensates the variations caused by noise, but still includes points on thick handles such that a robust fit can be performed.

3.3 Hierarchical Model Fitting

Having the 2D boundary points for each cluster, we fit shape primitives to them, lines and circles more specifically, in order to locate the vertical planes and cylinders in the object.

Initially a line and a circle is fit to the footprint, and whichever has the most inliers gets accepted. Before accepting a circle however, two conditions have to be met.

Since a single circle can approximate a rectangle much easier than a single line, special care has to be taken to correctly recognize rectangles. For this, an oriented bounding rectangle is computed around each region using principle component analysis, and the average of normalized distances to the closest boundary points is computed as:

$$\mu = \frac{1}{N}\sum_{i=1}^{N}\frac{dist(p_i, obr)}{width(p_i, obr)}, \tag{1}$$

where N is the number of boundary points p_i in a region, obr is the oriented bounding rectangle of the region, and the functions $dist()$ and $width()$ return the minimum distance of the point p_i to the sides of obr and the width of the bounding rectangle along the measurement direction respectively. Thus a fraction is computed between the distance of the point to a side of the obr along a principle component, and the width of the obr along that principle component.

Naturally, μ will have smaller values for rectangles than for circles or half-circles. The average normalized distance was found to be around 3% for rectangular footprints, and above 5% for non-rectangular ones, thus allowing us to decide when to neglect a first circular fit to the region (please see Figure 6).

In some cases checking the oriented bounding box is not enough, so circles have to be checked also against a set of parallel and perpendicular lines fit to the data. To do this, lines are fit to the cluster until there are not enough points left, and a subset

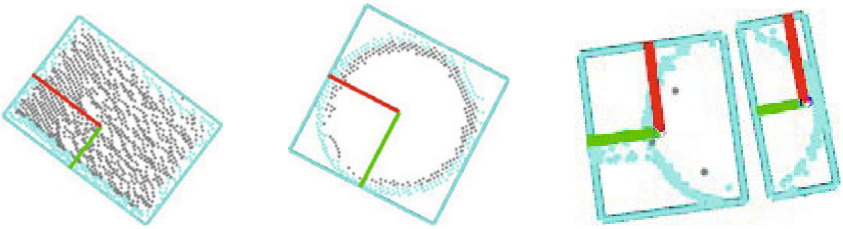


Fig. 6 Left to right: PCA analysis performed on clusters of a book, pan and the two parts of a mug.

of these lines is selected that are pairwise parallel or perpendicular, and have the most number of inliers. This exhaustive search has a low computational complexity, since it only has to be performed for points in a single cluster.

Please note that the orange points in Figure 3b are inliers to a line that was grouped to the previous parallel line found in the cluster.

In the subsequent iterations, the steps are repeated (except the check with the oriented bounding rectangle, of course) for the remaining points, until their number drops below a minimum threshold for which a robust fit can not be ensured anymore. Please see the results in Figure 7.

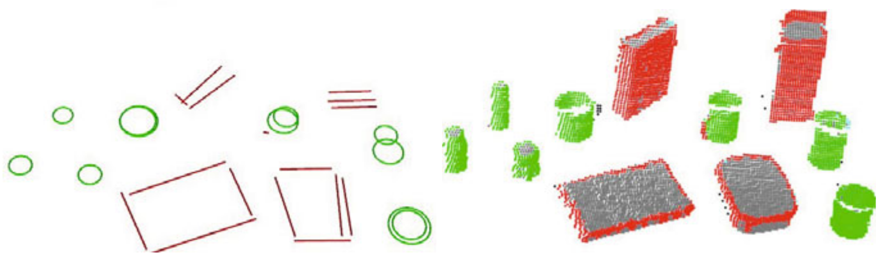


Fig. 7 Left: the 2D shapes that were fit to the cluster. Right: the inliers of those shapes in 3D.

3.4 Model Correction and Merging

After a set of shape primitives were fit to the region, these have to be corrected and possibly merged to improve the quality of the reconstruction.

Inliers of 2D circles which have a high overlap, are likely to belong to the same 3D cylinders, so merging them simplifies and also corrects the reconstruction. The criterion for merging circles, is that one of them includes the center of the other, since small measurement errors, or the presence of a handle for example can already modify the position and size of the best fit considerably.

When two circles from the same region are merged, a weighted average is computed for them, where the weights are the number of inliers of each circle. A re-fit using RANSAC would be redundant, because a search for circles was already performed in the region, and yielded the two circles separately. For circles from different regions, a complete re-fit is possible since their inliers were not considered already by the sample consensus method, so this gives accurate results even for small overlaps. An example for a merge between circles from different regions is illustrated in Figure 8. In these cases, the two regions are merged into one, since the two parts of a circle indicate a strong evidence that over-segmentation occurred.

Lines that form parallel and/or perpendicular groups in a region are also merged, since they are most probably part of a box for which the boundaries were identified (see Figure 9 for results).

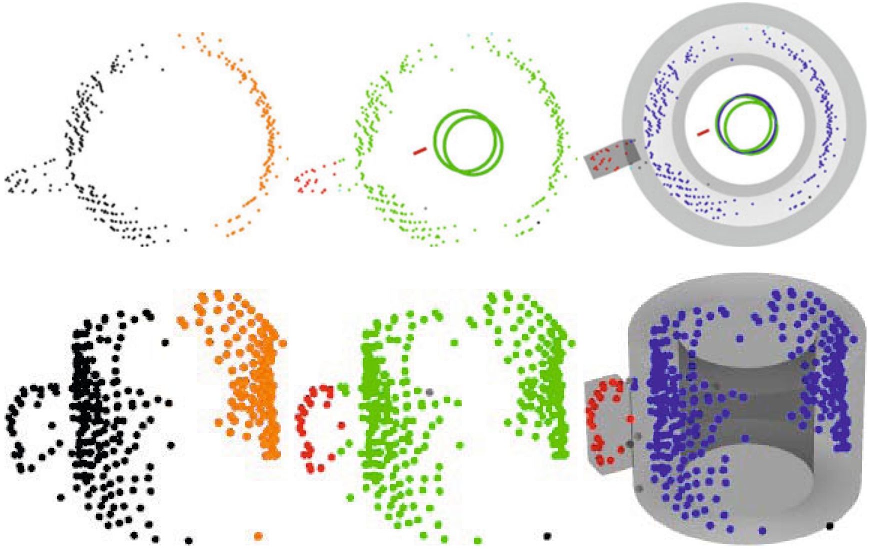


Fig. 8 From left to right (top row is viewed from above, while the bottom from the side): a) original measurement points returned from a mug, clustered in two disconnected regions; b) the inliers to the primitives identified in the regions (the shapes themselves can be seen in the background of the top row pictures) c) the two circles are merged and the model is re-fitted.

3.5 Model Reconstruction and Verification

The corrected lines and circles are transformed into boxes and thick cylinders respectively, using the minimum and maximum heights of their inliers, together with the distances of the inliers to the model as thickness (as shown in the right part of Figure 9).

In order to verify these models, we generate points on a grid on the sides of the boxes and on the surface of the cylinders defined by the initial circles. These points can fall into 3 categories:

1. points that are *invisible* from the current viewpoint;
2. points that are *verified* by measurements;
3. points that are *void*, meaning that they are in visible free space, thus the model of the object should not include those parts of the shapes.

To check which points are verified, we verify if there are measurement points in their neighborhoods, within a maximum distance d_{th}. Those points that are not *verified* are checked for visibility by verifying the points that lie along the vector that connects the point to the viewpoint, or at most at distance d_{th} from it. If the point is occluded by measurement points, it is marked as *invisible* and as *void* otherwise.

An example can be seen in Figure 10. This way we can form an image about the measure of these misalignments. Generally we can say that the error of the assumption that the sides of the objects are perpendicular to the estimated normal of the table, lies well below 5 degrees, as does the error in approximating the rotation of the object around the normal.

The resulting point categories give valuable feedback about the probability of a successful fit for that particular shape, but also on how well the object respected our assumptions. For their interpretation please see Section 5.

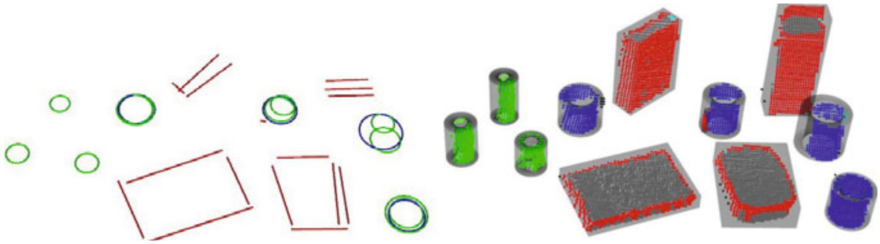


Fig. 9 Left: the corrected circles are marked with blue. Right: the reconstructed 3D models from the merged 2D shapes.

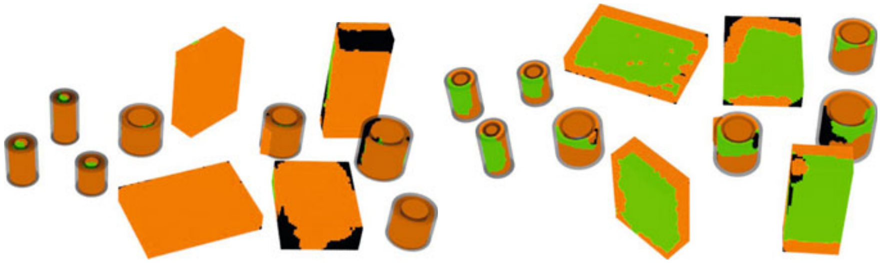


Fig. 10 Verification of object models, viewed from the front (left) and from the back (right). Orange points are marked as *verified*, green ones as *invisible* and black points as *void*.

4 Experimental Results

We applied our method to several views of tables containing objects of every day use on them (e.g., boxes, tetra packs, mugs, jars, small containers, plates and pans) at different distances and orientations, and obtained fairly robust results, as it can be seen in Figures 3, 9 and 11.

The small variations in the results for the same dataset, are due to the random element in the sample consensus approach, but the method gives consistent approximations. There is a small ambiguity for very thin, small containers, which are sometimes reconstructed as boxes instead of cylinders, but the small inaccuracies of the laser scanner make it very hard to distinguish circles with small radii from a box. Please see Figure 12 for an example.

In some cases an incorrect fit of a cylinder is accepted by the method over a part of a box,but they get rejected whenever they are verifiable, as presented in Figure 13.

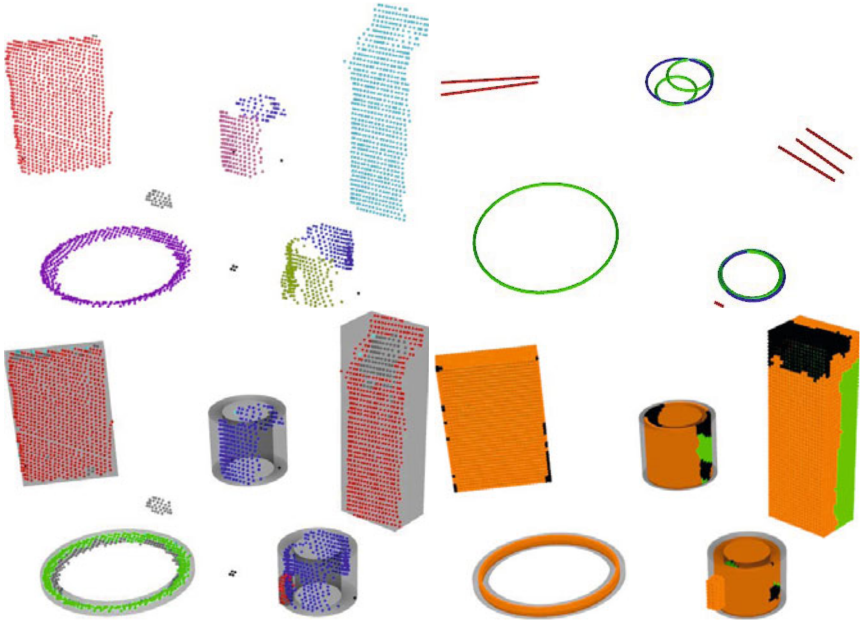


Fig. 11 From top to bottom, left to right: a) connected components, b) shape primitives and their corrections, c) the obtained boxes and thick cylinders, and d) the labels of the areas on the models. For the interpretation of the colors we kindly refer the reader to the explanations of the previous pictures.

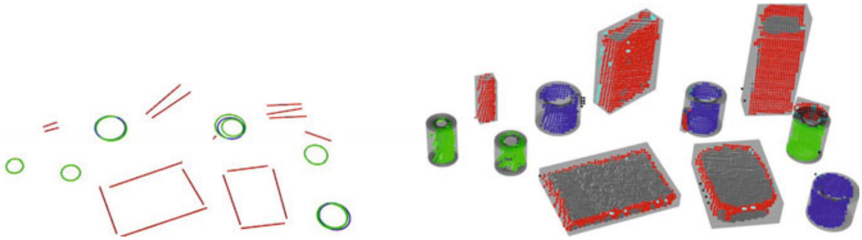


Fig. 12 A slightly different reconstruction of the scene presented previously in Figure 9. Left: the 2D shapes that were fit to the cluster. Right: the inliers of those shapes in 3D.

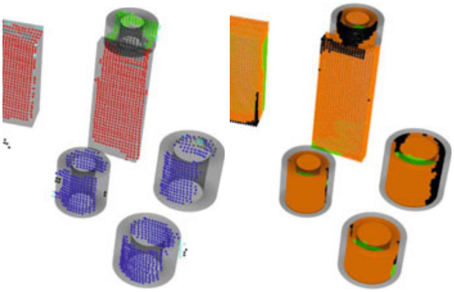


Fig. 13 Incorrect fit of the cylinder on top of the box, detected on the basis of the high amount of *void* parts.

5 Conclusions

In this paper, we presented a method for producing approximations of complete object models from a single partial view, by fitting planar patches (and combining them into boxes) together with cylindrical areas to 3D point cloud data. The processing steps include an analysis of the silhouettes of the object's 2D projections, and their decomposition into shape primitives. These are then used to recompute the parameters of 3D shapes that model the real objects well enough for estimiating grasping points. The approach can easily be extended to model different layers of objects separately for increased accuracy.

A method for merging parts of over-segmented objects is inherently embedded in our approach, as is the verification and correction of the models. The labels set by the model verification step provide means of checking the correctness of the model at each surface unit, refit the model if the current one is implausible (see Figure 13), and remove parts of the model if necessary. This way concavities can be recognized and since the model extends to the limits of the inliers, unexpected collisions can be avoided.

While the exact grasping strategy is not the scope of this paper (please see the approaches mentioned earlier for example), this information can also be used to optimize grasping (since poses for grabbing boxes and cylinders are relatively easy to generate) and improve its accuracy. In the case of pre-generated grasps for boxes and cylinders, apart from the collision checks with the other parts of the object and the surroundings, this can be achieved by excluding the generated end-effector poses that would require it to have contact in the *void* points for a successful grip. If grasps are to be generated after the completed model is obtained, good grasps can be obtained by adjusting the friction coefficients on the model based on these labels.

The points remaining after fitting the shape models might hold some information, so grouping them and fitting them into boxes might be useful, but this will have to be limited in order to avoid the unnecessary complication of the object models because of measurement noise. The points which can not be explained by the models might be introduced as triangular meshes to form hybrid object models, but the primary problem here is the resolution and accuracy of the measurements, which still has room for improvement, for example by accepting multiple return pulses instead of averaging them in the case when the laser beam hits an edge and the object behind it as well.

Our next steps will be to work on the problem of separating objects which are located close to each other, and to extract features from the fitted model for using them in training a a classifier that differentiates between plausible and less plausible configurations, i.e. to find out how probable is that a combinations of fitted models to a cluster is approximating the true 3D structure correctly (employing similar techniques as in [10]). For this, a large number of hand-labeled training sets of correct and incorrect fits is needed, for which the slight variations for the same data introduced by the random sample consensus method works to our advantage.

Acknowledgements. This work is supported by the CoTeSys (Cognition for Technical Systems) cluster of excellence.

References

1. Biegelbauer, G., Vincze, M.: Efficient 3D Object Detection by Fitting Superquadrics to Range Image Data for Robot's Object Manipulation. In: IEEE International Conference on Robotics and Automation (ICRA), Rome, Italy (2007)
2. Bone, G., Lambert, A., Edwards, M.: Automated Modeling and Robotic Grasping of Unknown Three-Dimensional Objects. In: Proceedings of the 2008 IEEE International Conference on Robotics and Automation, Pasadena, USA (2008)
3. Collet, A., Berenson, D., Srinivasa, S.S., Ferguson, D.: Object Recognition and Full Pose Registration from a Single Image for Robotic Manipulation. In: IEEE International Conference on Robotics and Automation (ICRA), Kobe, Japan (2009)
4. Fischler, M., Bolles, R.: Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Comm. of the ACM 24 (1981)
5. Harada, K., Kaneko, K., Kanehiro, F.: Fast grasp planning for hand/arm systems based on convex model. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Kobe, Japan, pp. 1162–1168 (2009)
6. Klank, U., Zia, M.Z., Beetz, M.: 3D Model Selection from an Internet Database for Robotic Vision. In: International Conference on Robotics and Automation (ICRA), Kobe, Japan (2009)
7. Lepetit, V., Fua, P.: Keypoint recognition using randomized trees. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(9), 1465–1479 (2006)
8. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), 91–110 (2004), http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94
9. Marton, Z.C., Rusu, R.B., Beetz, M.: On Fast Surface Reconstruction Methods for Large and Noisy Datasets. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Kobe, Japan (2009)
10. Marton, Z.C., Rusu, R.B., Jain, D., Klank, U., Beetz, M.: Probabilistic Categorization of Kitchen Objects in Table Settings with a Composite Sensor. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), St. Louis, MO, USA (2009)

11. Miller, A., Allen, P.K.: Graspit!: A Versatile Simulator for Robotic Grasping. IEEE Robotics and Automation Magazine 11(4), 110–122 (2004)
12. Rabbani, T., Heuvel, F.: Efficient hough transform for automatic detection of cylinder in point clouds. In: ISPRS WG III/3, III/4, V/3 Workshop, Laser scanning, Enschede, The Netherlands (2005)
13. Richtsfeld, M., Vincze, M.: Grasping of Unknown Objects from a Table Top. In: Workshop on Vision in Action: Efficient strategies for cognitive agents in complex environments (2008)
14. Rusu, R.B., Marton, Z.C., Blodow, N., Dolha, M., Beetz, M.: Towards 3D Point Cloud Based Object Maps for Household Environments. Robotics and Autonomous Systems Journal (Special Issue on Semantic Knowledge) (2008)
15. Rusu, R.B., Marton, Z.C., Blodow, N., Dolha, M.E., Beetz, M.: Functional Object Mapping of Kitchen Environments. In: Proceedings of the 21st IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Nice, France, September 22-26 (2008)
16. Saxena, A., Driemeyer, J., Ng, A.Y.: Robotic Grasping of Novel Objects using Vision. The International Journal of Robotics Research 27(2), 157–173 (2008)
17. Schnabel, R., Wahl, R., Klein, R.: Efficient RANSAC for Point-Cloud Shape Detection. Computer Graphics Forum 26(2), 214–226 (2007)
18. Thrun, S., Wegbreit, B.: Shape from symmetry. In: Proceedings of the International Conference on Computer Vision (ICCV). IEEE, Bejing, China (2005)
19. Ulrich, M., Wiedemann, C., Steger, C.: Cad-based recognition of 3d objects in monocular images. In: International Conference on Robotics and Automation, pp. 1191–1198 (2009)
20. Vosselman, G., Dijkman, S.: 3d building model reconstruction from point clouds and ground plans. In: International Archives of Photogrammetry and Remote Sensing, Annapolis, MD, October 22–24, 2001, vol. XXXIV-3/W4, pp. 37–43 (2005)
21. Zhang, Y., Koschan, A., Abidi, M.: Superquadric Representation of Automotive Parts Applying Part Decomposition. Journal of Electronic Imaging, Special Issue on Quality Control by Artificial Vision 13(3), 411–417 (2004)

Floating Visual Grasp of Unknown Objects Using an Elastic Reconstruction Surface

Vincenzo Lippiello, Fabio Ruggiero, and Bruno Siciliano

Abstract. In this paper a new method for fast visual grasp of unknown objects is presented. The method is composed of an object surface reconstruction algorithm and of a local grasp planner, evolving in a parallel way. The reconstruction algorithm makes use of images taken by a camera carried by the robot, mounted in an eye-in-hand configuration. An elastic reconstruction sphere, composed by masses interconnected each other by springs, is virtually placed around the object. The sphere is let to evolve dynamically under the action of external forces, which push the masses towards the object centroid. To smoothen the surface evolution, spatial dampers are attached to each mass. The reconstruction surface shrinks toward its center of mass until some pieces of its surface intercept the object visual hull, and thus local rejection forces are generated to push out the reconstruction points until they stay into the visual hull. This process shapes the sphere around the unknown object. Running in parallel to the reconstruction algorithm, the grasp planner moves the fingertips, floating on the current available reconstructed surface, according to suitable quality measures. The fingers keep moving towards local minima depending on the evolution of the reconstruction surface deformation. The process stops when the object has been completely reconstructed and the planner reaches a local minimum. Quality measures considering both hand and grasp proprieties are adopted. Simulations are presented, showing the effectiveness of the proposed algorithm.

1 Introduction

Operating in unstructured environments is a challenging research field which has not been widely investigated as far as the problem of grasping unknown objects is concerned.

Vincenzo Lippiello · Fabio Ruggiero · Bruno Siciliano
PRISMA Lab, Dipartimento di Informatica e Sistemistica,
Università di Napoli Federico II, Italy
e-mail: {vincenzo.lippiello,fabio.ruggiero,
bruno.siciliano}@unina.it

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 329–344.
springerlink.com

Grasping and manipulation tasks, in general, require a priori knowledge about the object geometry. One of the first approaches to grasping in unknown environments can be found in [26], where visual control of grasping is performed employing visual information to track both object and fingers positions. A method to grasp an unknown object using information provided by a deformable contour model algorithm is proposed in [16]. In [25] an omnidirectional camera is used to recognize the shape of the unknown object while grasping is achieved on the basis of a grasping quality measure, using a soft-fingered hand.

From another point of view, it is straightforward to recognize that two main tasks have to be performed to achieve unknown objects grasping: object recognition/reconstruction and grasp planning.

In literature, different methods have been proposed to cope with 3D model reconstruction of unknown objects. The main differences lies in how the available object images are processed and on the algorithms used for object reconstruction. A number of algorithms can be classified under the so called *volumetric scene reconstruction* approach [6]. This category can be further divided into two main groups: the *shape from silhouettes* and the *shape from photo-consistency* algorithms. Another method, proposed in [24], considers a surface that moves towards the object under the influence of internal forces, produced by the surface itself, and external forces, given by the image data.

The finite-elements method is used in [5] to reconstruct both 2D and 3D object boundaries. Using an active contour model, data extracted from taken images are employed to generate a pressure force on the active contour that inflate or deflate the curve, making its behavior like a balloon.

A technique for computing a polyhedral representation of the *visual hull* [12] –the set of points in the space that are projected inside each image silhouette– is studied in [9]. Other approaches rely on the use of *apparent contours* [4, 19], where the reconstruction is based on the spatio-temporal analysis of deformable silhouettes.

On the other hand, grasp planning techniques rely upon the choice of grasp quality measures used to select suitable grasp points. Several quality measures proposed in the literature depend on the properties of the grasp matrix, that means on the grasp geometry; others are based on the area of the polygon created by the contact points, or on the external resistent wrench.

In [7] two generally optimal criteria are introduced, where the total finger force and the maximum finger force are considered, while in [15] simple geometric conditions to reach an optimal force closure grasp both in 2-D and in 3-D are found. The geometric properties of the grasp are also used in [13] to define quality measures, as well as suitable task ellipsoids in the wrench space of the object are proposed to evaluate grasp quality also with respect to the particular manipulation task.

Measures depending on the hand configurations [20] define a set of quality measures based on the evaluation of the capability of the hand to realize the optimal grasp. A rich survey of these grasp quality measures can be found in [22].

To plan a grasp for a particular robotic hand, quality measures depending both on grasp geometry and on hand configuration should be taken into account. Few papers address the problem of grasping an unknown object using a given robotic hand, able to reach the desired contact points in a dexterous configuration [1, 3, 8, 10, 11].

A grasp control task is considered in [17], where several controllers are combined to reach different wrench closure configurations, while in [18] grasp prototypes – generalization of example grasps– are used as starting points in a search for good grasps.

In this paper, a new method for fast visual grasping of unknown objects using a camera mounted on a robot in an eye-in-hand configuration is presented. This method is composed of an *object surface reconstruction algorithm* and of a *local grasp planner*, which evolve in a synchronized parallel way. The reconstruction algorithm makes use of images taken with a camera carried by the robot. First, a rough estimation of the object position and dimensions is performed, and an *elastic reconstruction surface* with a spherical shape is virtually placed around the object. Then, the fingertips of the robotic hand are suitably attached on it, at a suitable *floating safety distance*. The reconstruction surface is sampled by points, which are endowed with a virtual mass, and are interconnected with each other by means of virtual springs resulting in a cross reticulum. A virtual spatial damper is also considered to smoothen the motion of the surface sample points. During the reconstruction process, the elastic surface evolves in a dynamic way leaning to shrink on itself under the action of reticular elastic forces and external forces, pointing to the contours of the visual hull defined by the object silhouettes. For each mass, the external *compression forces* becomes *repulsive forces* along the outgoing direction with respect to the visual hull if the reconstruction point comes into the visual hull, while they return to be attractive when the point comes out of the visual hull. The amplitude of the external forces is progressively reduced during the chattering around the contour of the visual hull to guarantee that a dynamic equilibrium between external and elastic forces is quickly reached. The current reconstruction surface is buffered and made available, with a fixed sample time, to the planner. This moves the fingers, floating on this current available reconstructed surface at an imposed security distance, according to suitable quality measures. Due to the consequent motion floating effect around the object, the proposed method has been called *Floating Visual Grasp*. The fingertips keep moving toward a local minimum on the current reconstructed surface as long as it continues to change due to the updates provided by the reconstruction process. Quality measures considering both hand and grasp proprieties are adopted for the local planner: the directions of the finger motion leading toward grasp configurations that are not physically reachable or causing collisions or loss of hand manipulability are discarded. Moreover, a discretized version of the quality measure proposed in [15] is applied to the survived possible motion directions to select those leading toward an optimal (in a local sense) grasp configuration. Notice that many other quality measures may be chosen in substitution of those proposed in [15], without affecting the the proposed framework.

Due to the intrinsic smoothness and safety with respect to collisions of the planned trajectories, the execution of the grasp can also be executed in real-time during the trajectory generation. This allows the proposed algorithm to be used for kinematic control purposes.

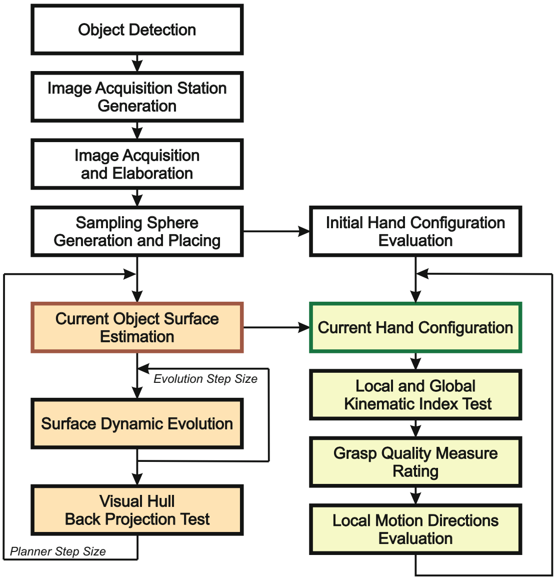

Object Detection
Image Acquisition Station Generation
Image Acquisition and Elaboration
Sampling Sphere Generation and Placing
Initial Hand Configuration Evaluation
Current Object Surface Estimation
Current Hand Configuration
Evolution Step Size
Surface Dynamic Evolution
Local and Global Kinematic Index Test
Grasp Quality Measure Rating
Visual Hull Back Projection Test
Local Motion Directions Evaluation
Planner Step Size

Fig. 1 Block diagram of the visual grasp algorithm.

Simulations results are presented to show the performance of the proposed algorithm.

2 Floating Visual Grasp Algorithm

The block diagram in Fig. 1 shows the data flow and the main elaboration steps of the proposed visual grasp algorithm. The elaboration processes may be arranged into three main groups: some *preparation steps*, the *object surface reconstruction algorithm*, and the *local grasp planner*.

The preparation steps of the algorithm start with a detection algorithm, that is based on a classical blob analysis, to evaluate the presence of an object in the field of view of the camera. Successively, by holding the optical axis perpendicular to the plane where the object has been detected, the camera is moved until the optical axis intercepts the centroid of the object. At the end of this step, the camera is exactly over the unknown object and ready to start the image acquisition process. Moreover, during this step, a rough estimation of the object center of mass is evaluated using the centroid of the object shape extracted from some images.

The image acquisition stations are chosen as illustrated in Fig. 2, while the concerning image acquisition step is carried out as follows: 1) an image is acquired from the top of the object; 2) a subset of n_1 images is taken from camera stations equally distributed over a circular path of radius r_1, with the optical axis of the camera pointing to the estimated center of the object and forming an angle α_1 with respect to the revolution axis $\mathbf{z}$; 3) a subset of n_2 images is acquired as in 2), but

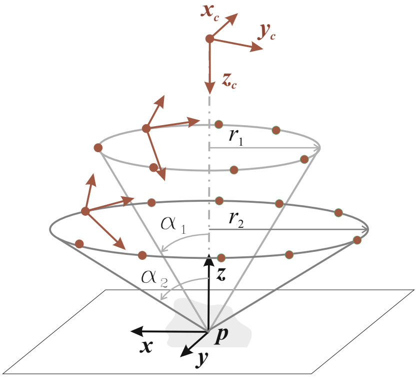


Fig. 2 Camera stations (bullets) and trajectories of the camera during image acquisition.

using a radius r_2 and an angle α_2. In the following, the total number of acquired images will be denoted as $n = n_1 + n_2 + 1$.

At this point, a blob analysis technique is employed to determine the silhouette of the object for each image. Each silhouette is also improved using suitable filtering techniques (e.g. dilatation and erosion iterative process) to reduce the effects of the image noise, and the centroid of the corresponding blob is evaluated. Then, the center of mass of the object, assuming homogeneous mass distribution, is estimated using a least-squares triangulation method.

On the basis of the dimension of each silhouette, the radius r_s of a 3D spherical surface that may contain the object is estimated, adding a suitable safety margin to the object estimated dimension. Finally, the *reconstruction sphere* with radius r_s, centered at the estimated center of mass of the object, and sampled with a number of n_s points is built. The position of each sample on the sphere is chosen in a way to achieve an initial uniform distribution of the reconstruction points, avoiding a thickening of points around the two poles (north and south) of the sphere.

A virtual mass is associated to each sample point of the reconstruction spherical surface, and four links are imposed with the closest cross points interposing springs. The resulting elastic reconstruction surface is shown in Fig. 3, where it is noticeable that the links between the masses dived the sphere into a certain number of parallels and meridians.

The initial grasp configuration of the hand can be set on the basis of the initial reconstruction sphere. In this paper, a three-fingered hand and point contact type at the fingertips are considered. Hence, a direct correspondence between the position of a point on the sphere and the position of each finger of the hand can be assumed. Due to the symmetry of the sphere, an infinite number of grasp configurations, ensuring force-closure grasp, could be initially selected. To this purpose, it is well known that a three contact grasp of a sphere is force-closure if the contact points are 120° apart on the same plane (neglecting moments and transversal forces). Therefore, the plane parallel to the floor halving the sphere, corresponding to the parallel with the maximal area that is graspable with respect to the hand kinematics, is chosen. Finally,

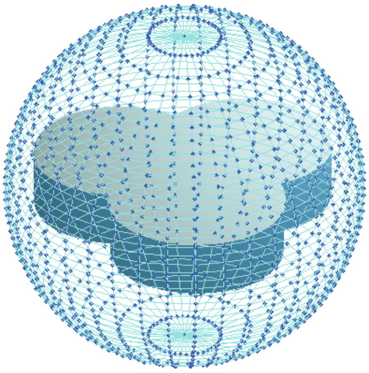


Fig. 3 Elastic reconstruction sphere surrounding the object.

three points 120° apart are selected on this parallel according to the orientation of the major and minor axis of the silhouette observed from the station just over the object –the observation station characterized by an optical axis perpendicular to the parallels of the sphere–.

At this point, both the object model reconstruction process and the local planner start in parallel and cooperate to the final goal. In particular, as shown in Fig. 1, the reconstruction algorithm updates in real-time the estimation of the current reconstructed object surface, while the local planner, on the basis of the current estimation, computes the fingers trajectories toward the current local optimal configuration for the grasp. Notice that a force optimization algorithm, e.g. [2], could be used for a proper distribution of grip forces.

These two parallel processes are independent and can be allocated under two different threads and, in a multi-processor system, also on different CPUs. In other words, the proposed method exhibits an intrinsic capability to be run in parallel. More details of these two crucial steps are provided in the next two sections.

3 Object Surface Reconstruction

The object surface reconstruction algorithm employed in this paper is an evolution of the method proposed in [14].

As described in the previous sections, from the set of n silhouettes of the object a reconstruction spherical surface is created and sampled by points. A virtual mass is associated to each sample point, and four links are imposed with the closest cross points with springs, resulting in a cross reticular topology for the reconstruction surface (see Fig. 4). The two poles of the sphere are connected with all the masses of the nearest parallel of the resulting spherical reticulum.

Each parallel of the sphere should have the same n_m number of points, corresponding to the number of meridian, allowing the construction of a cross reticulum fully linked. In other words, for each point, the existence of a couple of corresponding points on the closest parallels of the spherical grid is guaranteed. Without loss

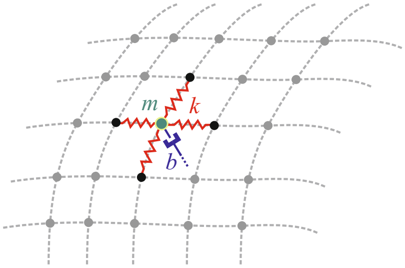


Fig. 4 Cross network topology of the reconstruction surface with the virtual mass, springs and spatial damper of the i-th sample points.

in generality, the number of parallels n_p is chosen equal to the number of meridians $n_p = n_m = \sqrt{n_s - 2}$. To avoid that the parallel nearest to the poles determines an unnecessary initial thickening of sample points around the poles, a suitable angular distribution of the parallels has been carried out, reducing (augmenting) the density of the parallels near the poles (equator).

The model of the system, defining the motion of each sample point of the reconstruction surface, is defined by the following dynamic equations:

$$m\ddot{\boldsymbol{x}}_{i,j} + b\dot{\boldsymbol{x}}_{i,j} + k(4\boldsymbol{x}_{i,j} - \boldsymbol{x}_{i-1,j} - \boldsymbol{x}_{i,j+1} - \boldsymbol{x}_{i+1,j} - \boldsymbol{x}_{i,j-1}) = \boldsymbol{f}_{i,j}, \tag{1}$$

for $i = 1, \ldots, n_m$ and $j = 1, \ldots, n_p$, denoting with $\boldsymbol{x}_{i,j}$ the position in the workspace of the sampling point at the intersection of the i-th meridian with the j-th parallel –reticular position (i, j)–, with m, k, and b the mass associated to the point, the constant spring linking the point with its nearest four points of the cross of the reticulum, and the constant spatial damper, respectively. Notice that subscript $i = j = 0$ ($i = n_m + 1$ and $j = n_p + 1$) for the representation of the four nearest points in the reticulum corresponds to $i = n_m$ and $j = n_p$ ($i = j = 1$), respectively.

The term $\boldsymbol{f}_{i,j}$ is the external force acting on the mass associated to the sample point (i, j), attractive with respect to the border of the visual hull, which is progressively reduced once the corresponding point comes in or out from the visual hull:

$$\boldsymbol{f}_{i,j} = \alpha_{i,j}(t_{i,j}) F_a \boldsymbol{n}_{i,j}, \tag{2}$$

where $\boldsymbol{n}_{i,j}$ is the unit vector pointing from the current point (i, j) to the estimated centroid of the object, that defines the direction of the force, and $\alpha_{i,j}(t_{i,j})F_a$ is the amplitude of the force. Also, F_a the maximum force module and $\alpha_{i,j}(t_{i,j}) \in (-1, 1]$ a discrete numerical sequence of scale factors defined as follow:

$$\alpha_{i,j}(t_{i,j}) = -\varepsilon \alpha_{i,j}(t_{i,j} - 1) \tag{3}$$

where $\varepsilon \in (0, 1)$ (e.g. $\varepsilon = 0.9$), $\alpha_{i,j}(0) = 1$, and $t_{i,j} = 0, \ldots, \infty$ is a discrete step time incremented every time the point (i, j) comes in or out of the visual hull.

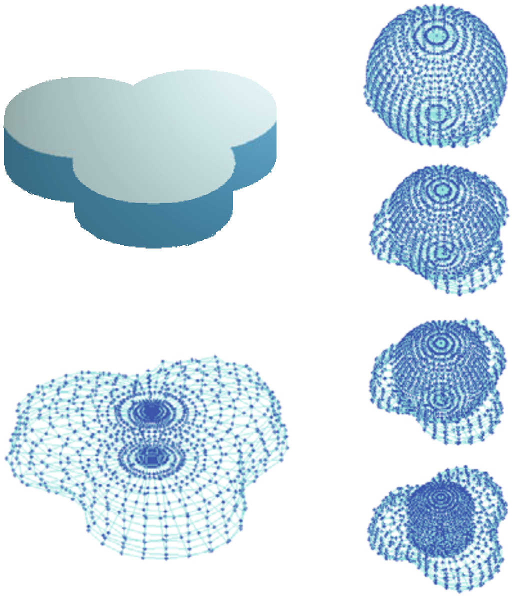


Fig. 5 Steps of the object surface reconstruction process.

The two poles have to be treated separately due to their topological peculiarity. The previous model becomes

$$m\ddot{\boldsymbol{x}}_n + b\dot{\boldsymbol{x}}_n + k(n_m\boldsymbol{x}_n - \sum_{j=1}^{n_m}\boldsymbol{x}_{1,j}) = \boldsymbol{f}_n \tag{4}$$

for the north pole, and

$$m\ddot{\boldsymbol{x}}_s + b\dot{\boldsymbol{x}}_s + k(n_m\boldsymbol{x}_s - \sum_{j=1}^{n_m}\boldsymbol{x}_{n_p,j}) = \boldsymbol{f}_s \tag{5}$$

for the south pole, where the subscripts n and s indicate quantities refereed to the north and south pole, respectively.

The stability of the system, for any non-trivial initial condition of the sphere, leads the reconstruction elastic surface to contract toward its center of mass until the visual hull is intersected. The dynamic evolution of the system reaches the equilibrium when the shape of the surface produces a dynamic equilibrium between the elastic forces generated by the grid and the repulsive forces depending on the contours of the visual hull. The result is that the initial elastic reconstruction sphere shapes itself on the unknown object.

The accuracy of the reconstruction process depends on the number of views, on the distribution of the observation stations, and on the density of points of the reconstruction sphere. On the other hand, the computational time of the algorithm increases if n and/or n_s are increased. However, considering that the final goal of the process is the object grasping and not the model reconstruction, which can be

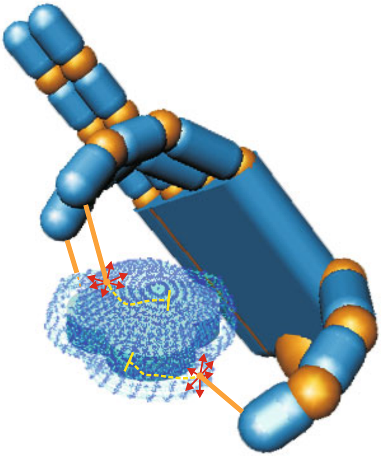


Fig. 6 Floating visual grasp.

considered as a secondary outcome of the proposed method, the accuracy of the reconstruction process needs only to be adequate for the requirements of the grasp planner algorithm.

In Fig. 5 some images showing intermediate steps of the reconstruction algorithm of a synthesized object are shown, with parameters: $\alpha_1 = 45°$, $\alpha_2 = 80°$, $n_1 = 4$, $n_2 = 8$, and $n_s = 1602$.

4 Local Grasp Planner

The current estimation of the object surface, which is stored in a buffer and is employed by the local grasp planner for updating the fingertips trajectories, is continuously updated during the dynamic evolution of the elastic surface. The local grasp planner, in accordance with the current reconstructed object surface, generates the fingertips trajectories in a floating manner, keeping a fixed *floating safety distance* δ_f between the fingers and the corresponding sample point along the outgoing normals, on the basis of suitable quality indices (see Fig. 6).

Namely, starting from the initial grasp configuration, chosen as described in the previous sections, the planner generates the motion of the fingers from the current position to a new point of the updated surface. In particular, the contact points of the grasp are first “virtually” moved to the updated surface, achieving an initial “target” grasp configuration. Then, for each contact point of the current target grasp configuration, the contour made by the eight neighboring points of the surface is selected. Considering the contours of all the contact points, the set of all the combinations of possible reachable grasp configurations is evaluated on the basis of suitable quality measures. If the current grasp configuration of the set has a value better than the value of the target configuration, this is chosen as the new target grasp configuration, establishing “de facto” the motion direction of each finger, as shown in Fig. 7.

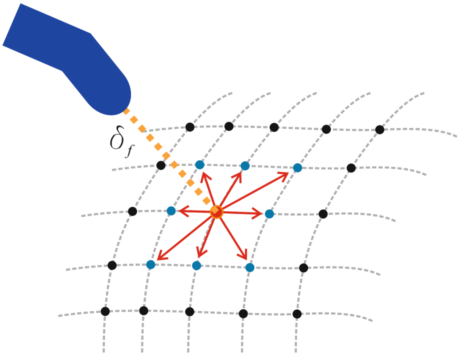


Fig. 7 Contour of neighbor points of the current target grasp point.

The process is repeated in a recursive manner until there are no more improvements of the quality measures, and it restarts at the next step time. The whole process ends when the object reconstruction algorithm reaches an equilibrium and the planner computes the final grasp configuration. Hence, the safety floating distance δ_f is progressively reduced to achieve the desired grasp action.

The floating distance is used to avoid collisions of the fingers with the object during the object reconstruction and approach, and before the final grasp configuration is reached. Moreover, the final progressive reduction of the floating distance implies that each fingertip moves perpendicularly to the surface.

Notice that only few points are considered as candidates for the next grasp configuration, so that the number of combinations to be inspected is limited, resulting in a computationally fast algorithm; moreover, a certain number of grasp configurations are discarded during the evaluation process. Namely, a *local kinematic index* is used to discard all the candidate grasp configurations that cannot be reached. Then, a *global kinematic index* is adopted to discard all the configurations causing finger collisions or lack of manipulability for the hand. Finally, a *grasp quality measure* is applied to the remaining configurations to evaluate possible improvements of the grasp quality.

The computational efficiency of the local planner jointed with the possibility to be executed in parallel with the reconstruction algorithm, eventually on two different elaboration units, resulting in a faster solution with respect to traditional methods, which first reconstruct the object and then plan the grasp on the whole knowledge of the environment.

In the next subsections, the quality indices and the finger trajectory planner are presented.

4.1 Local and Global Kinematic Indices

On the basis of the finger kinematics, the local kinematic index allows discarding all the candidate contact points that cannot be reached. With reference to a single contact point and finger, the kinematic test is carried out for all the contour points (see Fig.7). Namely, for each point, finger joints are computed using an inverse kinematics algorithm. Hence, those points for which joint limits are exceeded, or that are too close to a kinematic singularity, are discarded. This latter condition is evaluated on the basis of the condition number of the finger Jacobian. Avoiding singularities in the finger Jacobian allows being far from hand singularities.

A standard CLIK algorithm [21] is adopted to compute the inverse kinematics; in particular, the scheme based on the transpose of the Jacobian has been used to achieve a faster computation, together with a Singular Value Decomposition technique for the evaluation of the Jacobian condition number.

For the remaining points of the contour, the global kinematic index computes the distance between the fingers corresponding to all the possible grasp configurations. Hence, all the configurations for which the distances are under a given safety threshold, are discarded.

4.2 Grasp Quality Measure

The grasp quality is evaluated only for the configurations that are left after the kinematic tests. The method proposed in [15] is adopted, suitably modified to cope with the discretization of the grasp configurations, assuming neglectable moments and transversal forces.

Let us denote with $\boldsymbol{w} = \begin{bmatrix} \mathbf{f}^{\mathrm{T}} & \boldsymbol{\mu}^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$ the wrench vector collecting the force $\mathbf{f}$ and moment $\boldsymbol{\mu}$. Assuming that the finger forces are applied along the direction normal to the object surface, the force direction is specified only by the contact point.

Let $\mathscr{W}$ denote the space of wrenches, $\mathscr{W}_f \subset \mathscr{W}$ the space of unit forces acting in the grip plane, that is the plane containing the three contact points, through the center of grip, $\mathscr{W}_{\perp\mu} \subset \mathscr{W}$ the space of pure moments acting along the direction perpendicular to the grip plane. Moreover, let $\mathbf{g}^{-1}(-\mathbf{w})$ denote the set of finger forces which can resist the external wrench $\mathbf{w}$.

Finally, consider the quantity

$$Q_1 = \min_{\mathbf{w}\in\mathscr{W}_f} \left(\max_{\mathbf{f}\in\mathbf{g}^{-1}(-\mathbf{w})} \frac{1}{\|\,\mathbf{f}\,\|_f} \right), \tag{6}$$

which is a measure of the grasp ability to resist unit forces in the grip plane, and the quantity

$$Q_2 = \min_{\mathbf{w}\in\mathscr{W}_{\perp\mu}} \left(\max_{\mathbf{f}\in\mathbf{g}^{-1}(-\mathbf{w})} \frac{1}{\|\,\mathbf{f}\,\|_f} \right), \tag{7}$$

which is a measure of the grasp ability to resist unit moments normal to the grip plane.

The optimal grasp proposed in [15] is defined as the grasp that maximizes Q_2 among all grasps which maximize Q_1.

It can be proven (see [15]) that the optimum grasp with three fingers in a 2-D case under the above optimal criterion is reached when the normal forces are symmetric, with directions spaced 120° apart. Moreover, this grasp maximizes also the size of the outer triangle, defined as the triangle formed by the three lines perpendicular to the normal finger forces passing through the respective contact points. Under the same criterion, the optimum grasp with three fingers in a 3-D case is achieved when the maximum circumscribing prism-shaped grasp, that has the largest outer triangle, is selected among the grasps where the normal finger forces lie within the same grip plane and are in equilateral configuration.

Therefore, to reach the optimum in the 3-D case with three fingers, the planner has to seek three points in equilateral configuration on the object surface, so that the normal forces lie in the same grip plane, and for which the circumscribing prism grasp is maximum.

In the case presented in this paper, since the reconstructed object surface is sampled by points/masses, the above method cannot be directly applied. Differently from the continuous case, due to the presence of a finite set of sampled points, the existence of a "grip plane" containing all the normal forces is not guaranteed. This is mainly due to the fact that, because of the discretization, the surface normals are an approximation of the real ones. Considering that the optimal criterion requires that the desired normals have to be spaced 120° apart, a discretized implementation of the method of [15] is proposed here.

For each candidate configuration of three grasp points, the normal directions are estimated on the basis of the available point-wise approximation of the surface. Then, the unit vector normal to the grip plane containing the three points is evaluated. Denoting with ϑ_j the angle between the direction of the normal force applied to point j and the direction normal to the grip plane, a Coplanarity Error Index (CEI) can be defined as follow:

$$CEI = \frac{\sum_{j=1}^{3} \mid \vartheta_j - 90^\circ \mid}{3}. \tag{8}$$

Obviously, the closer CEI to zero, the more the normal forces lie in the same plane. The definition of a threshold Φ_{CEI} allows discarding all those configurations having a value of CEI higher than Φ_{CEI}; hence, all the remaining grasp configurations are assumed to have forces lying in the same grip plane and can be further processed.

The next step consists in looking for an equilateral grasp configuration. To this aim, for each grasp configuration, the unit vector normal to the object surface at each contact point is projected on the grip plane. Denoting with φ_j the angle between these projections for each of the 3 couple of points of the considered configuration, an Equilateral Error Grasp Index ($EEGI$) can be defined as:

$$EEGI = \frac{\sum_{j=1}^{3} \mid \varphi_j - 120^\circ \mid}{3}. \tag{9}$$

Clearly, the closer $EEGI$ to zero, the nearer the configuration to an equilateral grasp. The definition of a threshold Φ_{EEGI} allows discarding all those configurations with a value of $EEGI$ higher than Φ_{EEGI}; hence, all the remaining grasp configurations are assumed to be equilateral.

Among all the equilateral configurations, the maximum circumscribing prism has to be found; if the grasp configuration associated with the largest prism is different from the current target configuration, this is taken as the new grasp configuration.

Notice that, in the case that the grasp configuration changes, the whole process starts again with the new contact points, by considering the new contours and applying the complete sequence of index-based tests starting from the kinematic ones. The algorithm stops if the best grasp configuration remains unchanged at the end of the optimization process, or in the case that all the candidate grasp configurations are discarded during the process.

4.3 Finger Trajectory Planner

The local grasp planner produces a sequence of intermediate target grasp configurations at each iteration of the object reconstruction algorithm. These end with the optimal grasp configuration (in local sense), as illustrated in Fig. 6. The intermediate configurations are used to generate the finger paths.

Namely, the sequence of intermediate configurations is suitably filtered by a spatial low-pass filter in order to achieve a smooth path for the fingers on the object surface. To this purpose, notice that only the final configuration needs to be reached exactly, while the intermediate configurations can be considered as via points.

With respect to the smooth paths through the points of the filtered configurations, the actual finger paths generated by the finger trajectory planner keep a floating distance δ_f along the normal to the surface (as explained in Section 4). This feature produces a floating effect of the fingers over the reconstructing object surface during the reconstruction process while they move according to the deformation of the reconstruction sphere. When the final configuration is reached, this safety distance is progressively reduced to zero, producing the desired grasp action.

5 Simulations

The proposed method has been tested with simulations using synthesized objects. The first used object is shown in Fig. 5, while for the trajectory planning only the first three fingers of the virtual hand shown in Fig. 6 have been considered.

The dynamic parameters of the reconstruction sphere have been chosen as follows: the total mass of the sphere $M = 10^{-3}\ kg$, $k = 0.3 \cdot 10^{-3}\ N/m$, $b = 0.09 \cdot 10^{-3}\ Ns/m$, and $F_a = 5\ N$.

By setting $\Phi_{CEI} = 15^\circ$ and $\Phi_{EEGI} = 10^\circ$, the grasp configuration and the finger trajectories are those of Fig. 8. The final grasp configuration (which can be proven to be the global optimal grasp configuration) is characterized by the values $CEI = 13^\circ$ and $EEGI = 2.2^\circ$.

A prism with smooth lateral corners has also been considered. The grasp configuration and the corresponding trajectories are shown in Fig. 9. The values $CEI = 9.9^\circ$ and $EEGI = 1.04^\circ$ are obtained in the final configuration. Remarkably, an equilateral symmetry is achieved in the final grasp configuration: two fingers are placed on the smooth corners, and the other finger is placed in the middle of the opposite

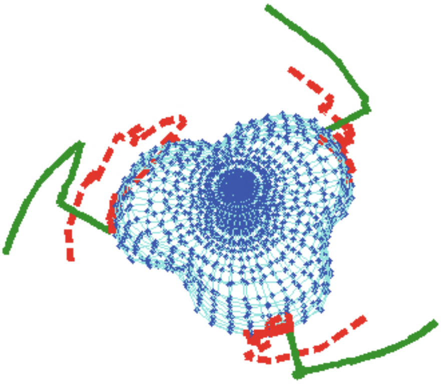


Fig. 8 Finger trajectories evaluated by the local grasp planner (continuous lines) and the corresponding sequence of grasp points on the reconstructed object surface (dotted lines).

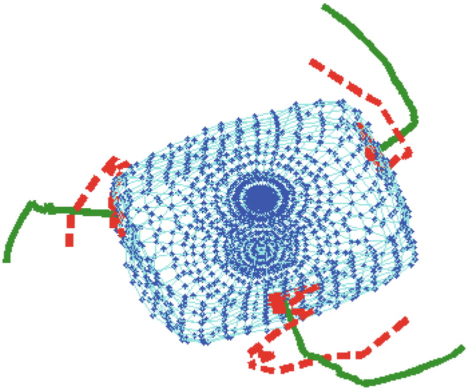


Fig. 9 Finger trajectories evaluated by the local grasp planner (continuous lines) and the corresponding sequence of grasp points on the reconstructed surface for the smooth prism (dotted lines).

surface. This configuration corresponds to an opposite grasp ensuring force closure; as before, it can be proven that this grasp configuration is optimal also in a global sense, although the proposed approach can only guarantee that a local minimum is achieved.

6 Conclusion and Future Work

6.1 Conclusion

A new method for fast visual grasp of unknown objects has been presented. The proposed method is composed of a fast iterative object surface reconstruction algorithm and of a local optimal grasp planner, which evolve in a synchronized parallel way. An eye-in-hand camera is adopted to acquire the images used by the reconstruction algorithm. A reconstruction elastic sphere sampled with points furnished of mass and linked with each other by springs, which is virtually placed around the object, dynamic evolves collapsing towards the visual hull of the object. An attractive force pushes each mass to the visual hull and is progressively reduced when the visual hull has been reached. During the reconstruction process, the planner moves the fingertips, floating on the current reconstructed surface at a safety distance, according to local and global kinematic indices and grasp quality measures. The effectiveness of the proposed method has been shown in simulation.

6.2 Future Work

The adoption of suitable pre-shaping techniques, that could be helpful also for the determination of the grasp as explained in [23], for the choice of the initial grasp configuration, which is based on visual information, may be a further improvement of the proposed algorithm. Moreover, quality indices connected to the tasks to be performed by the hand with the object and the adoption of other kinematic constraints, e.g. collision avoidance of the object with the hand's palm, could be considered. Finally, the extension of the proposed approach to the case of more than three fingers or to bi-manual manipulation, with the adoption of suitable quality measures, will be investigated.

Acknowledgements. The research leading to these results has been supported by the DEXMART Large-scale integrating project, which has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement ICT-216239. The authors are solely responsible for its content. It does not represent the opinion of the European Community and the Community is not responsible for any use that might be made of the information contained therein.

References

1. Borst, C., Fischer, M., Hirzinger, G.: Calculating hand configurations for precision and pinch graps. In: IEEE/RSJ Confonference on Intelligent Robots and Systems, Lausanne (2002)
2. Buss, M., Hashimoto, H., Moore, J.B.: Dexterous hand grasping force optimization. IEEE Transaction on Robotics and Automation 12(3), 406–418 (1996)
3. Chinellato, E., Fisher, R.B., Morales, A., del Pobil, A.P.: Ranking planar grasp configurations for a three-fingered hand. In: IEEE International Conference on Robotics and Automation,Taipei (2003)
4. Cipolla, R., Blake, A.: Surface shape from the deformation of apparent contours. Internation Journal of Computer Vision 9(2), 83–112 (1992)
5. Cohen, L.D.: On active contour models and balloons. Computer Vision, Graphics, and Image Processing: Image Understanding 53(2), 211–229 (1991)

6. Dyer, C.R.: Volumetric scene reconstruction from multiple views. In: Davis, L.S. (ed.) Foundations of Image Analysis. Kluwer, Boston (2001)
7. Ferrari, C., Canny, J.: Planning optimal grasps. In: IEEE International Conference on Robotics and Automation,Nice (1992)
8. Fischer, M., Hirzinger, G.: Fast planning of precision grasps for 3D objects. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, Grenoble (1997)
9. Franco, J.-S., Boyer, E.: Exact polyhedral visual hulls. In: British Machine Vision Conference (2003)
10. Guan, Y., Zhang, H.: Kinematic feasibility analysis of 3D grasps. In: IEEE International Conference on Robotics and Automation, Seoul (2001)
11. Hester, R.D., Cetin, M., Kapoor, C., Tesar, D.: A criteria-based approach to grasp synthesis. In: IEEE International Conference on Robotics and Automation, Detroit (1999)
12. Laurentini, A.: How far 3D shapes can be understood from 2D silhouettes. IEEE Transactions on Pattern Analysis and Machine Intelligence 17(2), 188–195 (1995)
13. Li, Z., Sastry, S.S.: Task-oriented optimal grasping by multifingered robot hands. IEEE Journal of Robotics and Automation 4(1), 32–44 (1988)
14. Lippiello, V., Ruggiero, F.: Surface model reconstruction of 3D objects from multiple views. In: IEEE International Conference on Robotics and Automation, Kobe (2009)
15. Mirtich, B., Canny, J.: Easily computable optimum graps in 2-D and 3-D. In: IEEE International Conference on Robotics and Automation, San Diego (1994)
16. Perrin, D., Smith, C.E., Masoud, O., Papanikolopoulos, N.P.: Unknown object grasping using statistical pressure models. In: IEEE International Conference on Robotics and Automation, San Francisco (2000)
17. Platt, R., Fagg, A.H., Grupen, R.A.: Manipulation gaits: sequences of grasp control tasks. In: IEEE International Conference on Robotics and Automation,New Orleans (2004)
18. Pollard, N.S.: Synthesizing grasps from generalized prototypes. In: IEEE International Conference on Robotics and Automation, Minneapolis (1996)
19. Prakoonwit, S., Benjamin, R.: 3D surface point and wireframe reconstruction from multiview photographic images. Image and Vision Computing 25, 1509–1518 (2007)
20. Shimoga, K.B.: Robot grasp synthesis algorithms: A survey. International Journal of Robotics Research 15(3), 230–266 (1996)
21. Siciliano, B., Sciavicco, L., Villani, L., Oriolo, G.: Robotics: Modelling, Planning and Control. Springer, London (2009)
22. Suarez, R., Roa, M., Cornella, J.: Grasp quality measures, Technical Report IOC-DT-P-2006-10, Universitat Politecnica de Catalunya, Institut d'Organitzacio i Control de Sistemes Industrials (2006)
23. Wren, D., Fisher, R.B.: Dextrous hand grasping strategies using preshapes and digit trajectories. In: IEEE International Conference on Systems, Man and Cybernetics, Vancouver (1995)
24. Xu, C., Prince, J.L.: Snakes, shapes, and gradient vector flow. IEEE Transactions on Image Processing 7(3) (1998)
25. Yoshikawa, T., Koeda, M., Fujimoto, H.: Shape recognition and grasping by robotic hands with soft fingers and omnidirectional camera. In: IEEE International Conference on Robotics and Automation, Pasadena (2008)
26. Yoshimi, B.H., Allen, P.K.: Visual control of grasping and manipulation tasks. In: IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, Las Vegas (1994)

Generality and Simple Hands

Matthew T. Mason, Siddhartha S. Srinivasa, and Andres S. Vazquez

Abstract. While complex hands seem to offer generality, simple hands are more practical for most robotic and telerobotic manipulation tasks, and will remain so for the foreseeable future. This raises the question: how do generality and simplicity trade off in the design of robot hands? This paper explores the tension between simplicity in hand design and generality in hand function. It raises arguments both for and against simple hands; it considers several familiar examples; and it proposes a concept for a simple hand design with associated strategies for grasping and object localization. The central idea is to use knowledge of stable grasp poses as a cue for object localization. This leads to some novel design criteria, such as a desire to have only a few stable grasp poses. We explore some of the design implications for a bin-picking task, and then examine some experimental results to see how this approach might be applied in an assistive object retrieval task.

1 Introduction

This is the first paper from a project that aims to develop robot grippers that are simple, yet also general and practical. By "simple", we mean hands with a few actuators, a few simple sensors, and without complicated mechanisms, so that the whole hand would be small, light, and inexpensive.

Matthew T. Mason
Carnegie Mellon University, Pittsburgh, PA
e-mail: matt.mason@ri.cmu.edu

Siddhartha S. Srinivasa
Intel Research, Pittsburgh, PA
e-mail: siddhartha.srinivasa@intel.com

Andres Vazquez
Universidad de Castilla-La Mancha, Spain
e-mail: andress.vazquez@uclm.es

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 345–361.
springerlink.com

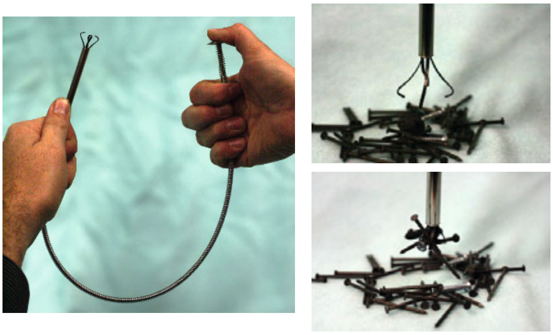


Fig. 1 The common "pickup tool" is very simple, but also very effective in achieving stable grasps over a broad class of shapes. Four fingers of spring steel are compliantly driven by a single actuator.

Can a hand be both simple and general? Perhaps generality requires complexity. Some arguments in favor of complexity are:

- Grippers for manufacturing automation are often simple and specialized, perhaps designed to grasp a single part. Human hands, in contrast, are complex and general;
- Hands grasp by conforming to the object shape. Motion freedoms are a direct measure of a hand's possible shape variations.
- Beyond grasping, many tasks benefit from more complexity. Manipulation in the hand, and haptic sensing of shape, to mention two important capabilities, benefit from more fingers, more controlled freedoms, and more sensors.
- The most general argument is: Design constraints have consequences. Restricting actuators, sensors, and fingers to low numbers eliminates most of the hand design space.

However, there *are* simple but general grippers: humans using prosthetic hooks, teleoperated systems with simple pincer grippers, or the simple pickup tool shown in Fig. 1. We conclude that while there is a tradeoff between simplicity and generality, the details of that tradeoff are important and poorly understood. Simple grippers offer a level of generality that is yet untapped in autonomous robotic systems.

To explore the tradeoff between simplicity and generality, we list capabilities required of a general purpose gripper (Section 1.2), and discuss clutter (Section 1.3). However, it simplifies the discussion if we begin with a specific example, the Pickup Tool of Fig. 1, and a simple hand concept inspired by the pickup tool (Section 1.1). The rest of the paper describes simulation and analysis of the simple hand grasping a variety of shapes, and an experimental study motivated by the simple hand concept.

1.1 Let the Fingers Fall Where They May

This section outlines our approach, illustrated by a classic robotic manipulation problem: picking a single part from a bin full of randomly posed parts. Our approach is inspired by the pickup tool shown in Fig. 1, which is very effective at capturing one or several parts from a bin, even when operated blindly. Rather than attempting to choose a part, estimate its pose, and calculate a stable grasp, we propose to execute a blind grasp, let the gripper and object(s) settle into a stable configuration, and only then address the problem of estimating the object pose (as well as whether a single object was captured).

The main problem addressed by this paper is how to determine whether a single object was captured, and how to estimate the object pose. We propose to use a table of the stable poses with corresponding finger encoder values, produced offline either by experiment or in simulation.

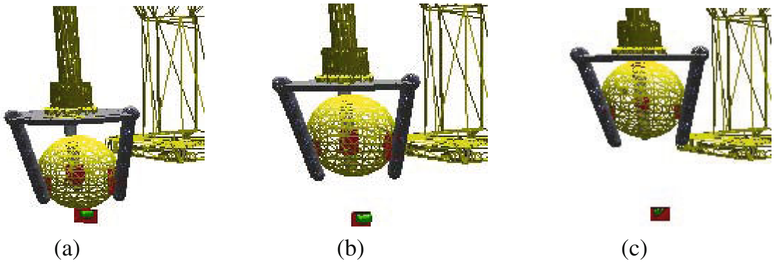
Our initial gripper concept departs from the pickup tool that inspired it. For industrial bin picking the pickup tool has some deficiencies, including a tendency to capture several parts at a time, in unpredictable poses. And while the fingers of spring steel, with the bent tips, and the motion emerging along their lengths are all very effective and interesting, we want to begin with a design that is easier to analyze and simulate, and which supports estimation of pose. Our initial design has rigid cylindrical fingers attached to a disk-shaped palm by revolute joints with encoders (Fig. 2). As with the pickup tool, all three fingers are compliantly coupled to a single actuator.

For the bin-picking task the goal is to retrieve a single familiar object from a bin of objects, and to accurately estimate its pose in the hand. For the sphere shown in the figure, estimation of position would suffice. For a non-spherical object we would require orientation as well as position. The key elements of the approach are:

- Low-friction palm and fingers so that for irregular objects there are only a few stable grasp configurations;
- Blind or nearly blind grasping motions;
- A table of stable grasp configurations and corresponding encoder values to determine whether a single object is present, and to estimate its pose;
- Offline training of pose and singulation classifiers, either in simulation or in the real world;
- Iteration of a reach, grasp, withdraw, and classify strategy until a successful grasp of a single object in a recognized pose is achieved.

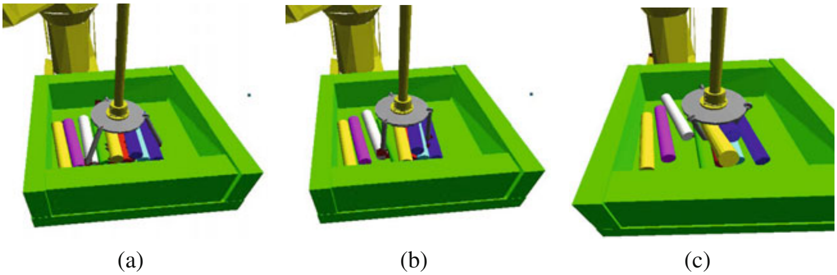
Figures 2 and 3 show a dynamic simulation of the concept, applied to a bin picking problem. The system performed blind grasps, identified successful single object grasps, and estimated pose up to symmetries. We also tried some variations: shaking the bin, and sweeping the bin with the fingers, preparatory to the grasping operation. Both behaviors improved performance, and the two behaviors together brought the success rate close to 50%. More advanced versions of this scenario would include the use of visual guidance, of strategically planned pregrasp poses, and a variety of motion and perceptual strategies that could be employed.

The central idea is to use knowledge of stable poses to determine whether a single part has been captured, and to localize that part in the grasp. The general idea is well known in parts orienting research (see [15, 11] for two examples) and has even been used in the context of simple hands ([18, 10]). Yet we believe the idea can be taken



(a) (b) (c)

Fig. 2 Grasping of a Sphere



(a) (b) (c)

Fig. 3 A blind grasp capturing two cylinders. The failure is detected and the operation will be repeated. The cylinders were heaped up by a blind sweeping operation prior to the grasp.

further, perhaps even leading to practical designs working across a broad range of manipulation tasks.

For the approach to work in its simplest form, the stable poses must map to isolated points in the space of encoder values. Hence the primary focus of the paper is the set of stable poses. We consider how hand design parameters affect the stable pose space, and we also examine the stable pose space for a typical assistive home robot in an object retrieval task, in Sections 2 and 3. First, we return to our discussion of generality in hand function.

1.2 Dimensions of Grasping

The bin-picking application described above is the framework that motivates several problems, some of which are addressed in the rest of the paper. In this section we place those problems in a broader context. While a precise definition of generality in hand function is elusive, we can at least identify the dimensions. What are the requirements of a grasp? What are the problems that a grasp might solve?

Capture and *Stability*. The main theme of grasping research seems to be stability, but capture seems equally important.

In-hand manipulation and *Grasp adjustment.* Controlled motion of the object in the grasp. Typical prior work has employed fingertip grasps with several actuators to achieve controllability, but there are other strategies including, for example, tapping two chopsticks against a tabletop to align the ends.

Clutter and *Singulation.* Clutter refers to everything that might limit access to the object. See Section 1.3 for further discussion. Singulation means extracting one part at a time.

Shape diversity and *Shape uncertainty.* Shape diversity: Does the grasp work over a broad range of shapes? Shape uncertainty: does the grasp work when the shape isn't entirely known? There is a difference. A net can capture an object without even knowing what the shape is. A modular fixturing kit can be configured to grasp a wide range of shapes, but not without knowledge of the shape.

Localization, Object recognition, Shape estimation. Localization means estimation of object pose in the hand. Object recognition assumes there is some class of parts, perhaps even finite, which includes the present object.

Placing. By "placing" we refer broadly to a variety of downstream requirements: dropping onto a conveyor, placing into a dishwasher, throwing, assembly, handing off to another (human or robotic) hand, and so forth.

Other task specific requirements. There are many different applications of grasping, with wildly varying requirements. Some applications have product inspection processes integrated with a hand, such as checking electrical continuity of an automotive part. Others have unusual force or compliance requirements, such as grasping an object for a machining operation. Some applications may involve complex processes involving multiple objects, such as dealing cards.

We can use the above list to characterize different tasks. The bin picking application, by its nature, poses challenges in clutter and singulation. Our approach to it, employing a simple hand iteratively employing a blind strategy, entails additional challenges of capture and localization. Since we propose to use a single hand design over a range of parts, we introduce the shape diversity issue.

In contrast, assistive robotic object retrieval, the application addressed in Section 3, poses challenges in capture, clutter, and shape diversity. Typical households could not afford to have one robot to retrieve spoons, another to retrieve forks, and so on.

In this paper we focus on capture, clutter, stability, and pose estimation, but ultimately a general-purpose gripper must address the entire list of requirements. Then there are many additional pragmatic issues: cost, weight, ruggedness, and ease of programming. Those are perhaps the main motivation for the use of simple grippers.

1.3 Grasping versus Clutter

Clutter is an important element of both bin-picking and object retrieval, but in the course of this work we have realized that clutter is almost ubiquitous. Previous work has seldom addressed clutter explicitly, but clutter often affects the design of the robot, the choice of grasp, the robot path, in fact just about every aspect of a system.

First consider the effect of clutter on hand design. Suppose you are designing a hand from simple geometrical elements: points, lines, or planes. If you want to capture an isolated object, stabilize it, and estimate its location, infinite planes would be ideal. A plane sweeps out lots of space, and captures and stabilizes objects very economically. Four planes in a tetrahedral configuration could squeeze an object of arbitrary shape and reduce the feasible poses to a small set. This idea is not entirely impractical. The tilted tray of Grossman and Blasgen [11] used three planes, with gravity instead of a fourth plane, to provide a universal parts orienting system. You might also view a common table as a variant of the idea: a single plane moving through space with a constant acceleration of 1g to sweep up all objects placed above it, in cases where only three degrees of constraint and localization are required.

However, if you are dealing with clutter, planes are terrible. They sweep up everything and avoid nothing. For clutter, points would be ideal. These observations are captured in the table below. Higher dimensional elements are better for grasping; lower dimensional elements are better for avoiding clutter.

	grasping	clutter
points	bad	good
lines	okay	okay
planes	good	bad

This table suggests an impractical approach: a set of levitated independently controlled points that drift through the interstices of the clutter, approach the object, and then switch to a coordinated rigid mode to lift the object. Consider the paths that the points follow to reach the object. If the clutter is unmoving, then those paths remain clear, and we could use hyper-redundant fingers, i.e. snakes or tentacles, to reach the object. Lifting the object clear of the clutter is still an issue, but the idea does suggest a practical compromise to address the problem of grasping in clutter: use very thin fingers, approaching the object along their lengths. The idea is reflected in many common manipulation tools, such as tweezers and laporoscopic forceps. You might even view the pickup tool (Fig. 1) as a variant of the idea, a single long thin element from which several fingers deploy. These tools are all designed for high-clutter environments: the pickup tool for retrieving small parts dropped into difficult to access places, and forceps for the extreme clutter encountered in surgery.

Clutter and grasping are in opposition. Grasping is easy, if there is no clutter. Clutter is easily avoided, if no grasp is required. The problem arises from the need to grasp one object while avoiding others. The problem is not just to grasp, but to grasp selectively.

Almost every grasping problem involves clutter. Even for an object isolated on a table, the table is clutter. Perhaps the most clutter-free grasping problem is a door knob: an affordance specifically designed for grasping, mounted on a stalk to minimize clutter. The only cases involving less clutter are catching butterflies or cleaning a pool, where it is practical to sweep the entire nearby volume with a net.

1.4 Previous Work

There is a long history of research in robotic hands, and related issues in perception, planning, and control. See [13] for an early but still highly relevant overview, and [3] for a more recent overview. Generality and simple hands are particularly relevant issues for domestic and assistive robotics. See [7, 24, 19] for some examples.

Some of the earliest work in robotic manipulation exhibited surprisingly general manipulation with simple effectors. Freddy II, the Edinburgh manipulator used a parallel-jaw gripper, grasping a variety of shapes [1]. Freddy II is also one of the rare robotics research projects to address grasping in clutter. The Handey system also used a simple parallel-jaw gripper exhibiting impressive generality [17].

Our gripper concept is similar to Hanafusa and Asada's [12], who analyzed the stability of planar grasps using three compliant fingers. Our concept is further informed by the analysis of Baker, Fortune and Grosse [2] who developed some interesting observations on grasp stability by compliant fingers. Since that time, most previous work on stable grasping has assumed a rigid grasp, or has assumed compliance primarily at the finger tips [20, 16], or has addressed the choice of control strategies for complex grippers.

Our approach contrasts with that often referred to as "dexterous" hands: the use of several fingers, with several actuators per finger, with resulting freedom in the placement of fingers, programmable compliance, and fully controllable in-hand manipulation [23, 14, 4]. This contrast is also reflected in our choice of hard frictionless contacts, despite several advantages of soft high-friction fingertips. As noted in [6] and elsewhere, soft high-friction contact reduces surface forces, increases stability, and increases the effective coupling between finger and object. However, our approach does not necessarily benefit from increased stability and coupling between finger and object.

Our approach has its roots in the parts orienting research community [11, 15, 10, 5, 8], where the ideas of using knowledge of stable poses and iterative randomized blind manipulation strategies are well known. As far as we know the present paper is the first to apply these ideas to problems including capture and singulation in cluttered environments.

One central question is the existence and characterization of stable poses without friction, and without force closure, sometimes referred to as higher-order form closure, or second-order stability. See [9] for examples involving planar polygons, and [21, 22] for closely related research.

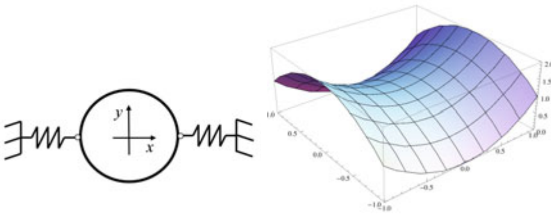


Fig. 4 Potential function for a Hanafusa and Asada-style hand with two fingers grasping a disk. With offsets that would bring the fingers to rest at the center, the potential function is a saddle. Adding two more fingers yields a constant potential function.

2 The Set of Stable Poses

This section addresses the set of stable poses of an object grasped by the simple gripper described earlier. The issue is not just whether a certain pose is stable, but whether the subspace of stable poses has a structure that supports pose estimation. The ideal would be a single stable pose with a large basin of attraction, requiring zero sensory information to determine the pose. The worst case is symmetric objects where the stable pose space includes submanifolds of poses, all of them mapping to the same finger encoder values. Friction also poses a challenge, yielding connected regions of stable poses, rather than isolated points, but without the stationary sensor mapping produced by symmetric objects.

Nonetheless, existence of stable grasps is an issue, and has some implications for the design of our hand. For a blind grasp to accomodate a range of shapes, fingers require a large range of motion. We consider two alternatives. Option one, *large offsets*, obtains the entire range of motion from the individual finger springs. Bounds on contact force require that the spring be soft, and that the spring offsets be sufficient to move the finger well past the maximum desired motion. Option two, *small offsets*, would use stiff finger springs with very small deflections. The large of range of motion could be obtained by driving the actuator until a threshold force is exceeded, or by a soft spring in series with the actuator. The following sections address both options for a disk in two dimensions, and for a sphere and a polyhedron in three dimensions.

2.1 Unstable Grasp of a Disk in 2D

While it may seem obvious that four fingers spaced equally about a disk would give a stable grasp, the reality is not so simple. Large offsets may yield an unstable grasp [2]. First consider a two-finger grasp (Fig. 4). While the finger compliances stabilize motions aligned with the fingers, the curvature of the disk yields a negative stiffness for transverse motions. The potential function is a saddle and the grasp is unstable. If we add two more fingers to obtain a four-finger grasp, the negative stiffness for one finger pair can nullify the positive stiffness of the other pair, yielding a metastable grasp.

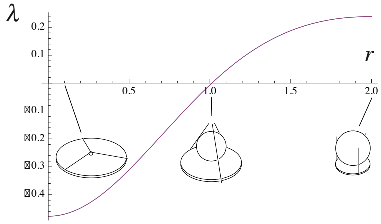


Fig. 5 Local stability of sphere grasped with large finger spring offsets. The stiffness matrix eigenvalues are negative for small spheres and positive for large spheres.

2.2 Stable Grasp of a Sphere

This section explores the stability of a sphere being grasped by a palm and three line fingers. The main focus is to examine variations of scale. We also reexamine the effect of large offsets versus small.

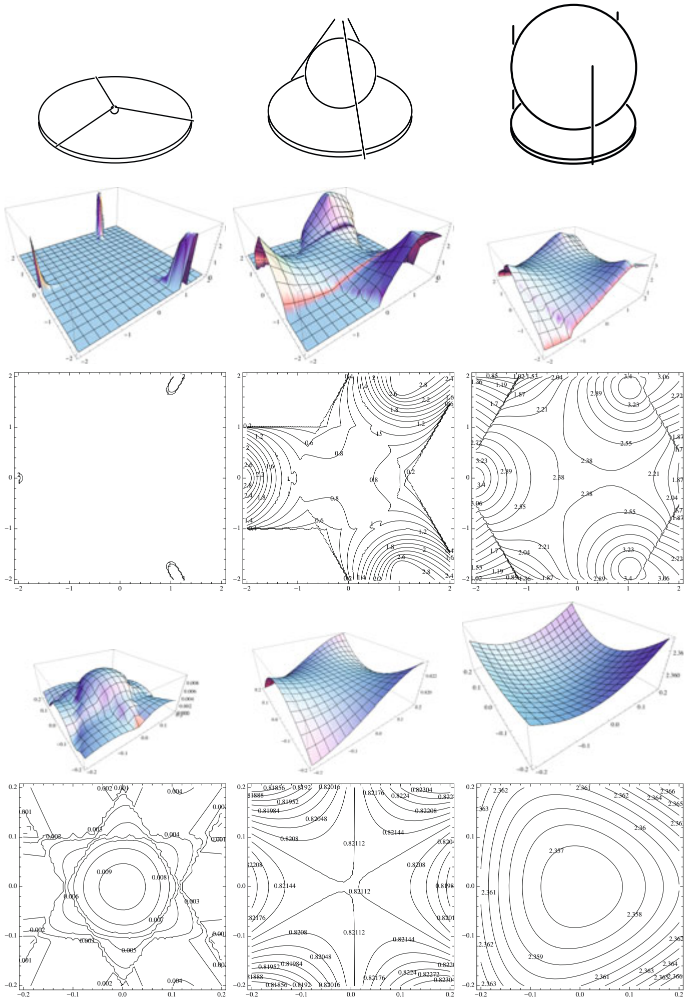
The hand is the simple gripper of Fig. 2, except the fingers are lines rather than cylinders. To find the total potential energy we start with the potential of a single finger. For a sphere of radius r located at (x,y,z), and a finger nominally aligned with the x-axis, the finger angle is:

$$\theta = \pi/2 - \tan^{-1}(z/x) - \tan^{-1}(\sqrt{r^2 - y^2}/\sqrt{x^2 + y^2 + z^2 - r^2})$$

The total potential is $U = 1/2k\Sigma(\theta_i - \theta_{i0})^2$, assuming stiffnesses k and offsets θ_{i0}. We assume the sphere is in palmar contact, i.e. $z = r$, and write the potential as a function of x and y. The gradient of this potential is the force in the x-y plane, and the Hessian of the potential is the stiffness matrix. With the sphere placed at the origin the gradient is zero as expected. We can determine stability by examining the eigenvalues of the Hessian. Fig. 5 shows the eigenvalues when the springs have large offsets—large enough to close the fingers all the way to the palm. The two eigenvalues are equal, and are negative for small spheres, and positive for large spheres. The grasp is unstable for small spheres, and stable for large. Note the contrast with the planar analysis of the previous section. For the largest sphere considered, the fingers are exactly vertical, mimicking the planar three finger grasp. However, the planar grasp is unstable, where the three-dimensional grasp is stable. Evidently when the sphere moves toward the gap between two fingers, the third finger's force inclines downward, so that the planar component drops off rapidly enough to eliminate the instability.

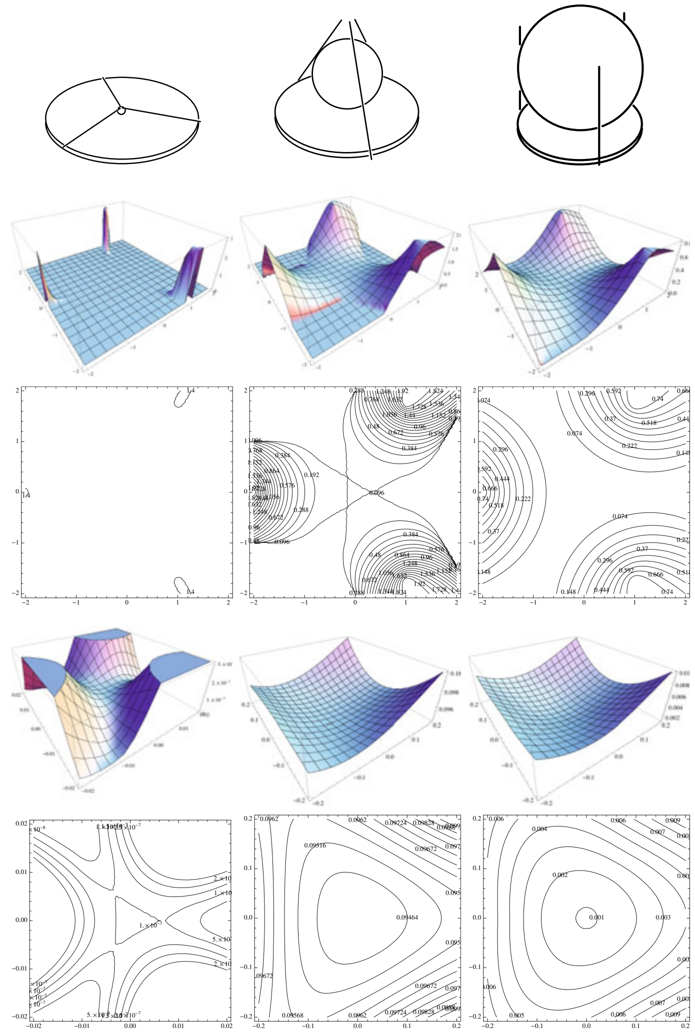
The eigenvalues of the Hessian give us the local stability, but tell us nothing about the basin of attraction. In Figs. 6 and 7 we plot the potential surface for three different sphere sizes and both large and small spring offsets. These plots tell a more complicated story. The global structure more closely resembles a three-lobed saddle, sometimes called a monkey saddle. Because a monkey saddle is fundamentally a third-order surface, the second-order analysis of the Hessian determines local stability. But the size of the basin of attraction, and the robustness of the stability, are determined by the size of the sphere and the spring offsets. As expected small offsets improve the basin of attraction for the large sphere, and can stabilize the grasp even for the small sphere. The medium size sphere corresponds to the crossover point of Fig. 5, and is now shown to be unstable, although it is stabilized with small offsets.

The obvious and unsurprising conclusion is that large offsets tend to yield unstable grasps. This is consistent with the general trend of grasp stability analysis over the years, which has focused on stiff grasps and small offsets. However, our goals are different. We do not aim to stabilize as many poses as possible. We would prefer a few stable grasps with large basins of attraction.



Radius 0.1 Radius 1 Radius 2

Fig. 6 Potential fields for large offsets. The offsets would close the fingers to the palm. The lower half of the figure is zoomed in to show detail not revealed in the upper half.



Radius 0.1 Radius 1 Radius 2

Fig. 7 Potential fields for small offsets. The lower half of the figure is zoomed. The spring deflections are $\pi/10$ for the large and medium spheres, $\pi/10,000$ for the small sphere.

One way to address this is to have very stiff finger springs, and to place a force threshold on the actuator, so that the actuator stalls with small offsets. We will explore a similar idea, which is to have four springs: three stiff finger springs and one soft actuator spring. A very soft actuator spring with a very large offset provides a reasonable approximation of a stalling actuator.

Introducing a fourth spring couples the fingers, and complicates the derivation of a potential function. In short, we now have a linear complementarity problem.

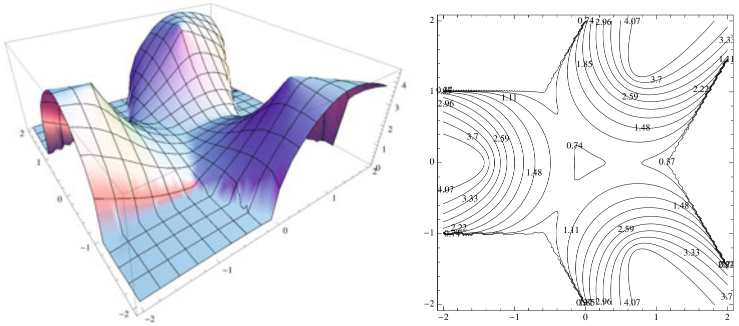


Fig. 8 Potential surface and contours for sphere of radius one grasped by gripper with four springs: each finger springs with stiffness 10 and a motor spring of stiffness 1.

The naive algorithm is exponential in the number of contacts, but 2^3 is only 8, so we embrace naiveté: For every subset of the three fingers, we assume that subset is in contact, and the rest follows easily. Fig. 8 plots the potential surface and several contours, for the medium sphere, where the finger springs are ten times stiffer than the motor spring.

2.3 Stable Grasp of a 3-4-5 Prism

This section explores the stability of a polyhedron grasped by the three finger hand, with three springs, with large offsets. We exhibit the level surfaces of the potential field in a three-dimensional slice of the six-dimensional configuration space. As expected with an irregular object, the stable poses are isolated points (Fig. 9).

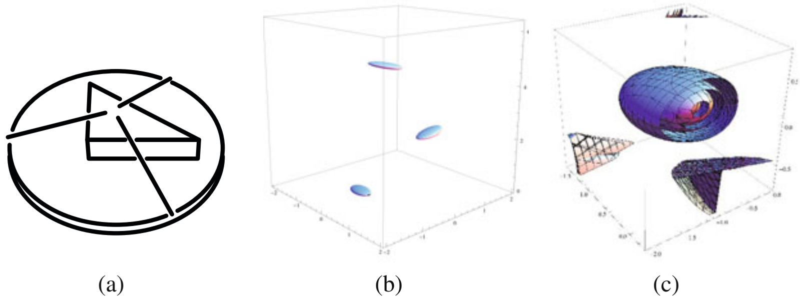

(a) (b) (c)

Fig. 9 Potential function for the simple three finger hand grasping a prism formed by extruding a triangle with side lengths of 3, 4, and 5. Height of the prism is one. The plot shows three closed contours enclosing isolated potential wells corresponding to three stable grasps obtained with the prism flat against the palm. There will be at least three more isolated stable grasps when the triangle is flipped over, and there may be other stable grasps we have not identified. The first plot shows all three potential wells represented by a single contour. The second plot is a closer view of one potential well, with some additional contours plotted.

Not all objects will exhibit isolated equilibria. That is obviously the case with symmetries such as the spheres examined in the previous section. Even for polyhedra, it is possible to define a shape and a positive-length path for that shape, while maintaining contact with a palm and three line fingers. The shape is a variation on the example of [9], a planar polygon with three concurrent edges, extruded to a polyhedral prism.

3 Simulation and Experiments with the Barrett Hand

This section explores the application of principles detailed in the previous sections to the commercially-available Barrett hand. In contrast to the compliant 3DOF simple hand, the Barrett hand (Fig. 10) is stiff, it is not frictionless and it has 7DOFs, 4 of which are active. We are not using a system carefully engineered to match our assumptions. Rather, we are asking whether our approach is applicable when the assumptions are violated. In particular, we explore the construction of grasp tables, and we examine a part of the space of stable poses. Our results are twofold: (1) departures from our assumptions do present challenges; and (2) even so, the structure of the set of stable poses will support partial estimation of pose.

3.1 Grasp Tables

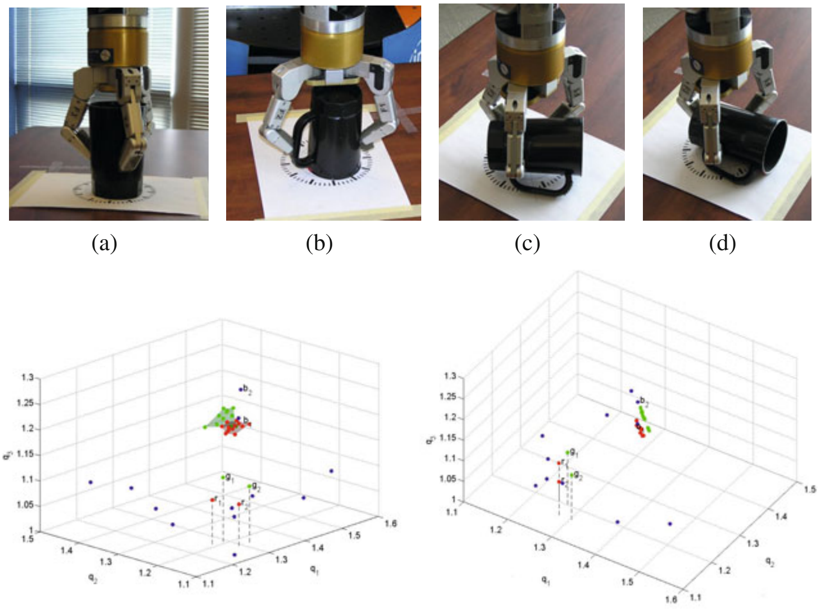
A grasp table is a set of samples of the mapping from grasp inputs to grasp outcomes. Depending on the problem, the inputs may include the relative pose of the object and the hand, the shape of the hand, the material properties of the surfaces, and the expected clutter. The output may include the pose of the object and the hand at the end of the grasping operation, a score of the goodness of the grasp, the location of contact points on the grasp, readings of sensors instrumented on the hand, the object or the environment, or even just a Boolean value denoting grasp success or failure.

Constructing a realistic grasp table depends crucially on the engine that takes the input, simulates contact physics, and produces the output. In this paper, we use real world experiments to produce a small grasp table (Section 3.2) and then address construction of grasp tables using kinematic simulations (Section 3.3).

In the past, grasp tables have served the purpose of precomputed and cached templates or behaviors that the robot searches through online to find a good match for a given scenario. The tables have usually been relatively small, with $20-50$ entries. However, localization introduces an additional mapping: from the space of outputs to the sensor space. Successful localization requires the inversion of the above mapping, *i.e.*, to estimate the relative pose of the object in the hand at the end of the grasping operation, from just the sensor readings. Section 3.3 describes the complexities introduced by this additional mapping, and our experimental results.

3.2 Experimentally Developed Grasp Table

Fig. 10 shows an experimentally produced grasp table. The initial mug position was fixed, while several stable orientations were sampled. As the hand closed, the object



(a) (b) (c) (d)

(e) (f)

Fig. 10 An experimentally derived grasp table

was constrained only by the table, simulating the effect of the palm. Three clusters appear in the sensor space (Fig. 10(e), Fig. 10(f)): red points for the upright mug, green points for the inverted mug, and blue points for the mug on its side. The upright and inverted poses are separate because the mug has a slight taper. Besides the clusters, we observe some outliers, for example point G1 where one finger landed on the handle instead of the body. (Fig. 10(b)), and point B1 (Fig. 10(d)) where capture failed.

3.3 Grasp Tables via Kinematic Simulation

To further our goal of producing a dense grasp table comprising many thousands of samples, we augment real-world samples with those derived from simulation.

In our experiments, we observed significant movement of the mug before it settled into a grasp. Simulating such contact dynamics is computationally expensive and inaccurate. Instead, we chose a two-step approach of building a kinematic grasp table (Fig. 11(a)) followed by a refinement step motivated by Section 2 in which the kinematic samples (Fig. 11(b)) are perturbed into the nearest statically stable minimum energy configuration using simulated annealing(Fig. 11(c)). Our preliminary results are encouraging: refinement eliminates unlikely grasps like Fig. 12(right-mid) and Fig. 12(right-bottom) while preserving likely grasps like Fig. 12(left-top) which are experimentally verifiable (Fig. 12(left-bottom)).

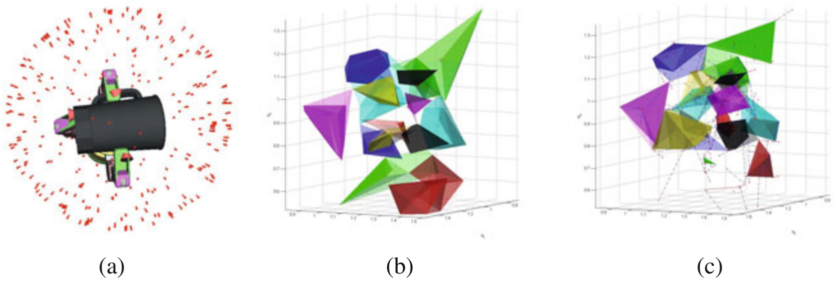


Fig. 11 Grasp table refinement: (a) Input samples, (b) kinematic grasp table clustered in sensor space, (c) clusters shrink and collapse after minimum energy refinement

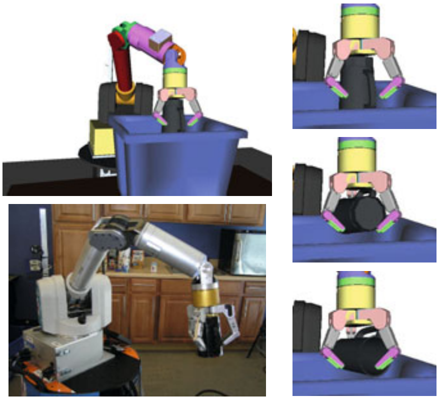


Fig. 12 Right: Pose ambiguity in the kinematic grasp table, Left: Predicted pose from grasp refinement and real-world experiment

Our ultimate goal is to infer object pose from sensor readings. For the Barrett hand with moderate friction, we observed significant pose ambiguity with the kinematic grasp table (Fig. 12(right-all)). With refinement, some of the unstable grasps were filtered out, resulting in fairly accurate pose estimation (Fig. 12(left)). However, the ambiguity could not be eliminated in many other configurations. This suggests two potential strategies: (1) to avoid ambiguous configurations altogether, and (2) to disambiguate by further sensing or sensorless action. Our future work will explore both strategies.

Acknowledgements. This work was supported by NSF, DARPA, Junta de Comunidades de Castilla-La Mancha, and Intel. We wish to thank Tom Fuhlbrigge, David Bourne, Mike Erdmann, Alberto Rodriguez, Garth Zeglin, Amir Degani, Koen Ruskamp, Tomas Lozano-Perez. This work is sponsored in part by the Defense Advanced Research Projects Agency. This work does not necessarily reflect the position or the policy of the Government. No official endorsement should be inferred.

References

1. Ambler, A.P., Barrow, H.G., Brown, C.M., Burstall, R.M., Popplestone, R.J.: A versatile computer-controlled assembly system. In: Third Int. Joint Conf. on Artificial Intelligence, pp. 298–307 (1973)
2. Baker, B., Fortune, S., Grosse, E.: Stable prehension with a multi-fingered hand. IEEE Int. Conf. on Robotics and Automation 2, 570–575 (1985)
3. Bicchi, A.: Hands for dexterous manipulation and robust grasping: A difficult road toward simplicity. IEEE Tran. on Robotics and Automation 16(6), 652–662 (2000)
4. Butterfass, J., Grebenstein, M., Liu, H., Hirzinger, G.: DLR-Hand II: Next generation of a dextrous robot hand. In: IEEE Int. Conf. on Robotics and Automation, vol. 1, pp. 109–114 (2001)
5. Canny, J., Goldberg, K.: "RISC" industrial robotics: Recent results and open problems. In: IEEE Int. Conf. on Robotics and Automation, pp. 1951–1958 (1994)
6. Cutkosky, M., Wright, P.: Friction, stability and the design of robotic fingers. The Int. Journal of Robotics Research 5(4), 20–37 (1986)
7. Dollar, A., Howe, R.: Towards grasping in unstructured environments: Grasper compliance and configuration optimization. RSJ Advanced Robotics 19(5), 523–543 (2005)
8. Erdmann, M., Mason, M.: An exploration of sensorless manipulation. IEEE Journal of Robotics and Automation 4(4), 369–379 (1988)
9. Farahat, A., Stiller, P., Trinkle, J.: On the geometry of contact formation cells for systems of polygons. IEEE Tran. on Robotics and Automation 11(4), 522–536 (1995)
10. Goldberg, K.: Orienting polygonal parts without sensors. Algorithmica 10(2), 201–225 (1993)
11. Grossman, D., Blasgen, M.: Orienting mechanical parts by computer-controlled manipulator. IEEE Tran. on Systems, Man, and Cybernetics 5(5), 561–565 (1975)
12. Hanafusa, H., Asada, H.: Stable prehension by a robot hand with elastic fingers. In: Proc. Seventh Int. Symp. Industrial Robots, pp. 361–368 (1977)
13. Hollerbach, J.: Grasping in human and robot hands. In: Vision and Action: The Control of Grasping, pp. 243–274 (1990)
14. Jacobsen, S., Wood, J., Knutti, D., Biggers, K.: The Utah/MIT dextrous hand: Work in progress. The Int. Journal of Robotics Research 3(4), 21–50 (1984)
15. Kriegman, D.: Let them fall where they may: Capture regions of curved objects and polyhedra. The Int. Journal of Robotics Research 16(4), 448–472 (1997)
16. Lin, Q., Burdick, J., Rimon, E.: Computation and analysis of natural compliance in fixturing and grasping arrangements. IEEE Tran. on Robotics 20(4), 651–667 (2004)
17. Lozano-Pérez, T., Jones, J., O'Donnell, P., Mazer, E.: Handey: A robot task planner. MIT Press, Cambridge (1992)
18. Mason, M., Salisbury Jr., J.: Robot Hands and the Mechanics of Manipulation. The MIT Press, Cambridge (1985)
19. Nguyen, H., Kemp, C.C.: Bio-inspired assistive robotics: Service dogs as a model for human-robot interaction and mobile manipulation. In: The Second IEEE/RAS-EMBS Int. Conf. on Biomedical Robotics and Biomechatronics, pp. 542–549 (2008)
20. Nguyen, V.: Constructing stable grasps. The Int. Journal of Robotics Research 8(1), 26–37 (1989)

21. Rimon, E., Burdick, J.: Mobility of bodies in contact. I. A 2nd-order mobility index formultiple-finger grasps. IEEE Tran. on Robotics and Automation 14(5), 696–708 (1998)
22. Rimon, E., Mason, R., Burdick, J., Or, Y.: A general stance stability test based on stratified morse theory with application to quasi-static locomotion planning. IEEE Tran. on Robotics 24(3), 626–641 (2008)
23. Salisbury, J., Craig, J.: Articulated hands: Force control and kinematic issues. The Int. Journal of Robotics Research 1(1), 4–17 (1982)
24. Saxena, A., Driemeyer, J., Ng, A.: Robotic grasping of novel objects using vision. The Int. Journal of Robotics Research 27(2), 157–173 (2008)

From Sensorimotor Primitives to Manipulation and Imitation Strategies in Humanoid Robots

Tamim Asfour, Martin Do, Kai Welke, Alexander Bierbaum, Pedram Azad, Nikolaus Vahrenkamp, Stefan Gärtner, Ales Ude, and Rüdiger Dillmann

Abstract. Regardless of the application area, one of the common problems tackled in humanoid robotics is the investigation towards understanding of human-like information processing and the underlying mechanisms of the human brain in dealing with the real world. Therefore, we are building humanoid robots with complex and rich sensorimotor capabilities as the most suitable experimental platform for studying cognitive information processing. The target system is supposed to interact and function together with humans. It is meant to be able to cooperate and to enter a dialogue with them. Therefore it needs to understand both what it perceives and what it does. To achieve this goal we are following a holistic approach and investigating how different partial results, achieved in sub-disciplines related to the development of humanoids, fit together to achieve complete processing models and integrative system architectures, and how to evaluate results at system level rather than focusing on the performance of component algorithms. In this paper we present our recent and current progress in this direction. For doing so, we present our work on the development of humanoid platforms, the programming of grasping and manipulation tasks, the multi-sensory object exploration as well as the learning of motor knowledge from human observation.

1 Introduction

Human-centered robotics is an emerging and challenging research field which has received significant attention during the past few years. The paradigmatic shift in

T. Asfour · M. Do · K. Welke · A. Bierbaum · P. Azad · N. Vahrenkamp · S. Gärtner · R. Dillmann
Karlsruhe Institute of Technology (KIT), Institute for Anthropomatics,
Humanoids and Intteligence Laboratory, Karlsruhe, Germany
e-mail: tamim.asfour@kit.edu

A. Ude
Jozef Stefan Institute, Department of Automatics, Biocybernetics, and Robotics,
Ljubljana, Slovenia
e-mail: ales.ude@ijs.si

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 363–378.
springerlink.com

the robotics at the end of the 1980s brought robots from factories to other working and everyday human environments and raised the important problem of engineering of cognitive robot systems, which have to safely coexist with humans, interactively communicate with humans and usefully manipulate objects in built-for-human environments. The new generations of robots appeared with this paradigmatic shift are meant to accomplish a wide range of new tasks and to play a wide range of new roles: from coworkers to assistants in private households and assistants for the disabled and the elderly.

Humanoid robots are associated with the idea of robots whose physical appearance is similar to that of the human body. Beyond a physical resemblance, humanoid robots are meant to resemble humans in their ways of acting in the world, of reasoning and communicating about the world. Currently, an encouraging spectrum of isolated elements in this area is realizable with focus on performance in well-defined, narrow domains. However, successful attempts in building humanoid robots for human-centered environments are still limited to systems designed for ”sunshine” environments, able to perform simple tasks of limited scope. Further, the transferability of the developed skills and abilities to varying contexts and tasks without costly redesign of specific solutions is still impossible.

The paper is organized as follows. In Section 2, we give a brief overview about the humanoid robots we are developing. Section 3 describes the already implemented capabilities, which are necessary to perform manipulation task in a kitchen environment. Section 4 presents our current work and progress on the development of advanced sensorimotor capabilities such as object exploration and imitation of human movements.

2 The Humanoid Robots ARMAR-IIIa and ARMAR-IIIb

In designing our robots, we desire a humanoid that closely mimics the sensory and sensory-motor capabilities of the human. The robots should be able to deal with a household environment and the wide variety of objects and activities encountered in it. Therefore, the humanoid robots ARMAR-IIIa and ARMAR-IIIb (see Fig. 1 and [3]) have been designed under a comprehensive view so that a wide range of tasks can be performed. The upper body of the robots has been designed to be modular while retaining similar size and proportion as an average person. For the locomotion, a holonomic mobile platform is used. From the kinematics control point of view, both robots consist of seven subsystems: head, left arm, right arm, left hand, right hand, torso, and a mobile platform. For detailed description of the robot, the head, the hand, the artificial skin, and the mechanics, the reader is referred to [3, 4, 28, 19] and [1] respectively. The specification of the robots is given in table 1.

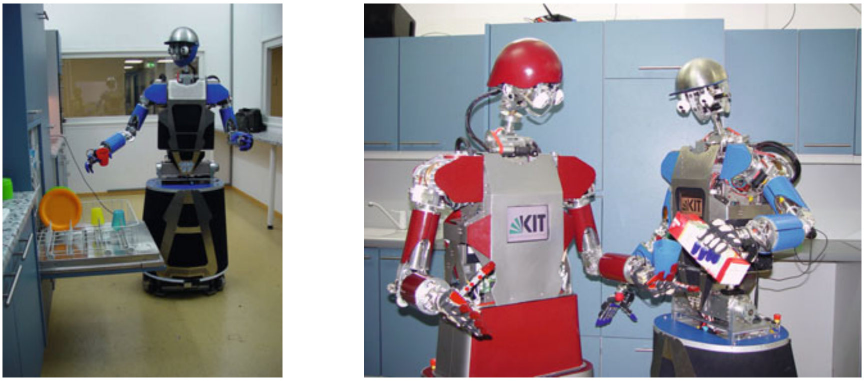


Fig. 1 The humanoid robots ARMAR-IIIa (blue) and ARMAR-IIIb (red) in the kitchen. Each robot has a 7 DoF head, two 7 DoF arms and two 8 DoF five-fingered hands, a 3 DoF torso and a holonomic mobile platform. The head is equipped with two eyes. Each eye is equipped with two cameras, one with a wide-angle lens for peripheral vision and one with a narrow-angle lens for foveal vision.

Table 1 Specification of the humanoid robot ARMAR-III

Weight		135 kg (incl. 60 kg Battery)	
Height		175 cm	
Speed		1 m/sec	
DOF	Eyes	3	Common tilt and independent pan
	Neck	4	Lower Pitch, Roll, Yaw, upper Pitch
	Arms	2 × 7	3 DOF in each shoulder, 2 DOF in each elbow, and 2 in each wrist
	Hands	2 × 8	Five-fingered hands with 2 DOF in each Thumb, 2 DOF in each Index and Middle, and 1 DOF in each Ring and Pinkie.
	Torso	3	Pitch, Roll, Yaw
	Platform	3	3 wheels arranged in angles of 120 degrees
Actuator		DC motors + Harmonic Drives in the arms, neck, eyes, torso and platform. Fluidic actuators in the hand.	
Sensors	Eyes	2 Point Grey (www.ptgrey.com) Dragonfly cameras in each eye, six microphones and a 6D inertial sensor (www.xsens.com).	
	Arms	Motor encoders, axis sensors in each joint, torque sensors in the first five joints, and 6D force-torque sensor (www.ati-ia.com) in the wrist.	
	Platform	Motor encoders and 3 laser-range finders (www.hokuyo-aut.jp).	
Power Supply		Switchable 24V Battery and 220 V external power supply.	
Operating system		Linux with the Real-Time Application Interface RTAI/LXRT-Linux.	
Computers and Communication		Industrial PCs and PC/104 systems connected via Gigabit Ethernet and 10 DSP/FPGA control units (UCoM) which communicate with the control PC via CAN bus.	
User Interface		Graphical user interface (GUI) connected to the robot via wireless LAN and natural speech communication.	

3 Implemented Sensorimotor Capabilities

3.1 Object Recognition and Pose Estimation

One crucial requirement for performing manipulation tasks is the visual perception of the objects of interest. The humanoid robot ARMAR-III is capable of recognizing a set of objects typical for a kitchen environment and of estimating their 6-DoF poses by exploiting stereo vision. Two fully integrated systems have been developed for two different classes of objects: objects offering enough textural information for the application of local point features, and single-colored objects, which are defined by their shape only.

For recognizing textured objects and estimating their pose, local point features are exploited. For application of a visual servoing approach for grasping, as described in Section 3.3, the object pose must be computed within the visual servoing control loop. State-of-the-art features such as SIFT [25] or SURF [11] do not meet the required performance requirements. Therefore, a combination of the Harris corner detector [20] and SIFT descriptor was developed, including an effective extension for achieving scale-invariance, as presented in [8, 5]. With these novel features, feature computation requires approx. 20 ms for images of size 640×480 on conventional hardware, compared to approx. 500–600 ms for conventional SIFT and 150–240 ms for SURF. Furthermore, a stereo-based 6-DoF pose estimation method that builds on top of the 2D localization result was developed. This pose estimation method was compared to conventional pose estimation based on 2D-3D correspondences in [9, 5]. An extensive experimental evaluation on synthetic and real image data proves the superior performance of the proposed approach in terms of robustness and accuracy.

The method developed for single-colored objects, as presented in [7, 5], requires global segmentation of the object, which is accomplished by color segmentation in the case of single-colored objects. Subsequently, the segmented views from the left

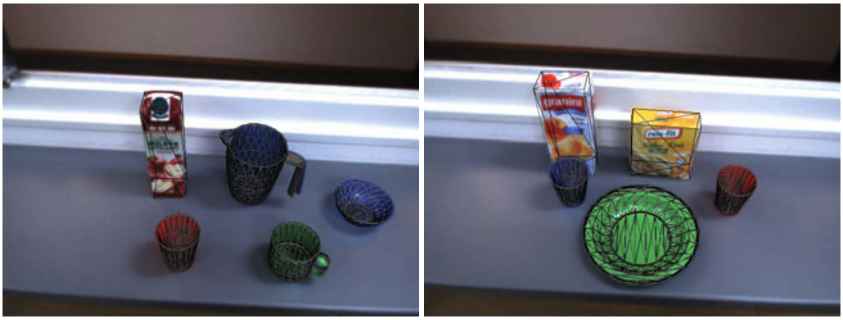


Fig. 2 Exemplary results with the integrated proposed object recognition and pose estimation systems. The computed object pose has been applied to the 3D object model and the wireframe model has been overlaid.

and right camera image are used to compute the identity of the objects and to estimate their 6-DoF pose by combining stereo triangulation with appearance-based view matching. For this purpose, training views are generated in simulation in order to produce a view space covering all relevant views of the object. An initial pose estimate is computed by splitting up position and orientation information first: The initial position estimate is computed by stereo triangulation, and the initial orientation estimate is given by the orientation that was stored along with the matched view. Since by splitting up position and orientation estimation, the pose estimate becomes error-prone, a pose correction algorithm was developed, which utilizes a 3D model of the object and online 3D rendering.

Both systems run at frame rate (30 Hz) for recognizing a single object and estimating its pose, as applied throughout grasping using visual servoing. The developed methods and algorithms are presented in detail [5], including an extensive evaluation and discussion. Fig. 2 shows exemplary results computed with the developed systems.

3.2 Single and Dual Arm Motion Planning

For collision-free motions, sampling-based algorithms are used to plan single and dual arm grasping and manipulation motions. Such probabilistic algorithms, e.g. the Rapidly-exploring Random Trees (RRT), can be used to search for collision-free motions in large configuration spaces [23, 24, 34]. The realized planning algorithms are able to deal with the different kinematic chains of the robot (e.g. platform, torso, arm or hand) and allow for the generation of collision-free motions for varying planning problems in the context of grasping and manipulation.

To grasp a known object, for which a set of grasping positions is defined (see Fig. 3(a)), several problems have to be addressed: grasp selection, solving the inverse kinematics (IK) and redundancy resolution. The IK-RRT algorithm we presented in [31] for planning single and dual arm grasping motions, integrates the problems of selecting a reachable grasp out of a set of grasping poses, calculating a

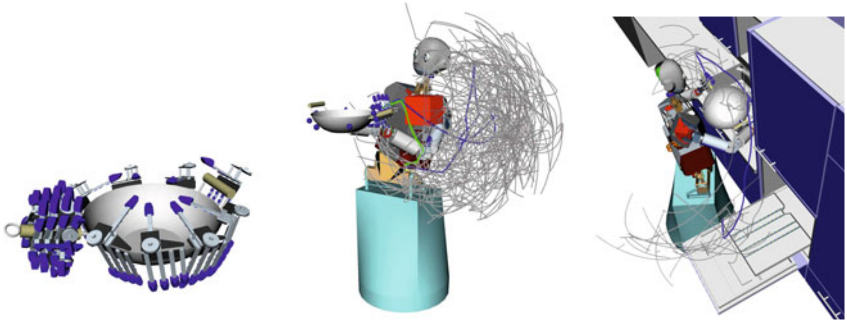


Fig. 3 (a) A wok with 15 associated grasping poses for the left hand of ARMAR-III. (b),(c) Results of the IK-RRT planning algorithm for single-arm and re-grasping tasks.

feasible solution to the IK-problem and generating a collision-free grasping motion to one integrated planning scheme. The planner utilizes a probabilistic IK-solver, which is queried during the RRT-based planning in order to generate potential target configurations (see Fig. 3(b)). A dual arm IK-RRT planner, an extension of the IK-RRT concept, generates collision-free re-grasping motions for two-handed manipulations. For this purpose, the basic concept of the IK-RRT algorithm is extended to account for feasible re-grasping position without the need to specify hand-over position in advance (see Fig. 3(c)). In [30], a multi end-effector RRT-planner is presented, which in addition takes into account the selection of the end-effector that should be used to grasp an object. For this purpose, a parallelization of the search process is realized to account for multiple end-effectors, where each of them is associated with multiple possible grasping poses for the target object.

3.3 Visual Servoing for Grasping

For the realization of grasping and manipulation tasks, the underlying robot system has either to be exactly calibrated or supervision techniques have to be used to achieve reliable positioning of the end-effectors. Even in exact calibrated systems, a supervision is desirable in order to perceive changes in the state of the environment and to react to errors or disturbances. Therefore, visually controlled execution, also referred to in the literature as *visual servoing* is implemented to allow the successful execution of grasping. More specifically *position-based visual Servoing* approaches ([35, 39, 17]) are used. The implemented visual servoing controller on ARMAR-III utilizes stereo-vision for continuous hand and object tracking. The position of the hand is derived from the visually determined 3D position of an artificial marker attached to the wrist, while the orientation is determined using the forward kinematics of the arm. The Cartesian distance between the hand pose and the target object pose is used to calculate velocity commands using the Pseudoinverse oh the Jacobian. For safety reasons, force feedback from the 6D force/torque sensor is used to counteract impact forces [33]. A *bimanual Visual Servoing* controller is implemented for dual arm grasping tasks, where both hands together with one or two objects are tracked in order to retrieve the current relations in the workspace [32]. Results of grasping and manipulation tasks based on the proposed visual servoing techniques on ARMAR-III are shown in Fig. 4.

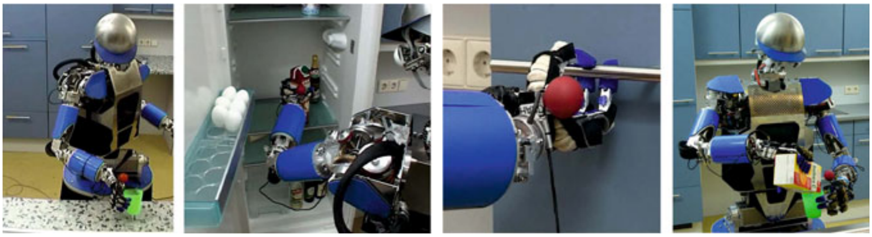


Fig. 4 Visual Servoing enables ARMAR-III to grasp and manipulate known objects with one or two hands.

4 Towards Cognitive Capabilities

The robots capabilities described above are integrated in a holistic way to allow the implementation of complex grasping and manipulations tasks in a kitchen environments, in which the robot was able to grasp daily objects placed on a table, to open the fridge and grasp objects inside it as well as to open, load and close the dishwasher. However, we equipped the robot with basic knowledge about the objects, which has to be manipulated and with sensorimotor knowledge about the actions that have to be performed. Humanoid robots for human daily lives are supposed to interact and function together with humans. They are meant to be able to cooperate and to enter a dialogue communicating with them. Therefore, research should combine the study of perceptual representations that facilitate motor control, motor representations that support perception, and learning based on actively exploring the environment and interacting with people that provides the constraints between perception and action. This will then allow, e.g., to learn the actions that can be carried out on and with objects. This leads to what we call Object-Action-Complexes (OAC), a paradigm introduced within the European PACO-PLUS project (see [26, 22, 18, 40]), to emphasize the notion that objects and actions are inseparably intertwined and that categories are therefore determined (and also limited) by the action a cognitive agent can perform and by the attributes of the world it can perceive.

In the following we describe our research results on acquiring multi-modal object representations, which develops through exploration making use of the interplay between different sensorial modalities, such as of visual and haptic information. In addition, we describe how new actions can be learned from human demonstration, represented adapted and applied for executing manipulation tasks, thus replacing engineering approaches, that are currently used to generate actions.

5 Visuo-Haptic Object Exploration

For humanoid robots operating in human centered environments the ability of adaptation is a key issue. Autonomous adaptation to new tasks, domains, and situations is a necessary prerequisite for performing purposeful assistance functions in such environments. The combination of sophisticated sensor systems in humanoid platforms together with the ability to interact with the environment allows autonomous exploration in order to gain and consequently adapt knowledge about the surrounding world in a goal-oriented manner.

5.1 Visually-Guided Haptic Object Exploration

Haptic exploration of unknown objects is of great importance for acquiring multimodal object representations, which enable a humanoid robot to autonomously execute grasping and manipulation tasks. To acquire such representations, an exploration strategy is needed, which guides the robot hand along the surface of unknown objects and build a 3D object representation based on acquired tactile point clouds. We propose an exploration strategy that makes use of the dynamic potential field approach. To demonstrate the capabilities of this strategy, experiments are performed in a physics simulation using models of the five-finger robot hand.

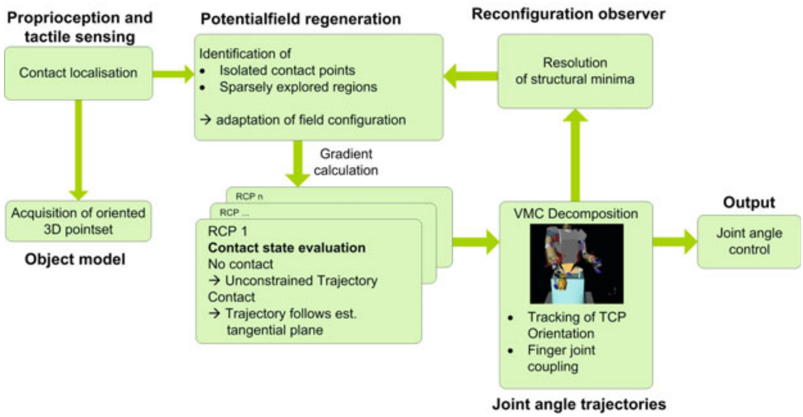

Proprioception and tactile sensing
Contact localisation
Acquisition of oriented 3D pointset
Object model
Potentialfield regeneration
Identification of
• Isolated contact points
• Sparsely explored regions
→ adaptation of field configuration
Gradient calculation
RCP n
RCP ...
RCP 1
Contact state evaluation
No contact
→ Unconstrained Trajectory
Contact
→ Trajectory follows est. tangential plane
Reconfiguration observer
Resolution of structural minima
VMC Decomposition
• Tracking of TCP Orientation
• Finger joint coupling
Joint angle trajectories
Output
Joint angle control

Fig. 5 Haptic exploration scheme based on dynamic potential fields.

Fig. 5 shows the concept of the exploration process, which is initiated through a rough estimate about the objects position, orientation and dimension. In real experiments, this information can be provided by the robot's stereo-vision system, and has to be provided manually in the simulation framework. The trajectories of the finger tips which are equipped with tactile sensors are calculated as described in [12]. The resulting object representation is an oriented 3D point set constructed from the contact data of the tactile sensors on the finger tips. Such a representation has the advantage that it may be enhanced directly with data from a stereo camera system and that it may be processed using well known algorithms, e.g. for the purpose of object recognition. A snapshot of an exploration example and a resulting 3D point set are shown in Fig. 6.

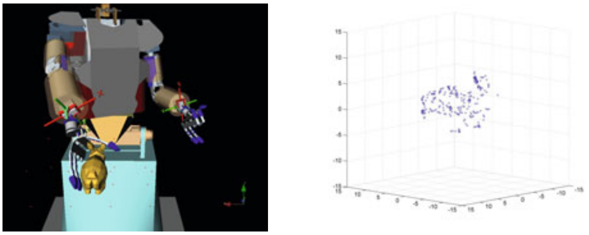


Fig. 6 Haptic exploration of a rigid bunny and the resulting 3D point set.

Initially, a potential field consisting of only attractive forces is defined in a uniform grid, which covers the exploration space. During the exploration the finger tips trajectories are continuously calculated by the gradient of the potential field, while contact points and normals are acquired and stored as oriented 3D point set. The potential field is updated by deleting attractive forces when contact is detected and adding repelling forces at the contact location. Local minima which arise during the exploration are resolved by a reconfiguration observer, which detects when the TCP velocity and the mean velocity of the finger tips fall below predefined minimum velocity values. In such a situation, the a new hand configuration is used to continue the exploration. In a subsequent step, the oriented 3D point set acquired during the exploration are processed to identify geometric object properties and potential grasp candidates (see [13]).

5.2 Active Visual Object Exploration and Search

In the field of visual perception, an important aspect of world knowledge generation consists of the acquisition of internal representations of objects. While many available recognition systems rely on representations, which have been acquired offline, exploration provides the opportunity to autonomously acquire internal representations. The benefits of such capabilities are two-fold: First, autonomous acquisition of internal representations of objects simplifies the generation of new object knowledge. Second, together with the ability of recognizing known and identifying unknown object instances in the environment, autonomous acquisition allows to adapt to changing environments as required in a human-centered world. As a consequence, we follow an integrated perspective on visual object modelling and recognition. The goal is to equip the humanoid robot ARMAR-III with the ability to acquire visual object representations autonomously which can then be used for object search and recognition in future tasks.

Fig. 7 illustrates the acquisition of visual object representations on the humanoid platform ARMAR-III. During the acquisition process, unknown objects are held in the five-fingered hand and different view angles of the object are generated in order to cover as much visual information as possible. The object views are captured with the robot's cameras and contain major parts of the hand and the scene. In order to process the object views and to derive viable representations, object-background and object-hand segmentation are performed based on a set of cues which are generated using visual as well as proprioceptive information (see [37]).

The segmented views are combined to form a multi-view object representation. The focus is put on the acquisition of the appearance based part of object representations, which can be used for a visual search task. The acquisition of geometric representations as required for grasping only based on visual input is difficult. Rather, the generation of the geometric part is mainly achieved using approaches of haptic exploration (see Sec. 5.1). The generated views are combined in an aspect graph representation, where each node corresponds to one view of the object and edges express the neighbor relations. The views are processed using different

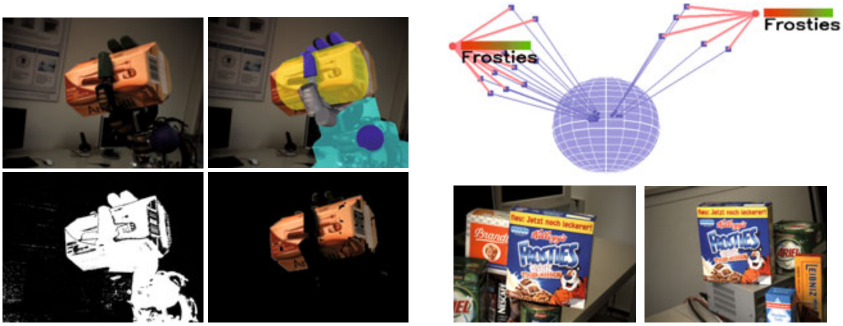


Fig. 7 Left: Exploration of an unknown object held in the hand of ARMAR-IIIb. Hand-object and background-object segmentation is performed in order to generate an appearance based representation. Right: Result of a visual search task. The scene memory contains two instances of the searched object. The active camera system focuses on both instances alternately.

feature extraction methods. A compact representation is derived by clustering the views according to their similarity (see [38]). The resulting aspect graph and its associated prototypical features are stored in the knowledge base.

With the generated representations, visual object search is performed on the active head of the humanoid robot ARMAR-III (see [36]). During the search process, hypotheses of object locations are generated based on the views of the perspective cameras of the head. Using the generated hypotheses, attention mechanisms guide the gaze of the foveal cameras in order to perform a more detailed inspection of the scene. The system successively fills an ego-centric scene memory with the results of the hypotheses generation and verification processes. Thus, consistent and persistent information is accumulated in the scene memory with respect to the searched object as depicted in Fig. 7. The content of the scene memory is available for higher level tasks.

6 Learning from Human Observation

6.1 *Markerless Human Motion Capture*

Markerless human motion capture is a prerequisite for learning from human observation in a natural way. The sensor system to be used for this perceptual capability is the wide angle camera system built-in in the head of ARMAR-III. The two main problems are to capture real 3D motion despite the small baseline of 90 mm as well as to meet the real-time requirements for online imitation learning. As probabilistic framework, a particle filter is used. In our earlier work [10], we have introduced the integration of a 3D head/hand tracker as an extra cue into the particle filter. This additional cue allows to reduce the effective search space by dragging the probability distribution into a relevant subspace of the whole search space. In our more recent work [6, 5] we have focused on improving the accuracy and smoothness of the acquired trajectories by analyzing and solving the typical problems with markerless human motion capture using particle filters. In order to achieve this goal, a priorioritzed fusion method, adaptive shoulder positions, and adaptive noise in particle sampling have been introduced. Furthermore, the redundant inverse kinematics of the arm, given a hand and a shoulder position, were integrated into particle sampling, in order increase robustness to unexpected movements, to allow immediate recovery from mistrackings, and to support application of the system at lower frame rates. After sampling a subset of the particles according to the redundant inverse kinematics, several runs of an Annealed Particle Filter [15] are performed to refine the probability distribution.

6.2 Master Motor Map

To allow the reproduction of human motion acquired from different human motion capture systems on different robot embodiments as well as to allow the development and evaluation of action recognition systems independent from the data source. To overcome the data compatibility problems arising form different data formats of motion, we have specified the so-called Master Motor Map (MMM) as an interface for exchanging motor knowledge between different embodiment, and as a framework for decoupling data from various sources accounting for perception, visualization, reproduction, analysis, and recognition of human motion.

The MMM is defined as a three-dimensional reference kinematic model of the human body enriched with body segment properties. The strategy with respect to the kinematic model is to define the maximum number of degrees of freedom (DoF) that might be used by any applied module. The numbers of DoF and the Euler angle

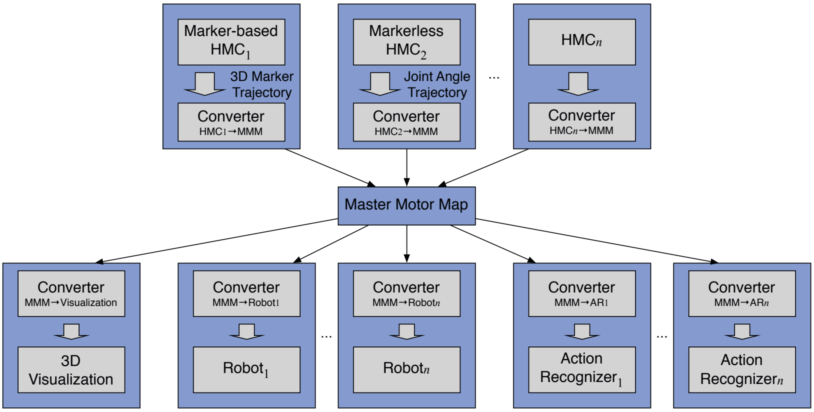


Fig. 8 Illustration of the proposed framework

Joint	DoF	Euler Angles
Root	6	$R_{X'Z'Y'}(\alpha,\beta,\gamma)$
Pelvis	3	$R_{X'Z'Y'}(\alpha,\beta,\gamma)$
Torso	3	$R_{X'Z'Y'}(\alpha,\beta,\gamma)$
Neck	3	$R_{X'Z'Y'}(\alpha,\beta,\gamma)$
Skullbase	3	$R_{X'Z'Y'}(\alpha,\beta,\gamma)$
Hip R/L	3 + 3	$R_{X'Z'Y'}(\alpha,\beta,\gamma)$
Knee R/L	1 + 1	$R_{X'Z'Y'}(\alpha,0,0)$
Ankle R/L	3 + 3	$R_{X'Z'Y'}(\alpha,\beta,\gamma)$
Toe R/L	1 + 1	$R_{X'Z'Y'}(0,\beta,0)$
Sternoclavicular R/L	3 + 3	$R_{X'Z'Y'}(\alpha,\beta,\gamma)$
Shoulder R/L	3 + 3	$R_{X'Z'Y'}(\alpha,\beta,\gamma)$
Elbow R/L	2 + 2	$R_{X'Z'Y'}(\alpha,\beta,0)$
Wrist R/L	2 + 2	$R_{X'Z'Y'}(\alpha,0,\gamma)$
Total	**54**	

Fig. 9 Number of degree of freedom and euler angle conventions for the joints of MMM

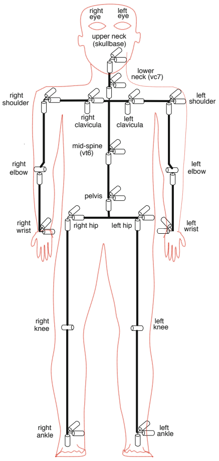


Fig. 10 Illustration of the MMM kinematic model

conventions are listed in table 9. The kinematic model of the MMM is illustrated in Fig. 10. In order to compute gross body dynamics, the kinematic model is enriched with body segment properties, such as mass distribution, segment length, moment of inertia, etc. For this purpose, the linear equations presented in [14] are applied since they represent the most complete and practical series of predictive equations for college-aged Caucasian adults, providing all frontal, sagittal, and horizontal moments of inertia. The body segment properties are adjusted with respect to the kinematics of the MMM and listed in table 2. Using the MMM, different motions have been reproduced on the humanoid robot ARMAR III (see [16]). Furthermore, we are building up a database of movements in the MMM format, consisting of both markerless and marker-based human motion capture data, which will serve as a motion library for the implementation of manipulation tasks on a humanoid robot.

6.3 Goal-Directed Imitation

Generation of complex motor control policies can be achieved through imitation. A human movement is recorded and later reproduced by a robot. One of the challenges need to be mastered for this purpose is the generalization to different contexts since we cannot demonstrate every single movement that the robot is supposed to make. Therefore, motor knowledge extracted from human observation must be represented in a way, which allows the adaptation of learned actions to new situations. We investigated different approaches for the representations of movement primitives based on splines [29], Hidden Markov Models [2] and applied dynamic motor primitives (DMP) as proposed in [21]. A DMP provides a representation of a movement

Table 2 Adjusted body segment properties for the MMM model. Segment masses are relative to body masses; segment lengths are relative to body heights. Both segment center of mass and radii of gyration are relative to the respective segment lengths.

Segment	**Segment Length/ Total Body Height**	**Segment Weight/ Total Body Weight**	**Center of Mass/ Segment Length [x y z]**	**Radius of Gyration/ Segment Length [rxx, ryy, rzz]**
Hip	0.26	0.11	[0 4 0]	[38 36.5 34]
Spine	0.10	0.10	[4 46 0]	[32 26 28.6]
Chest	0.18	0.17	[0 46 0]	[35 28.5 31.3]
Neck	0.05	0.024	[0 20 0]	[31.6 22 31.6]
Head	0.13	0.07	[12 13 0]	[31 26 30]
Shoulder R/L	0.10	0.021	[0 0 -66]	[26 26 12]
Upper Arm R/L	0.16	0.027	[0 -57.3 0]	[26.8 15.7 28.4]
Lower Arm R/L	0.13	0.016	[0 -53.3 0]	[31 14 32]
Hand R/L	0.11	0.006	[0 -36 0]	[23.5 18 29]
Thigh R/L	0.25	0.14	[0 -33 0]	[25 11.4 25]
Shank R/L	0.23	0.04	[0 -44 0]	[25.4 10.5 26.4]
Foot R/L	0.15	0.013	[39 -6 0]	[12 19.5 21]

segment by shaping an attractor landscape described by a second order dynamical system. Using this formulation, discrete and periodic movements can be described. In [27], we applied a motion representation based on dynamic movement primitives (DMPs), which has the advantage that perturbations can be directly handled by the dynamics of the system.

Using the human motion capture system as described in Section 6.1 and the unified representation by the MMM (see Section 6.2), we implemented an on-line imitation learning framework on the robot. Starting from the observation of a human performing a specific task, motion data is obtained, which is segmented regarding the velocity and position changes of the hand or the object motion. After mapping these motion segments onto the robot using the MMM Interface, DMPs are learned and stored in a motion library. Semantic information is added manually to the movement primitives such that they can code object-oriented actions. To reproduce these actions on a robot, a sequence of DMPs is selected and chained, either manually or through a symbolic planner, to achieve the task goal. The imitation of a pick-and-place action consists e.g. of learned DMPS for the different movement segments: approach, place and retreat. The adaption and execution of a DMP for a placing movement is shown in Fig. 11.

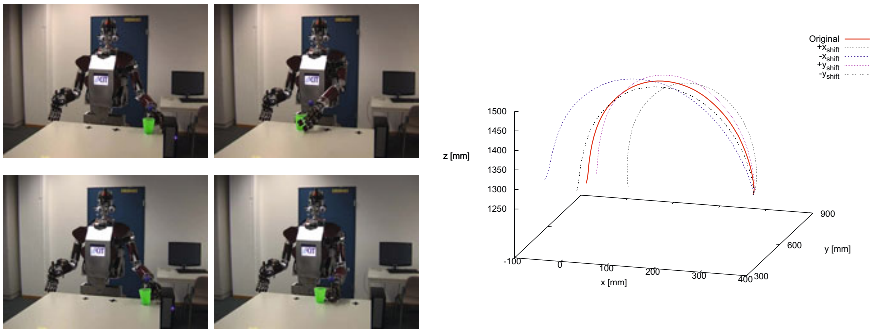


Fig. 11 Left: Execution of a place movement for different goal positions by the humanoid ARMAR-IIIb. Right: The red solid line represents the trajectory of the observed movement, while the other lines denote the adpation of the learned DMP to several goal positions.

7 Conclusion

In this paper, we have presented both past and current progress towards the realization of humanoid robots able to perform complex tasks in human-centered environments. We gave an overview about the mechatronics of the humanoid robots ARMAR-III as well as the methods and techniques applied to object recognition and localization, vision-based grasping and collision-free motion planning. In addition, we presented first steps of our research towards the acquisition of sensorimotor capabilities through multi-sensory exploration of the environment, observation of humans and goal-directed imitation of the observed movements.

Acknowledgements. The work described in this paper was partially conducted within the EU Cognitive Systems projects PACO-PLUS (FP6-027657) GRASP (FP7-215821) funded by the European Commission and the German Humanoid Research project SFB588 funded by the German Research Foundation (DFG: Deutsche Forschungsgemeinschaft).

References

1. Albers, A., Brudniok, S., Burger, W.: Design and development process of a humanoid robot upper body through experimentation. In: IEEE/RAS International Conference on Humanoid Robots, pp. 77–92 (2004)
2. Asfour, T., Azad, P., Gyarfas, F., Dillmann, R.: Imitation Learning of Dual-Arm Manipulation Tasks in Humanoid Robots. International Journal of Humanoid Robotics 5(2), 183–202 (2008)
3. Asfour, T., Regenstein, K., Azad, P., Schröder, J., Vahrenkamp, N., Dillmann, R.: ARMAR-III: An Integrated Humanoid Platform for Sensory-Motor Control. In: IEEE/RAS International Conference on Humanoid Robots (2006)
4. Asfour, T., Welke, K., Azad, P., Ude, A., Dillmann, R.: The Karlsruhe Humanoid Head. In: IEEE/RAS International Conference on Humanoid Robots, Daejeon, Korea, December 2008, pp. 447–453 (2008)
5. Azad, P.: Visual Perception for Manipulation and Imitation in Humanoid Robots. PhD thesis, Universität Karlsruhe (TH), Karlsruhe, Germany (2008)

6. Azad, P., Asfour, T., Dillmann, R.: Robust Real-time Stereo-based Markerless Human Motion Capture. In: IEEE/RAS International Conference on Humanoid Robots, Daejeon, Korea, December 2008, pp. 700–707 (2008)
7. Azad, P., Asfour, T., Dillmann, R.: Accurate Shape-based 6-DoF Pose Estimation of Single-colored Objects. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, USA (2009)
8. Azad, P., Asfour, T., Dillmann, R.: Combining Harris Interest Points and the SIFT Descriptor for Fast Scale-Invariant Object Recognition. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, St. Louis, USA (2009)
9. Azad, P., Asfour, T., Dillmann, R.: Stereo-based vs. Monocular 6-DoF Pose Estimation using Point Features: A Quantitative Comparison. In: Autonome Mobile Systeme (AMS), Karlsruhe, Germany (2009)
10. Azad, P., Ude, A., Asfour, T., Dillmann, R.: Stereo-based Markerless Human Motion Capture for Humanoid Robot Systems. In: IEEE International Conference on Robotics and Automation, Rome, Italy, April 2007, pp. 3951–3956 (2007)
11. Bay, H., Tuytelaars, T., Gool, L.V.: SURF: Speeded Up Robust Features. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3951, pp. 404–417. Springer, Heidelberg (2006)
12. Bierbaum, A., Rambow, M., Asfour, T., Dillmann, R.: A potential field approach to dexterous tactile exploration. In: International Conference on Humanoid Robots 2008, Daejeon, Korea (2008)
13. Bierbaum, A., Rambow, M., Asfour, T., Dillmann, R.: Grasp affordances from multi-fingered tactile exploration using dynamic potential fields. In: IEEE/RAS International Conference on Humanoid Robots (2009) (accepted to)
14. de Leva, P.: Adjustments to zatsiorsky-seluyanov's segment inertia parameters. Journal of Biomechanics 29(9), 1223–1230 (1996)
15. Deutscher, J., Blake, A., Reid, I.: Articulated Body Motion Capture by Annealed Particle Filtering. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Hilton Head, USA, pp. 2126–2133 (2000)
16. Do, M., Azad, P., Asfour, T., Dillmann, R.: Imitation of human motion on a humanoid robot using nonlinear optimization. In: International Conference on Humanoid Robots, Humanoids 2008 (2008)
17. Chaumette, F., Hutchinson, S.: Visual servo control, part I: Basic approaches. IEEE Robotics and Automation Magazine 13(4), 82–90 (2006)
18. Geib, C., Mourao, K., Petrick, R., Pugeault, N., Steedman, M., Krüger, N., Wörgötter, F.: Object action complexes as an interface for planning and robot control. In: Workshop: Toward Cognitive Humanoid Robots. IEEE-RAS International Conference on Humanoid Robots (December 2006)
19. Göger, D., Weiß, K., Burghart, K., Wörn, H.: Sensistive skin for a humanoid robot. In: International Workshop on Human-Centered Robotic Systems, (HCRS 2006), Munich (2006)
20. Harris, C., Stephens, M.: A Combined Corner and Edge Detector. In: Alvey Vision Conference, Manchester, UK, pp. 147–151 (1988)
21. Ijspeert, A.J., Nakanishi, J., Schaal, S.: Movement imitation with nonlinear dynamical systems in humanoid robots. In: Proceedings of the IEEE International Conference on Robotics and Automation (2002)
22. Krüger, N., Piater, Wörgötter, F., Geib, C., Petrick, R., Steedman, M., Ude, A., Asfour, T., Kraft, D., Omrcen, D.D., Hommel, B., Agostino, A., Kragic, D., Eklundh, J., Kruger, V., Dillmann, R.: A formal definition of object action complexes and examples at different levels of the process hierarchy. Technical report (2009)

23. Kuffner, J., LaValle, S.: RRT-connect: An efficient approach to single-query path planning. In: IEEE International Conference on Robotics and Automation (April 2000)
24. LaValle, S., Kuffner, J.: Randomized kinodynamic planning. Internation Journal of Robotics Research 20 (June 2001)
25. Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision (IJCV) 60(2), 91–110 (2004)
26. PACO-PLUS. Perception, action and cognition through learning of object-action complexes, http://www.paco-plus.org
27. Pastor, P., Hoffmann, H., Asfour, T., Schaal, S.: Learning and generalization of motor skills by learning from demonstration. In: Proceedings of the IEEE International Conference on Robotics and Automation, Kobe, Japan (2009)
28. Schulz, S., Pylatiuk, C., Kargov, A., Oberle, R., Bretthauer, G.: Progress in the development of anthropomorphic fluidic hands for a humanoid robot. In: IEEE/RAS International Conference on Humanoid Robots, Los Angeles (November 2004)
29. Ude, A., Atkeson, C.G., Riley, M.: Programming Full-Body Movements for Humanoid Robots by Observation. Robotics and Autonomous Systems 47(2-3), 93–108 (2004)
30. Vahrenkamp, N., Barski, A., Asfour, T., Dillmann, R.: Planning and execution of grasping motions on a humanoid robot. In: 9th IEEE-RAS International Conference on Humanoid Robots, Humanoids 2009(December 2009)
31. Vahrenkamp, N., Berenson, D., Asfour, T., Kuffner, J., Dillmann, R.: Humanoid motion planning for dual-arm manipulation and re-grasping tasks. In: Intelligent Robots and Systems, IROS (October 2009)
32. Vahrenkamp, N., Böge, C., Welke, K., Asfour, T., Walter, J., Dillmann, R.: Visual servoing for dual arm motions on a humanoid robot. In: 9th IEEE-RAS International Conference on Humanoid Robots, Humanoids 2009 (December 2009)
33. Vahrenkamp, N., Wieland, S., Azad, P., Gonzalez, D., Asfour, T., Dillmann, R.: Visual servoing for humanoid grasping and manipulation tasks. In: 8th IEEE-RAS International Conference on Humanoid Robots, Humanoids 2008, pp. 406–412 (December 2008)
34. Weghe, M.V., Ferguson, D., Srinivasa, S.: Randomized path planning for redundant manipulators without inverse kinematics. In: IEEE-RAS International Conference on Humanoid Robots (November 2007)
35. Weiss, L., Sanderson, A.C., Neuman, C.P.: Dynamic sensor-based control of robots with visual feedback. IEEE Journalon Robotics and Automation RA-3(5) (October 1987)
36. Welke, K., Asfour, T., Dillmann, R.: Active multi-view object search on a humanoid head. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA 2009), pp. 417–423 (2009)
37. Welke, K., Issac, J., Schiebener, D., Asfour, T., Dillmann, R.: Autonomous acquisition of visual multi-view object representations for object recognition on a humanoid robot. In: Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2010 (2010) (submitted to)
38. Welke, K., Oztop, E., Cheng, G., Dillmann, R.: Exploiting similarities for robot perception. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3237–3242 (2007)
39. Wilson, G.b.W.J., Williams Hulls, C.C.: Relative end-effector control using cartesian position based visual servoing. IEEE Transactions on Robotics and Automation 12, 684–696 (1996)
40. Wörgötter, F., Agostini, A., Krüger, N., Shylo, N., Porr, B.: Cognitive agents – a procedural perspective relying on the predictability of Object-Action-Complexes (OACs). Robotics and Autonomous Systems 57, 420–432 (2009)

Enhanced Mother Environment with Humanoid Specialization in IRT Robot Systems

Masayuki Inaba, Key Okada, Tomoaki Yoshikai, Ryo Hanai, Kimitoshi Yamazaki, Yuto Nakanishi, Hiroaki Yaguchi, Naotaka Hatao, Junya Fujimoto, Mitsuharu Kojima, Satoru Tokutsu, Kunihiko Yamamoto, Yohei Kakiuchi, Toshiaki Maki, Shunnichi Nozawa, Ryohei Ueda, and Ikuo Mizuuchi

Abstract. In the research to realize high standard task-oriented assistant robots, a general and strategic way of development is essential. Otherwise high functionality and potential for evolution of those robots cannot be achieved. Robotic systems are socially expected to assist our daily life in many situations. As a result, projects related to those robots are becoming large, involving many researchers and engineers of universities and companies. This motivated us a new strategy to construct robotic systems based on *mother environment* and humanoid *specialization* to keep developing and refining functional elements of robots in an evolutionary way. The mother environment is an entity that creates brains of humanoid robots, where various robotics function elements, libraries, middle-wares and other research tools are integrated. Then the brain of each robot is developed utilizing the functional elements in the mother. We call this process specialization of a humanoid. To enhance this specialization process, we introduce a generator, which realizes conversion of functions in the mother environment for the real-time layer. After the research of these specific robots, enhanced robotics functions are incorporated into the mother again. We call this process *feedback*. In this chapter, we present these ideas using concrete implementation examples in IRT projects[1], where several robots to assist our daily life are developed.

1 Introduction

A general and strategic way of development is essential in the research of task-oriented daily assistant robots, because they are expected to have high functionality and potential for continuous evolution. Developing each robot in a specific, restricted development environment is not a good idea. Instead, designing the development environment itself to have a set of generalized functions and large potential

Masayuki Inaba
The University of Tokyo, Eng.bldg.2-73A1, 7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan
e-mail: inaba@jsk.t.u-tokyo.ac.jp

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 379–396.
springerlink.com

for evolution is better. In this method, part of the generalized functions are modified and enhanced depending on each task and then incorporated again into the developmental environment. This framework to incorporate the mother environment into the cycle of enhancement and continuous development is important for evolution of robot functions, which is required repeatedly.

Practical task-oriented assistant robots are prone to hardware and environment dependent and expected assistive tasks and situations they work have a large variety. Therefore the standard of required functions becomes higher and more sophisticated incessantly and higher level of development environments are required as well. As for the development environments, however, there are many functions and structures used in common, even though tasks differ. Conversely, specific task-oriented robots can stop their development before evolving to the standard of practical use unless they have development environment with high potential.

Then what is the methodology of developing these robot systems with generality? Our approach is to share mother environment and integrate it into the evolution cycles of humanoid functions. The mother is a development platform for research of humanoid functions where variety of functions are integrated. In order to extend elemental functions in humanoids further, we let the mother have a mechanism for specialization to generate specific task-oriented robots from humanoids. The functions are specialized and intensified for each robot. Then we experiment and evaluate them through experiments in the robot and next generation humanoids, accumulating the result in the mother environment.

The rest of this chapter is structured as follows. First, we introduce the idea of the mother environment in the next section and overview software infrastructure that plays important role in continuous development of robot software. Next, section 3 describes additional functions we have implemented recently for IRT projects. Section 4 presents robots developed in the IRT project and explains how the software was developed. After that, the idea of feedback is stated with concrete examples in 5. Finally, we conclude this chapter in 6.

2 Mother Environment

In the development over many years by many researchers, to keep research tools easily accessible and usable is very important. Research tools include robotics functions, libraries, middlewares and so on. Further to keep enhancing robotics functions, we believe testing them in various robots and conditions is also effective. Our approach is to make them accessible in a Lisp-based development environment. Fig.1 shows our research history on the environment. COSMOS[2][3] stands for cognitive sensor motor operation studies and is the first Lisp-based research platform in our lab (JSK). On the COSMOS system, vision-based research had progressed[4] in rope handling[5], vision-based programming[6], teaching by showing[7] and so on.

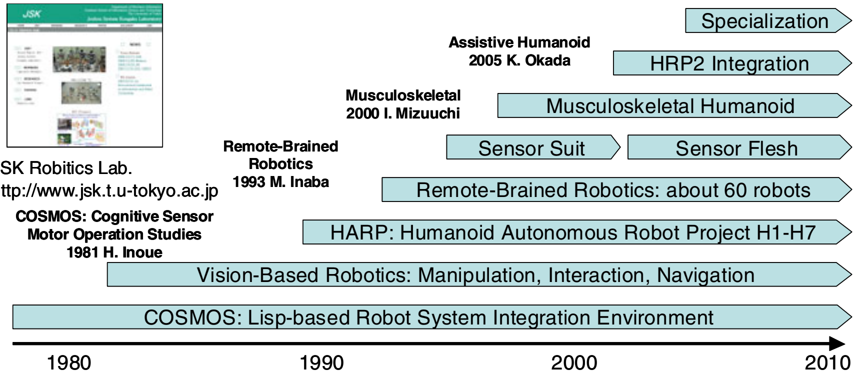


Fig. 1 Research history in JSK

In order to progress advanced vision based research, we adopted new object-oriented Lisp system, EusLisp[8], which has 3D geometric modeler. With this new powerful Lisp system and a new approach of remote brained robots, we could start research on *mother environment* [9][10][11]. The mother environment is a development environment with various software functions and grows *brains* of many robots. Here, brains mean software of robots that generates intelligent behaviors of them. Fig.2 shows the development flow of the research of the mother environment. In the mother environment, robot researchers share and revise software modules with different kind of robot bodies and tasks. Then the functions newly developed or enhanced for the specific robots are feedbacked and enhance the mother itself.

This chapter describe the specialization flow from the integrated humanoid, HRP2JSK to IRT robot systems and the integration into new generation of HRP2JSK series.

2.1 Humanoids in JSK Robotics Lab

To give a concrete image of robots we are working on, we show some of them in Fig.3. The development started with *Remote-brained robot* series[12][13][14] some decades ago. These robots had main computers outside of the bodies because it was difficult to embed high performance computers in the robots' bodies. On the other hand, by the appearance of embeddable, smaller and more powerful PCs, it became possible to start a life-size humanoid autonomous project: HARP[15]. The seventh prototype of HARP, H7[16] had an on-board PC with dual CPUs. Vision-based humanoid behaviors were realized in HARP [17][18][19]. Control software of H7 became a basis of *HRP2* which is the final prototype of the human cooperative project[20]. In our lab (JSK Lab.), we have been developing a series of HRP2JSK[21][22] through refining HRP2 as a Lisp-based integration platform for daily assistive tasks that require complex recognition and motion generation. In the

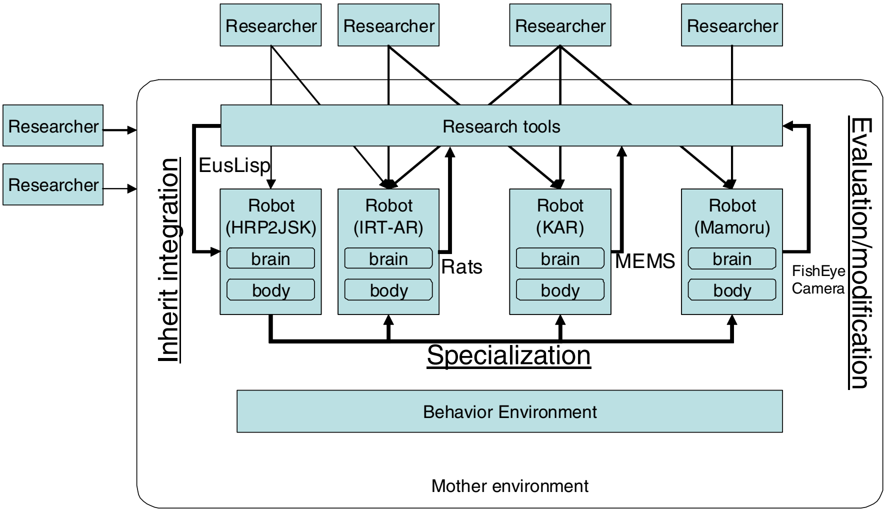


Fig. 2 Mother environment and development process

Fig.3(d), HRP2JSK is cleaning the floor with a cleaner based on vision. Cla[23] is a first remote-brained robot with spine. Kenta is the first prototype of a life-size musculo-skeletal humanoid[24][25]. Kotaro[26] is the first musculo-skeletal humanoid with collar and blade bones. Their latest version, Kojiro[27] has as many as 82 DOFs and 109 wire-driven muscles. In the figure, it is hitting a Japanese drum. Research interest is mainly in how to construct and control this extremely complex robotic hardware. In addition to the spined body structure, we have worked on tactile sensor suit[28] to cover the whole body of a robot using electrical conducted fabric. The robots named 'macra' and 'macket'[29][30] are developed for 'sensor flesh', which has soft thick exterior embedded with distributed multi-axis tactile sensors. For realizing various contact state recognition during very close interaction between humans or environments and a robot, such kind of sensor exterior is necessary. Sensing elements inside of the 'sensor flesh' are improved in macket. Implementation of the sensor flesh is improved in macket. It uses 3D soft deformable sensors newly developed for ourselves, solving many problems which arise when we use commercially available 3-axis force/torque rigid sensors for macra. The rightmost one is a prototype of a high power humanoid leg[31]. A main feature of this leg is that it employs current vector control and liquid-cooling to drive 200[W] motors. It can lift up and down as heavy as 62.5[kg] of weight.

2.2 Lisp-Based Development Environment

The robots presented in section 2.1 share the most part of software in common and developed using the same software infrastructure. What is important is that robotics functions are integrated in Lisp based programming environment, EusLisp[8][32]

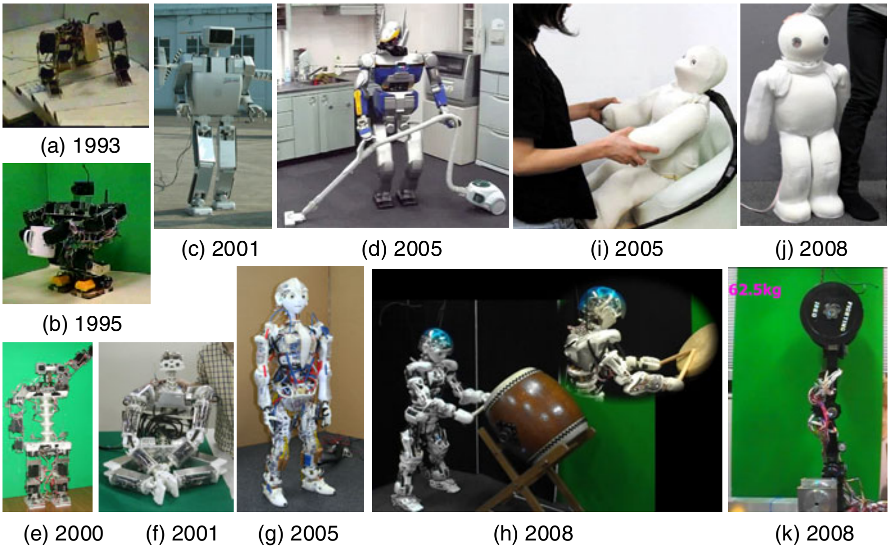


Fig. 3 Remote-brained robot(a),(b),(e), H7(c), HRP2(d), Cla(e), Kenta(f), Kotaro(g), Kojiro(h), macra(i), macket(j) and High power leg(k) developed in JSK Robotics Lab, The University of Tokyo.

from the higher level to the real-time reactive layer(Fig.4). Here, the higher level consists of functions such as planner, vision, auditory, inference, prediction, goal management and execution control. It uses memory layer as an internal expression of the robot and environment. The real-time reactive layer consists of functions such as reactive control and sensor motor body monitoring. It usually consists of periodic, concurrent processes and requires real-time response. Examples are Hrpsys of OpenHRP[33][34] and Nervous System(NS), which is a plug-in system with the similar design and interface with Hrpsys.

Body models (virtual bodies) and real bodies are designed to offer the same interface to the programmer. Consequently by switching a real robot and a body model, the same application run. This is done, typically by changing a line in the application programs. Models, algorithms and behavior programs to realize tasks are written in EusLisp itself. On the other hand, real-time reactive layer is implemented in languages like C or C++ and controlled from EusLisp. That is, it offers interface to EusLisp. A lot of useful external libraries are integrated into EusLisp using foreign language interface. Foreign language interface is a specification to link and execute program of different execution model. Most of recent programming languages support. Examples are JNI in Java and Boost Python in Python. Common Lisp implementations such as Allegro Common Lisp or SBCL also support this. Examples of external libraries are algorithms of image processing(OpenCV[35]), collision detection(PQP[36]), physical calculation, sound recognition, numerical calculation, and libraries for various devices. Using these techniques, every layer is integrated under consistent interface design of EusLisp.

We will add some more explanations of EusLisp since it is a core of the integration. EusLisp was developed by Matsui et al. to meet a demand for an extensible solid modeler that can easily be made use of from higher level symbolic processing system because an essence of the robotics research is sensory data processing where 3D shape models of robots and environment play crucial roles. In addition to it, it supports several functions required for robot programing such as multi-threading, object-orientation. foreign language interface and graphics. Object system of EusLisp is single inheritance model, where every data structure except number is represented by an object. Graphics means XWindow and OpenGL support. EusLisp provides users with common interface to various software like robot models, sensors and other functions, which enables new comers to the lab. to use robots in short time. External dynamics engines such as ODE[37], PhysX[38] are also integrated into EusLisp and the engine is controlled from EusLisp interactively.

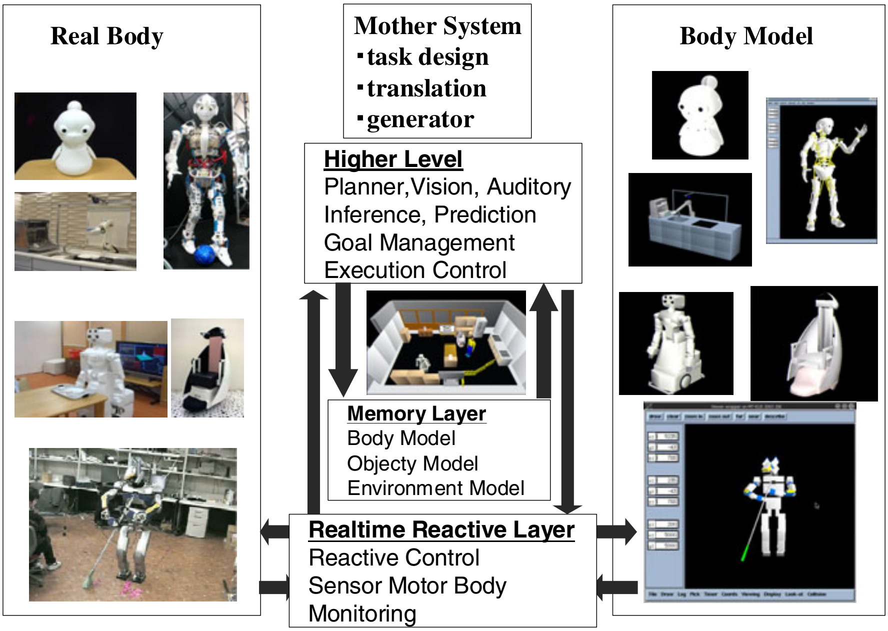


Fig. 4 Lisp-based integration environment for humanoids and IRT systems robot

3 Enhancement of the Mother Environment Modules

We have extended the Lisp-based environment in three ways these days to support broader ways of structuring the system. The first one is a framework to develop software using the C/C++ program generated from the Lisp-based environment. The second one is an improvement of real-time response of the Lisp-based environment.

And the last one is interface to integrate the Lisp-based environment with robotics middleware, which enable us to structure the robot system in more flexible way. In the rest of this section, we describe these extensions in further detail.

3.1 Generator for Humanoid Specialization (RATS)

Lisp has a novelty that we can write program and execute it interactively, without rebooting the system. However, when we want to implement lower layer functions, this has problems. Memory management of Lisp can cause a long pause time for timing sensitive applications. Small footprint is required for embedded robotics applications. Furthermore, though this is not only true for lower layer, faster execution speed is favorable, especially for computations using matrices and vectors.

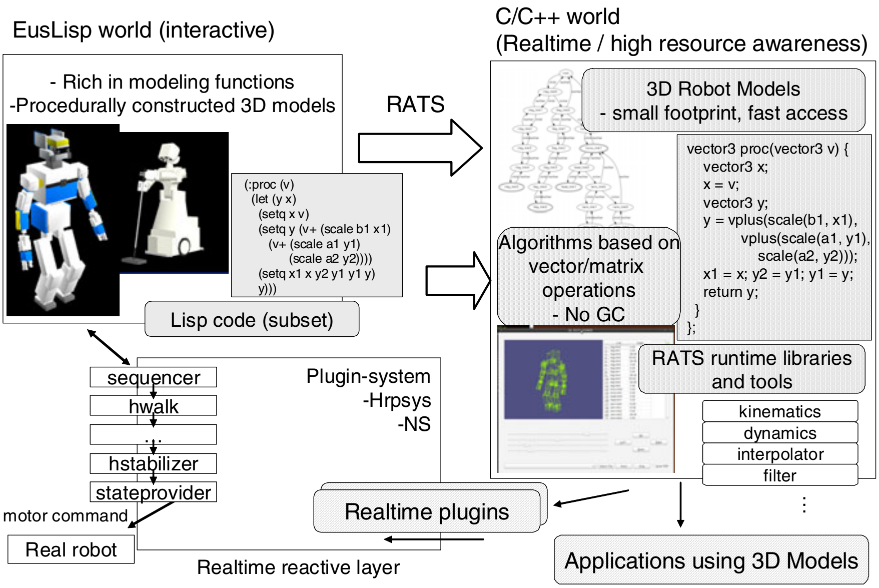
This motivated us to develop system to export model-based program written in EusLisp in a way that suits the requirements of the lower layer functions, where timing and resource issues are more important. Fig.5 shows a conceptual diagram of this system and the development process using it.

Firstly, the model description of robot is translated. In EusLisp, models of robots are written procedurally in Lisp program. Analyzing the structure of robots from the Lisp code statically is difficult, but once the structure of the robot is fixed, it does not change usually while executing application programs. In addition, the description of robot is standardized in the lab by imposing several conventions to model developers. Therefore, models used in EusLisp can be dumped by analyzing the structure constructed in memory at runtime and interpreting several classes showing the structure of robots such as links and joints. Models are allocated dynamically in EusLisp, but this causes a lot of overhead since accesses to models require several method calls and symbol look-ups. Therefore, in dumped program they are allocated statically to realize fast access. This improves locality of the model structures in memory and reduces footprint as well.

Secondly, a restricted set of EusLisp program that does some computation using robot models are translated. What we would like to do here is to convert EusLisp program to C/C++ program without high-functional runtime memory management that runs fast and requires small footprint. However, converting general Lisp program to the program stated above is not possible. So we used a similar approach to the one adopted by Embedded MATLAB[39]. The point of translation is as follows.

- only restricted set of constructs, primitives are allowed
- annotation of static types by programmer is required
- give up following Lisp semantics correctly, and interpret many constructs in the most common and conventional meanings statically at the time of translation

Thirdly, we prepared runtime libraries and tools such as a viewer for this framework from scratch. Some functions can be converted from the code of EusLisp, but some programs are difficult to convert and some functions do not have the same functions in EusLisp. By using these three components: dumped robot models, translated C/C++ program and runtime libraries written C/C++, users can easily and efficiently develop a stand-alone C/C++ program, or real-time controller for plug-in systems such as OpenHRP.


EusLisp world (interactive)
- Rich in modeling functions
-Procedurally constructed 3D models
Lisp code (subset)
RATS
C/C++ world
(Realtime / high resource awareness)
3D Robot Models
- small footprint, fast access
Algorithms based on vector/matrix operations
- No GC
RATS runtime libraries and tools
kinematics
dynamics
interpolator
filter
⋮
sequencer
hwalk
...
hstabilizer
stateprovider
Plugin-system
-Hrpsys
-NS
motor command
Real robot
Realtime plugins
Realtime reactive layer
Applications using 3D Models

```
(:proc (v)
  (let (y x)
    (setq x v)
    (setq y (v+ (scale b1 x1)
      (v+ (scale a1 y1)
          (scale a2 y2))))
    (setq x1 x y2 y1 y1 y)
    y)))
```

```
vector3 proc(vector3 v) {
   vector3 x;
   x = v;
   vector3 y;
   y = vplus(scale(b1, x1),
          vplus(scale(a1, y1),
            scale(a2, y2)));
   x1 = x; y2 = y1; y1 = y;
   return y;
  }
};
```

Fig. 5 Framework to develop using C/C++ program generated from the Lisp-based environment

3.2 *Real-Time Garbage Collection Support to Improve Realtime Response of Lisp-Based System*

For an application to be real-time, underlying programming system needs to be real-time as well as OS. On the other hand, applications generate a lot of garbage memory blocks while executing. For example, most of the computation in Lisp generates intermediate data such as vectors, matrices or strings. Part of large data structures like search trees of planning, 3D geometrical models of robots, environments and objects to manipulate can be garbage as well. Hence, the system needs to recycle those unused blocks. As a result, applications stop its execution while this recycle process (GC) is occurring. We have implemented a GC algorithm to reduce this pause times of applications. The algorithm is based on the snapshot algorithm[40] and executes GC concurrently[41] with applications using a separate thread. It uses the write barrier technique to trap writes of applications and realize a consistent heap image for GC process.

3.3 Interoperability with Robotics Middleware

The use of robotics middleware has been becoming popular in recent years, especially for the development of a large, distributed system. Some of famous middlewares would be ROS[42] by Willow Garage, RT-Middleware by AIST and MSRS[43] by Microsoft. Sometimes by utilizing these middleware, the effort to develop system decreases a lot. Using package or component libraries of the middleware is also useful. To make use of these benefits, we added functions to smoothly connect EusLisp and these middlewares and compose the whole system.

Fig.6 shows different ways of structuring the system. The left column(A) is an integration on EusLisp. The right column(B) is an integration using software parts generated by RATS. (A-a) is a typical structure to develop the whole functions of the application in EusLisp. Each asynchronous function is expressed as a thread and these threads communicate with each other using shared memory. The higher layer of many robots including HRP2-JSK is implemented like this. (A-b) is the structure using extensions to use distributed middleware. Two ways of connections are realized. One way is a function to manage components from EusLisp. For example, components are activated/deactivated in EusLisp. The other way is a function to use EusLisp itself as a component or components of middleware. Program running on EusLisp behaves as a component and communicate with other components using the interface defined in the middleware framework. 'macra' and 'macket' use JRTM, an integrated Lisp programming environment with RT-middleware architecture[44][44]. Among IRT robots, Mamoru described in the next section uses this. By using ROS-Eus, which is ROS interface of EusLisp, we realized a demonstration in one week by combining functions of autonomous navigation in ROS with functions of audio visual interaction, human detection and hand shaking motion in EusLisp. Generally real-time response of the system in the left column is improved by the use of real-time GC described in the previous subsection.

The right column is usually the way of structuring the real-time reactive layer. Conventionally plug-ins are written in C++, but RATS helps develop plug-ins using robot models. Programmers are able to use robot models and algorithms developed in EusLisp in their plug-in code by using the generator. (B-a) is a single loop plug-in architecture. In this plug-in architecture, functions like collision check or motion interpolation are registered on demand as dynamic link libraries and the system executes the functions periodically in a specified order. Nervous system(NS) of Kotaro/Kojiro and Hrpsys of HRP2 use this architecture. KAR uses the nervous system in its reactive layer. We also developed Hrpsys compatible system for AR. The reactive layer of AR is complex. It uses a distributed middleware of Toyota in the embedded OS (B-b), too.

As you can see that one robot appears in some pictures, integration methods described here are combined with each other to construct a larger system.

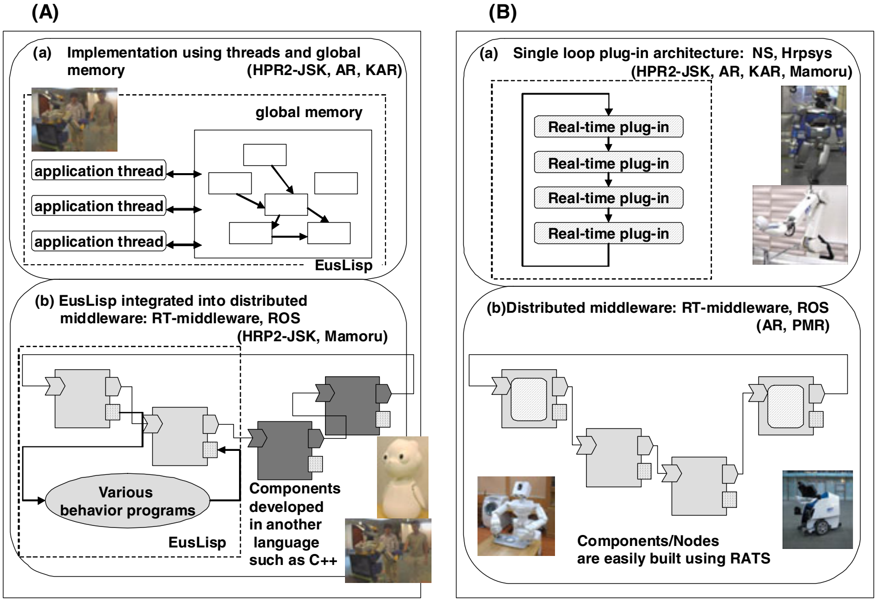

(A)
(a) Implementation using threads and global memory (HPR2-JSK, AR, KAR)
global memory
application thread
application thread
application thread
EusLisp
(b) EusLisp integrated into distributed middleware: RT-middleware, ROS (HRP2-JSK, Mamoru)
Various behavior programs
EusLisp
Components developed in another language such as C++
(B)
(a) Single loop plug-in architecture: NS, Hrpsys (HPR2-JSK, AR, KAR, Mamoru)
Real-time plug-in
Real-time plug-in
Real-time plug-in
Real-time plug-in
(b)Distributed middleware: RT-middleware, ROS (AR, PMR)
Components/Nodes are easily built using RATS

Fig. 6 (A)Structure using EusLisp, (B)Structure using RATS-generated program (Realtime layer)

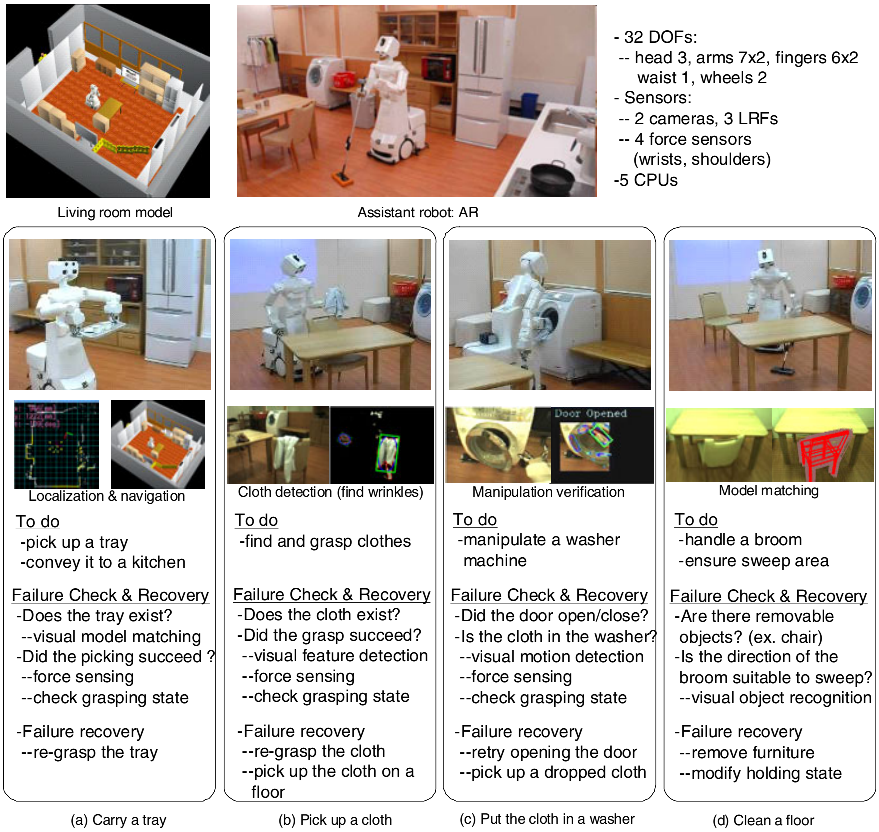
4 Development of IRT Specialized Robots

IRT is a project that started about three years ago. In the project, a group of the University of Tokyo is working with several companies such as Toyota, Fujitsu, Panasonic. The objectives are to support aging society using Information Technology(IT) and Robot Technology(RT), meaning IRT. Therefore each robot has very practical, specific tasks with high expectation in the society and its hardware is not necessarily humanoid but designed to well suit the tasks. This section describes three robots from the aspects of hardware, system architecture, and expected tasks.

We also focus on how the software infrastructure was used. Functions integrated to the humanoid platform we have developed were used to develop these robots. Sometimes they were used as they are or changed to develop program that are more suitable to the specific robot. Sometimes they were exported to the specific robot using program translation techniques. We call this idea *humanoid specialization*.

4.1 Tidy-Up Assistant Robot – AR

A tidy-up assistant robot is a prototype to help our life by covering several houseworks. Upper body consists of a head (3 DOFs), a waist (1 DOF) and 2 arms (7 DOFs), and its end-effector equips 3 fingers which are composed of 2 joints. Lower body is 2 wheeled mobile platform. These configuration enables the robot to achieve



- 32 DOFs:
-- head 3, arms 7x2, fingers 6x2
waist 1, wheels 2
- Sensors:
-- 2 cameras, 3 LRFs
-- 4 force sensors
(wrists, shoulders)
-5 CPUs

Living room model

Assistant robot: AR

Localization & navigation

To do
-pick up a tray
-convey it to a kitchen

Failure Check & Recovery
-Does the tray exist?
--visual model matching
-Did the picking succeed ?
--force sensing
--check grasping state

-Failure recovery
--re-grasp the tray

Cloth detection (find wrinkles)

To do
-find and grasp clothes

Failure Check & Recovery
-Does the cloth exist?
-Did the grasp succeed?
--visual feature detection
--force sensing
--check grasping state

-Failure recovery
--re-grasp the cloth
--pick up the cloth on a floor

Manipulation verification

To do
-manipulate a washer machine

Failure Check & Recovery
-Did the door open/close?
-Is the cloth in the washer?
--visual motion detection
--force sensing
--check grasping state

-Failure recovery
--retry opening the door
--pick up a dropped cloth

Model matching

To do
-handle a broom
-ensure sweep area

Failure Check & Recovery
-Are there removable objects? (ex. chair)
-Is the direction of the broom suitable to sweep?
--visual object recognition

-Failure recovery
--remove furniture
--modify holding state

(a) Carry a tray

(b) Pick up a cloth

(c) Put the cloth in a washer

(d) Clean a floor

Fig. 7 Tidy-up assistant robot (AR)

both moving on wide range quickly and dextrous object handling. On the other hand, this robot mounts a stereo camera on the head, and a LRF (Laser Range Finder) on the wheelbase. Force sensors are equipped on wrists and shoulders of the both arms. The robot hardware was developed by Toyota Motor Corporation.

We designed and developed software system and proved the system by implementing a demonstration to perform several houseworks sequentially. One of the important things to develop helpful robots which works in real environment is to let it have abilities to handle various failures while working. Because our system provides a wealth of recognition functions and pre-defined environment information, this enables the robot to detect mistakes of chores and to plan to recover the condition on the spot[45]. The bottom row of Fig.7 shows tasks and failure detection methods while conducting the tasks together with recovery methods tried after detecting those failures.

The geometrical simulator and motion planning parts are implemented by using EusLisp (keeping with past framework). Although all of recognition functions and motion commands are implemented by C/C++, they were also promoted to call from EusLisp through interface functions. This means that all of task level programming can be written by Lisp and it enables us to develop the demonstration interactively. As for recognition, model based recognition of EusLisp was used to detect objects such as chairs, a button of a washing machine. In addition, we developed a new image based method for clothes by utilizing wrinkle feature[46].

4.2 Kitchen Assistant Robot – KAR

This is an arm type kitchen assistant robot(Fig.8) [47][48]. This has 8 DOFs: 1 for a slider, 6 for the arm, 1 for the hand. The intended task of this robot is dexterous handling of dishes, which is an essential skill to help us with daily housework. One example task sequence is to carry several kinds of dishes put on a tray to a dish washer after rinsing them in the water. The robot hardware is developed by Panasonic Co.,Ltd and we designed and developed hands suitable for dish handling. For this task, many sensors are embedded in its end effector. one 6-axis force sensor is in the wrist. 6 MEMS tactile sensors[49], two 3-axis force sensors, 5 proximity sensors and eight pressure sensors are on the palm and the thumb.

We developed a manipulation method based on groping, which means motion planning is combined with sensing and recognition of manipulation targets using these sensors. This technology enabled the robot to manipulate fragile, sometimes stacked dishes in a skillful way, resulting in the realization of clearing dishes, which is one of the most expected housework for service robots. The system is designed as follows. To handle dishes dexterously in a cluttered environment, the robot needs to adjust motion quickly responding to sensor inputs. On the other hand, however, to achieve tasks of high-level objectives, sequential program flow is favorable. Then, a key idea is to separate main program control flow from troublesome monitoring tasks. Monitors are executed concurrently in separate threads and keep checking the status of work, detect errors and unexpected events while performing tasks. A framework to create and delete a proper set of monitors depending on task phases was developed. The events from monitors are processed using a ring queue.

Since the robot is a kind of an arm of a humanoid, we could easily implement the geometrical simulator and robot models using EusLisp. Functions like RRT-connect motion planning, collision detection existed in EusLisp was also used. Its real-time reactive layer is composed of Nervous plug-in system (NS).

4.3 Recall Assistant Robot – Mamoru

This is a small robot like a kid or an animal character. The intended task is daily watching of people's life and giving useful advises or presenting information at a proper timing. The height is 451[mm]. Each arm has 2DOFs and flutters. The neck has 2DOFs. 2 multi-view cameras are embedded in the eyes. It has 16ch microphones and a speaker to interact with humans and detect events in the room.

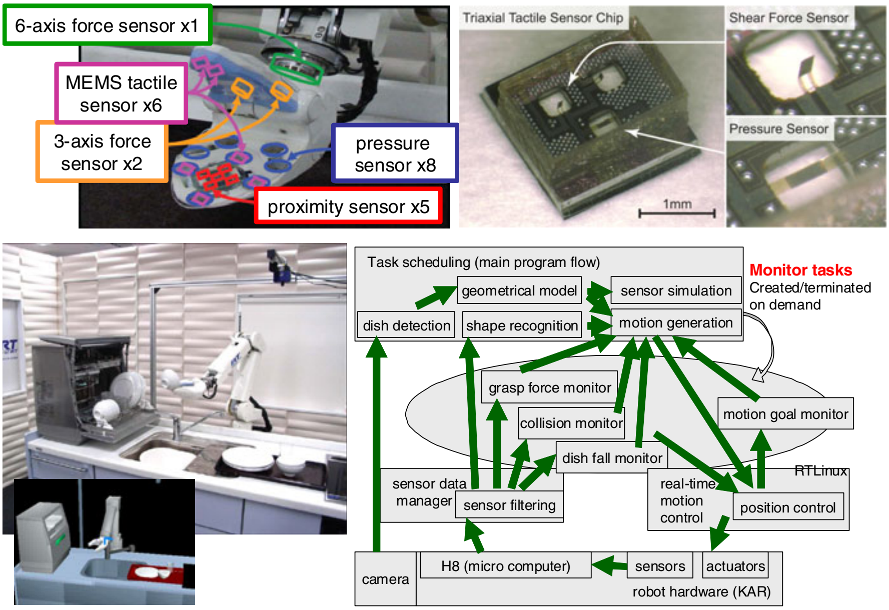


Fig. 8 Kitchen assistant robot (KAR)

To watch the behavior of people living in the room and detect events such as opening and closing of doors or drawers, the robot has to have both wide view angle and good eyesight. In addition, it is favorable to be able to have several region of interests because the robot can keep watching background human activities while communicating with a specific human which often requires high resolution for example to recognize his or her face. This was realized by a small DSP embedded programmable camera that sends images of specified region of interests and resolutions interleavingly, realizing efficient USB traffic. This robot has a new embedded tracking vision module developed by Fujitsu Co.,Ltd [50]. This tracking vision provides realtime 3D optical flow generation based on normalized correlation image processing. The performance of the tracking capability was presented in the original vision system[51]. Design issues are also important for this robot. Since it lives with people for long time, it need to be liked by various people from children to elderly people.

To develop the robot, software libraries for image processing and audio interaction such as sound localization, voice recognition inherent in the humanoid platform were used. NS was used to implement its real-time reactive layer. As for enhancement, it used JRTM, which is an RT-middleware interface of EusLisp.

One of the tasks it realized was giving a notice to people who tried to take medicine soon after they took it once. For this task, we developed several functions, for example, recognition of human behavior based on location related semantic information, recognition of taking medicine behavior using multi-view angle cameras and recognition of product class, name, freshness date from bar-code/characters.

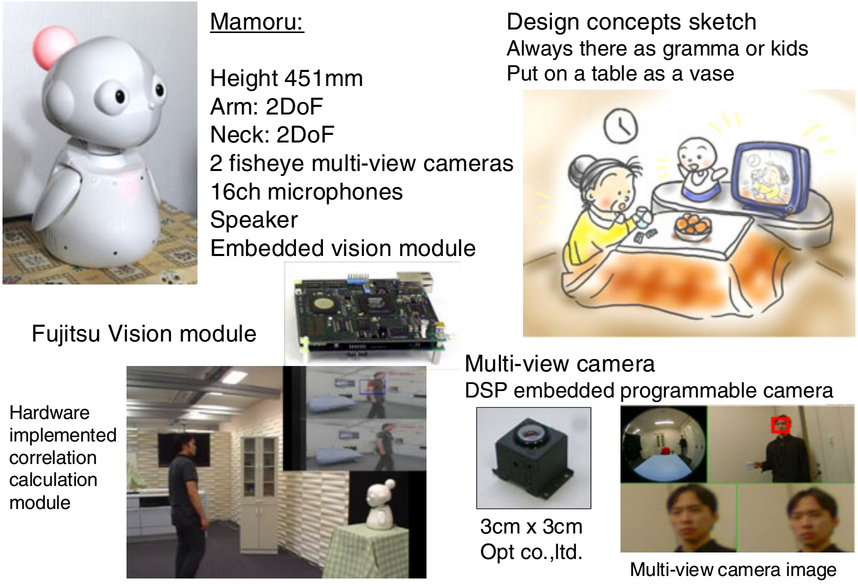


Fig. 9 Recall assistant robot

5 Feedback to the Mother Environment

For the continuous development, elemental functions enhanced in specialized robots are feedbacked to the mother environment to work in the humanoid platform together with some new devices. Fig.10 shows the concept of this specialization and feedback process.

Mamoru used functions for audio visual interaction that had existed on HRP2JSK as a specialization. Then through IRT development the audio visual interaction was enhanced with multi-resolution, multi region of interests support cameras and the vision hardware module and feedbacked to the mother[52]. Now these functions can easily be used to describe tasks on the humanoid platform as the feedback process. In the case of KAR, tightly coupled sensor motor interaction on multi-modal sensors developed for KAR hand is feedbacked into the mother modules for the next humanoid integration. The architecture to execute parallel monitoring tasks and get noticed by them were also incorporated. The parallel structure module provides dexterous, groping-based handling capability on the humanoid. In the same way, AR used vision-based task execution functions integrated in EusLisp. Feedbacks from AR to new humanoid development in mother environment include an architecture to describe systematic failure-recovery control flow, new visual recognition method of clothes, and the way of writing task sequences. The rightmost column of Fig.10 shows HRP2s after these feedback enhancements. HRP2JSK-NT shows hardware integration. It has 2 multi resolution cameras, 1 stereo camera and 2 high-resolution cameras for visual processing. It also has omni-directional audio and is equipped with a sensor embedded hand. The system of HRP2JSK-VZ consists of several middlewares and integrated using EusLisp. It uses a navigation stack of ROS, components of RT-middleware for vision and head motion control and OpenHRP to control the upper body.

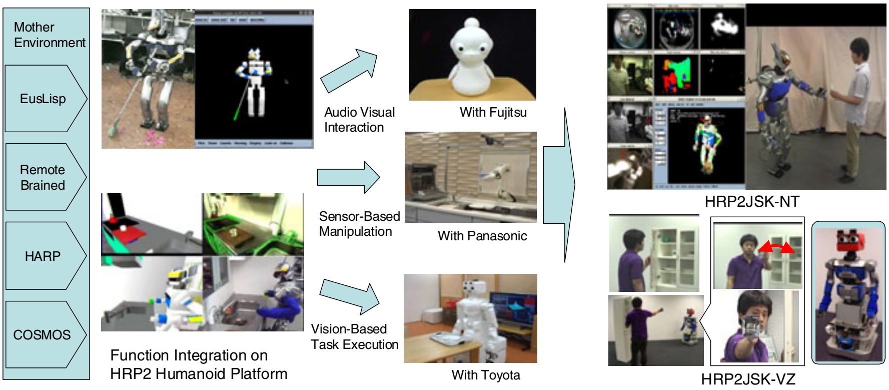


Fig. 10 Specialization and Feedback

6 Concluding Remarks

In this chapter, we presented a development method based on the mother environment and humanoid specialization to realize high level of assistant robots. The mother environment is a development platform for research of humanoid functions where variety of functions are integrated. In our case, this is a Lisp-based environment and various humanoid functions are used through the common interface. Then we introduced a mechanism to generate specific task-oriented robots from humanoids by specializing and intensifying functions in the mother. As concrete examples of the specialization, we showed three different types of robots, AR for task sequence planning and execution, KAR for tightly coupled sensor motor interaction in dexterous dish manipulation, and Mamoru for vision based recognition of human behavior with real-time tracking vision in multiple narrow and wide views. Finally, we explained the process of feedbacks to integrate enhanced functions into the new generation of humanoids from the specialized tasks oriented IRT robot systems. HRP2JSK-NT and HRP2JSK-VZ are the new vehicles of research on mother environment in our laboratory JSK.

References

1. IRT Research Initiative, http://www.irt.i.u-tokyo.ac.jp/index_e.shtml
2. Ogasawara, T., lnoue, H.: Cosmos: A total programming system for integrated intelligent robot study. Journal of Robot Society of Japan 2(6), 507–525 (1984)

3. Inoue, H.: Building a bridge between ai and robotics. In: Proceedings of International Joint Conference on Artificial Intelligence, pp. 1231–1237 (1985)
4. Inoue, H.: Vision based robot behavior: Tools and testbeds for real world ai research. In: Proceedings of International Joint Conference on Artificial Intelligence, pp. 767–773 (1993)
5. Inoue, H., Inaba, M.: Hand-eye coordination in rope handling. In: Robotics Research: The First International Symposium, pp. 163–174. MIT Press, Cambridge (1984)
6. Inaba, M., Inoue, H.: Vision-based robot programming. In: Robotics Research; 5th International Symposium on Robotics Research (ISRR5), pp. 129–136 (1989)
7. Kuniyoshi, Y., Inaba, M., Inoue, H.: Learning by watching: Extracting reusable task knowledge from visual observation of human performance. IEEE Transactions on Robotics and Automation 10(6), 799–822 (1994)
8. Matsui, T., Inaba, M.: EusLisp: An Object-Based Implementation of Lisp. Journal of Information Processing 13(3), 327–338 (1990)
9. Inaba, M., Kagami, S., Kanehiro, F., Nagasaka, K., Inoue, H.: Mother operations to evolve embodied robots based on the remote-brained approach, pp. 319–326 (1997)
10. Inaba, M.: Developmental processes in remote-brained humanoids. In: Shirai, Y., Hirose, S. (eds.) Robotics Research: The Eighth International Symposium, pp. 344–355 (1998)
11. Inaba, M., Kagami, S., Kanehiro, F., Hoshino, Y., Inoue, H.: A platform for robotics research based on the remote-brained robot approach. The International Journal of Robotics Research 19(10), 933–954 (2000)
12. Inaba, M.: Remote-brained robotics: Interfacing ai with real world behaviors. In: Robotics Research: The Sixth International Symposium, pp. 335–344 (1993)
13. Inaba, M., Kagami, S., Ishikawa, T., Kanehiro, F., Takeda, K., Inoue, H.: Vision-based adaptive and interactive behaviors in mechanical animals using the remote-brained approach. Robotics and Autonomous Systems 17(1-2), 35–52 (1996)
14. Inaba, M., Kanehiro, F., Kagami, S., Inoue, H.: Remote-brained ape-like robot to study full-body mobile behaviors based on simulated models and real-time vision. Advanced Robotics 11(6), 653–668 (1997)
15. Sato, T., Inoue, H.: Humanoid autonomous system.RWC Technical Report, pp. 109–110 (1994)
16. Kagami, S., Nishiwaki, K., Kuffner, J.J., Kuniyoshi, Y., Inaba, M., Inoue, H.: Design and implementation of software research platform for humanoid robotics: H7. In: Proceedings of the 2001 IEEE-RAS International Conference on Humanoid Robots, pp. 253–258 (2001)
17. Nishiwaki, K., Kagami, S., Kuffner, J.J., Okada, K., Kuniyoshi, Y., Inaba, M., Inoue, H.: Online humanoid locomotion control using 3d vision information. In: Proceedings of 8th International Symposium on Experimental Robotics, ISER 2002 (2002)
18. Kagami, S., Kuffner, J.J., Nishiwaki, K., Kuniyoshi, Y., Okada, K., Inaba, M., Inoue, H.: Humanoid arm motion planning using stereo vision and rrt search. Journal of Robotics and Mechatronics 15(2), 200–207 (2003)
19. Okada, K., Inaba, M., Inoue, H.: Integration of Real-time Binocular Stereo Vision and Whole Body Information for Dynamic Walking Navigation of Humanoid Robot. In: Proceedings of International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI 2003), pp. 131–136 (2003)
20. Inoue, H., Tachi, S., Tanie, K., Yokoi, K., Hirai, S., Hirukawa, H., Hirai, K., Nakayama, S., Sawada, K., Nishiyama, T., Miki, O., Itoko, T., Inaba, H., Sudo, M.: HRP: Humaonid Robotics Project of MITI. In: Proceedings of the First IEEE-RAS International Conference on Humanoid Robots, Humanoids 2000 (2000)

21. Okada, K., Kojima, M., Sagawa, Y., Ichino, T., Sato, K., Inaba, M.: Vision based behavior verification system of humanoid robot for daily environment tasks. In: 2006 6th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2006), pp. 7–12 (December 2006)
22. Okada, K., Ogura, T., Haneda, A., Kousaka, D., Nakai, H., Inaba, M., Inoue, H.: Integrated System Software for HRP2 Humanoid. In: Proceedings of The 2004 IEEE International Conference on Robotics and Automation (April 2004)
23. Mizuuchi, I., Yoshida, S., Inaba, M., Inoue, H.: The development and control of the flexible-spine of a human-form robot. Advanced Robotics 17(2), 179–196 (2003)
24. Inaba, M., Mizuuchi, I., Tajima, R., Yoshikai, T., Sato, D., Nagashima, K., Inoue, H.: Building spined mustle-tendon humanoid, pp. 113–127 (2003)
25. Mizuuchi, I., Tajima, R., Yoshikai, T., Sato, D., Nagashima, K., Inaba, M., Kuniyoshi, Y., Inoue, H.: The Design and Control of the Flexible Spine of a Fully Tendon-Driven Humanoid "Kenta". In: Proceedings of the 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 2527–2532 (October 2002)
26. Mizuuchi, I., Yoshikai, T., Sodeyama, Y., Nakanishi, Y., Miyadera, A., Yamamoto, T., Niemel, T., Hayashi, M., Urata, J., Namiki, Y., Nishino, T., Inaba, M.: Development of musculoskeletal humanoid kotaro. In: Proceedings of The 2006 IEEE International Conference on Robotics and Automation, pp. 82–87 (May 2006)
27. Mizuuchi, I., Nakanishi, Y., Sodeyama, Y., Namiki, Y., Nishino, T., Muramatsu, N., Urata, J., Hongo, K., Yoshikai, T., Inaba, M.: An advanced musculoskeletal humanoid kojiro. In: Proceedings of the 2007 IEEE-RAS International Conference on Humanoid Robots, Humanoids 2007 (December 2007)
28. Inaba, M.: Extended vision with robot sensor suit: Primary sensor image approach in interfacing body to brain. In: Giralt, G., Hirzinger, G. (eds.) Robotics Research: The Seventh International Symposium, pp. 499–508 (1996)
29. Hayashi, M., Sagisaka, T., Ishizaka, Y., Yoshikai, T., Inaba, M.: Development of functional whole-body flesh with distributed three-axis force sensors to enable close interaction by humanoids. In: Proceedings of The 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3610–3615 (October 2007)
30. Yoshikai, T., Hayashi, M., Kadowaki, A., Goto, T., Inaba, M.: Design and development of a humanoid with soft 3d-deformable sensor flesh and automatic recoverable mechanical overload protection mechanism. In: Proceedings of IEEE/RSJ 2009 International Conference on Intelligent Robots and Systems, pp. 4977–4983 (2009)
31. Urata, J., Hirose, T., Yuta, N., Nakanishi, Y., Mizuuchi, I., Inaba, M.: Thermal control of electrical motors for high-power humanoid robots. In: Proceedings of The 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems, September 2008, pp. 2047–2052 (2008)
32. Matsui, T.: Multithread Object-Oriented Language EusLisp for Parallel and Asynchronous Programming in Robotics. In: Workshop on Concurrent Object-based Systems, IEEE 6th Symposium on Parallel and Distributed Processing (1994)
33. Hirukawa, H., Kanehiro, F., Kaneko, K., Kajita, S., Fujiwara, K., Kawai, Y., Tomita, F., Hirai, S., Tanie, K., Isozumi, T., Akechi, K., Kawasaki, T., Ota, S., Yokoyama, K., Honda, H., Fukase, Y., Maeda, J., Nakamura, Y., Tachi, S., Inoue, H.: Humanoid Robotics Platforms developed in HRP. In: Proceedings of the 2003 IEEE-RAS International Conference on Humanoid Robots, Humanoids 2003 (2003)
34. Kanehiro, F., Fujiwara, K., Kajita, S., Yokoi, K., Kaneko, K., Hirukawa, H.: Open Architecture Humanoid Robot Platform. In: Proceedings of the 2002 IEEE International Conference on Robotics and Automation (ICRA 2002), pp. 24–30 (2002)
35. Bradski, G., Kaehler, A.: Learning opencv (2008)

36. Larsen, E., Gottschalk, S., Lin, M.C., Manocha, D.: Fast proximity queries with swept sphere volumes. In: Proceedings of The 2000 IEEE International Conference on Robotics and Automation, pp. 3719–3726 (2000)
37. Open Dynamics Engine ODE, http://ode.com
38. PhysX, http://www.nvidia.com/page/home.html
39. The MathWorks, http://www.mathworks.com/products/featured/embeddedmatlab/
40. Yuasa, T.: Real-time garbage collection on general-purpose machines 11(3), 181–198 (1990)
41. Printezis, T., Detlefs, D.: A generational mostly-concurrent garbage collector. In: ISMM 2000: Proceedings of the 2nd international symposium on Memory management, pp. 143–154. ACM Press, New York (2000)
42. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T.B., Leibs, J., Wheeler, R., Ng, A.Y.: Ros: an open-source robot operating system. In: International Conference on Robotics and Automation. Open-Source Software workshop (2009)
43. Jackson, J.: Microsoft robotics studio: A technical introduction. In: IEEE Robotics and Automation Magazine (2007)
44. Ando, N., Suehiro, T., Kitagaki, K., Kotoku, T., Yoon, W.: Rt-middleware: Distributed component middleware for rt (robot technology). In: Proceedings of IEEE International Conference on Intelligent Robots and Systems, pp. 3555–3560 (2005)
45. Yamazaki, K., Ueda, R., Nozawa, S., Mori, Y., Maki, T., Hatao, N., Okada, K., Inaba, M.: A demonstrative research for daily assistive robots on tasks of cleaning and tidying up rooms. In: First International Symposium on Quality of Life Technology, pp.J (June 2009)
46. Yamazaki, K., Inaba, M.: A cloth detection method based on wrinkle features for daily assistive robots. In: IAPR International Conference on Machine Vision and Applications, May 2009, pp. 366–369 (2009)
47. Mizuuchi, I., Fujimoto, J., Yamamoto, K., Sodeyama, Y., Muramatsu, N., Ohta, S., Hongo, K., Hirose, T., Inaba, M.: A kitchen assistant robot with a variety of sensors embedded in the hand to clear away dishes. In: First International Symposium on Quality of Life Technology, pp.K (June 2009)
48. Fujimoto, J., Mizuuchi, I., Sodeyama, Y., Yamamoto, K., Muramatsu, N., Ohta, S., Hirose, T., Hongo, K., Okada, K., Inaba, M.: Picking up dishes based on active groping with multisensory robot hand. In: 18th IEEE International Symposium on Robot and Human Interactive Communication, September 2009, pp. 220–225 (2009)
49. Tanaka, Y., Nakai, A., Iwase, E., Goto, T., Matsumoto, K., Shimoyama, I.: Triaxial tactile sensor chips with piezoresistive cantilevers mountable on curved surface. In: Proceedings of Asia-Pacific Conference on Transducers and Micro-Nano Technology 2008 (APCOT2008), pp.1B1–1 (2008)
50. Nakao, M., Sawasaki, N.: Development of image recognition module with linux for common basis for next-generation robots and rt components for image recognition. In: The 27th Annual Conference of the Robotics Society of Japan (2009)
51. Inoue, H., Tachikawa, T., Inaba, M.: Robot vision system with a correlation chip for realtime tracking, optical flow and depth map generation. In: Proceedings of the 1992 IEEE International Conference on Robotics and Automation, pp. 1621–1626 (1992)
52. Kojima, M., Shirayama, S., Ueki, R., Okada, K., Inaba, M.: Recognition and recording of human access to a shelf by a care assit robot with cameras with multi varaible views. In: The 27th Annual Conference of the Robotics Society of Japan

Learning

Cédric Pradalier

On the path to increased autonomy and performance, robotic systems must become more and more able to adapt to a changing environment and to learn from their experience. In some cases, learning could be the best way to acquire new behaviours, i.e. learning from the demonstration of a human teacher.

- In “Learning Visual Representations for Interactive Systems”, Piater et al. use a reinforcement learning approach to learn how to grasp objects of diverse shapes. An important aspect of this type of learning is the absence of supervision: the robot tries by itself to grasp objects and move them around, and, from this experience, learns what the best grasping configurations are.
- In “Learning Mobile Robot Motion Control from Demonstrated Primitives and Human Feedback”, Argall et al. address the difficult question of teaching motion primitives to a robotic system based on a combination of demonstration and corrective feedback while the robot is trying to instantiate the motion primitives. Interestingly, their approach allows assembling complex behaviours based on previously learnt primitive behaviours.
- In “Perceptual Interpretation for Autonomous Navigation through Dynamic Imitation Learning”, David et al. take advantage of human demonstrations to learn the characteristics of traversable terrains. An interesting aspect of this work is the comparison with human-engineered solutions to the same problem, and the conclusion that learning-based techniques achieve equivalent performances with much less tweaking.
- In “An inverse optimal control approach to human motion modelling”, Mombaur et al. aim to learn the principle of human motion planning. By observing human motions, they are using a learning approach to reverse-engineer the rules of optimal control that humans are using, assuming that human motion follows some kind of optimality criterion.
- In “Towards Motor Skill Learning for Robotics”, Peters et al. focus on learning sensory-motor skills, breaking down the problem into learning primitive skills, and then learning to combine these skills into complex behaviours. As for Argall et al., the approach here is based on imitation learning with a human teacher providing a first model of the task to be executed.

- In “Learning Landmark Selection Policies for Mapping Unknown Environments”, Strasdat et al. aim at reducing the computational burden of mapping unknown environments by learning to identify useful landmarks only, discarding the others. Integrated in a SLAM framework, this approach uses a reinforcement learning technique and a policy compression based on neural network, to achieve performance superior to hand-crafted landmark selection heuristics.

In this group of papers, learning from demonstrations is a recurring topic. The need to rely on demonstrations is linked to one of the core challenges of learning systems applied to robotics: the dimension of the parameter space to be explored is huge in all but the most trivial problems. Introducing a human input helps reducing the search to a channel of parameters around the demonstrated trajectory.

Learning Visual Representations for Interactive Systems

Justus Piater, Sébastien Jodogne, Renaud Detry, Dirk Kraft, Norbert Krüger, Oliver Krömer, and Jan Peters

Abstract. We describe two quite different methods for associating action parameters to visual percepts. Our *RLVC* algorithm performs reinforcement learning directly on the visual input space. To make this very large space manageable, RLVC interleaves the reinforcement learner with a supervised classification algorithm that seeks to split perceptual states so as to reduce perceptual aliasing. This results in an adaptive discretization of the perceptual space based on the presence or absence of visual features. Its extension RLJC also handles continuous action spaces. In contrast to the minimalistic visual representations produced by RLVC and RLJC, our second method learns structural object models for robust object detection and pose estimation by probabilistic inference. To these models, the method associates grasp experiences autonomously learned by trial and error. These experiences form a non-parametric representation of grasp success likelihoods over gripper poses, which we call a *grasp density*. Thus, object detection in a novel scene simultaneously produces suitable grasping options.

1 Introduction

Vision is a popular sensory modality for autonomous robots. Optical sensors are comparatively cheap and deliver more information at higher rates than other sensors. Moreover, vision appears intuitive as humans heavily rely on it. We appear to

Justus Piater · Sébastien Jodogne (now at Euresys s.a., Belgium) · Renaud Detry
INTELSIG Laboratory, Université de Liège, Belgium
e-mail: Justus.Piater@ULg.ac.be, Renaud.Detry@ULg.ac.be

Dirk Kraft · Norbert Krüger
University of Southern Denmark
e-mail: kraft@mip.sdu.dk, norbert@mip.sdu.dk

Oliver Kroemer · Jan Peters
Max Planck Institute for Biological Cybernetics, Tübingen, Germany
e-mail: oliverkro@tuebingen.mpg.de, Jan.Peters@tuebingen.mpg.de

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 399–416.
springerlink.com

think and act on the basis of an internal model of our environment that is constantly updated via sensory – and mostly visual – perception. It is therefore a natural idea to attempt to build autonomous robots that derive actions by reasoning about an internal representation of the world, created by computer vision.

However, experience shows that it is very difficult to build generic world models by sensory perception. For example, detailed shape recovery from passive optical sensors is hard, and the appearance of a scene – that is, its raw image representation – varies about as greatly with imaging conditions as it does with semantic content.

Paraphrasing Vapnik's Main Principle of inference from a small sample size [22], it may thus not be a good idea to try to solve the hard intermediate problem of building a reliable world model in order to solve the easier problem of extracting just the information the robot needs to act appropriately. This leads to the idea of linking perception more-or-less directly to action, without the intermediate step of reasoning on a world model, whose roots go back at least to the learned value functions of Samuel's famous checker player [19].

In this chapter, we give an overview of two examples of our own research on learning visual representations for robotic action within specific task scenarios, without building generic world models. The first problem we consider is the direct linking of visual perception to action within a reinforcement-learning (RL) framework (Sect. 2). The principal difficulty is the extreme size of the visual perceptual space, which we address by learning a percept classifier interleaved with the reinforcement learner to adaptively discretize the perceptual space into a manageable number of discrete states. However, not all visuomotor tasks can be reduced to simple, reactive decisions based on discrete perceptual states. For example, to grasp objects, the object pose is of fundamental importance, and the grasp parameters depend on it in a continuous fashion. We address such tasks by learning intermediate object representations that form a direct link between perceptual and action parameters (Sect. 3). Again, these representations can be learned autonomously, permitting the robot to improve its grasping skills with experience. The resulting probabilistic models allow the inference of possible grasps and their relative success likelihoods from visual scenes.

2 Reinforcement Learning of Visual Classes

Reinforcement learning [2, 21] is a popular method for learning perception-action mappings, so-called *policies*, within the framework of Markov Decision Processes (MDP). Learning takes place by evaluating *actions* taken in specific *states* in terms of the *reward* received. This is conceptually easy for problems with discrete state and action spaces. Continuous and very large, discrete state spaces are typically addressed by using function approximators that permit local generalization across similar states. However, for reasons already noted above, function approximators alone are not adequate for the very high-dimensional state spaces spanned by images: Visually similar images may represent states that require distinct actions, and very dissimilar images may actually represent the exact same scene (and thus state) under different imaging conditions. Needless to say, performing RL directly on the combinatorial state space defined by the image pixel values is infeasible.

The challenge therefore lies in mapping the visual space to a state representation that is suitable for reinforcement learning. To enable learning with manageably low numbers of exploratory actions, this means that the state space should either consist of a relatively small number of discrete states, or should be relatively low-dimensional and structured in such a way that nearby points in state space mostly admit identical actions that yield similar rewards.

The latter approach would be very interesting to explore, but it appears that it would require strong, high-level knowledge about the content of the scene such as object localization and recognition, which defeats our purpose of learning perception-action mappings without solving this harder problem first. We therefore followed the first approach and introduced a method called Reinforcement Learning of Visual Classes (RLVC) that adaptively and incrementally discretizes a continuous or very large discrete perceptual space into discrete states [8, 11].

2.1 RLVC: A Birdseye View

RLVC decomposes the end-to-end problem of learning perception-to-action mappings into two simpler learning processes (Fig. 1). One of these, the *RL agent*, learns discrete state-action mappings in a classical RL manner. The state representation and the determination of the current state are provided by an *image classifier* that carves up the perceptual (image) space into discrete states called *visual classes*. These two processes are interleaved: Initially, the entire perceptual space is mapped to a single visual class. From the point of view of the RL agent, this means that a variety of distinct world states requiring different actions are lumped together – *aliased* – into a single perceptual state (visual class). Based on experience accumulated by the RL agent, the image classifier then identifies a visual feature whose presence or absence defines two distinct visual subclasses, splitting the original visual class into two. This procedure is iterated: At each iteration, one or more perceptually-aliased visual classes are identified, and for each, a feature is determined that splits it in a way that maximally reduces the perceptual aliasing in both of the resulting new visual classes (Fig. 2). Thus, in a sequence of attempts to reduce perceptual aliasing, RLVC builds a sequence $\mathscr{C}_0, \mathscr{C}_1, \mathscr{C}_2, \ldots$ of increasingly refined, binary decision trees $\mathscr{C}_k$ with visual feature detectors at decision nodes. At any stage k, $\mathscr{C}_k$ partitions the visual space S into a finite number m_k of visual classes $\{V_{k,1}, \ldots, V_{k,m_k}\}$.

2.2 Reinforcement Learning and TD Errors

An MDP is a quadruple $\langle S, A, \mathscr{T}, \mathscr{R} \rangle$, where S is a finite set of states, A is a finite set of actions, $\mathscr{T}$ is a probabilistic transition function from $S \times A$ to S, and $\mathscr{R}$ is a scalar reward function defined on $S \times A$. From state s_t at time t, the agent takes an action a_t, receives a scalar reinforcement $r_{t+1} = \mathscr{R}(s_t, a_t)$, and transitions to state s_{t+1} with probability $\mathscr{T}(s_t, a_t, s_{t+1})$. The corresponding quadruple $\langle s_t, a_t, r_{t+1}, s_{t+1} \rangle$ is called

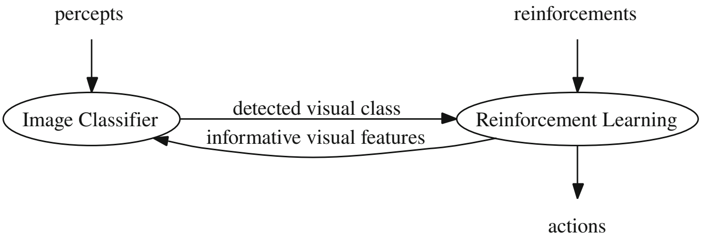


Fig. 1 RLVC: Learning a perception-action mapping decomposed into two interacting subproblems.

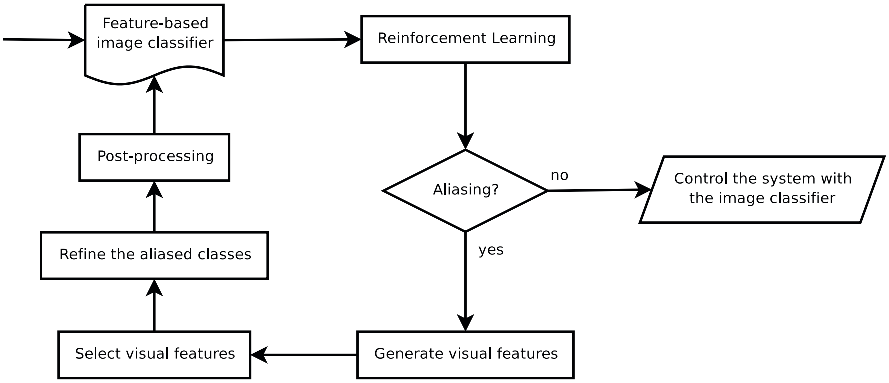

Feature-based image classifier
Reinforcement Learning
Post-processing
Aliasing?
no
Control the system with the image classifier
Refine the aliased classes
yes
Select visual features
Generate visual features

Fig. 2 Outline of the RLVC algorithm.

an *interaction*. For infinite-horizon MDPs, the objective is to find an optimal *policy* $\pi^* : S \longrightarrow A$ that chooses actions that maximize the expected *discounted return*

$$R_t = \sum_{i=0}^{\infty} \gamma^i r_{t+i+1} \tag{1}$$

for any starting state s_0, where $0 \leq \gamma < 1$ is a discount factor that specifies the immediate value of future reinforcements.

If $\mathscr{T}$ and $\mathscr{R}$ are known, the MDP can be solved by dynamic programming [1]. Reinforcement learning can be seen as a class of methods for solving unknown MDPs. One popular such method is Q-learning [23], named after its state-action value function

$$Q^\pi(s,a) = \mathrm{E}^\pi\left[R_t \mid s_t = s, a_t = a\right] \tag{2}$$

that returns the expected discounted return by starting from state s, taking action a, and following the policy π thereafter. An optimal solution to the MDP is then given by

$$\pi^*(s) = \operatorname*{argmax}_{a \in A} Q^*(s,a). \tag{3}$$

In principle, a Q function can be learned by a sequence of α-weighted updates

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\left(\mathrm{E}[R_t \mid s = s_t, a = a_t] - Q(s_t,a_t)\right) \quad (4)$$

that visits all state-action pairs infinitely often. Of course, this is not a viable algorithm because the first term of the update step is unknown; it is precisely the return function (2) we want to learn. Now, rewards are accumulated (1) by executing actions, hopping from state to state. Thus, for an interaction $\langle s_t, a_t, r_{t+1}, s_{t+1}\rangle$, an estimate of the current return $Q(s_t,a_t)$ is available as the discounted sum of the immediate reward r_{t+1} and the estimate of the remaining return $Q(s_{t+1},a_{t+1})$, where $s_{t+1} = \mathscr{T}(s_t,a_t)$ and $a_{t+1} = \pi(s_t)$. If the goal is to learn a value function Q^* for an optimal policy (3), then this leads to the algorithm

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\Delta_t \quad (5)$$

$$\Delta_t = r_{t+1} + \gamma \max_{a' \in A} Q(s_{t+1},a') - Q(s_t,a_t) \quad (6)$$

that, under suitable conditions, converges to Q^*. Δ_t is called the *temporal-difference error* or *TD error* for short.

2.3 Removing Perceptual Aliasing

RLVC is based on the insight that if the world behaves predictably, $r_{t+1} + \gamma \max_{a' \in A} Q(s_{t+1},a')$ approaches $Q(s_t,a_t)$, leading to vanishing TD errors (6). If however the magnitudes of the TD errors of a given state-action pair (s,a) remain large, this state-action pair yields unpredictable returns. RLVC assumes that this is due to perceptual aliasing, that is, the visual class s represents distinct world states that require different actions. Thus, it seeks to split this state in a way that minimizes the sum of the variances of the TD errors in each of the two new states. This is an adaptation of the splitting rule used by CART for building regression trees [3].

To this end, RLVC selects from all interactions collected from experience those whose visual class and action match s and a, respectively, along with the resulting TD error Δ, as well as the set $F_\oplus \in F$ of features present in the raw image from which the visual class was computed. It then selects the feature

$$f^* = \operatorname*{argmin}_{f \in F} \left\{ p_f \sigma^2\{\Delta_f\} + p_{\neg f} \sigma^2\{\Delta_{\neg f}\} \right\} \quad (7)$$

that results in the purest split in terms of the TD errors. Here, p_f is the proportion of the selected interactions whose images exhibit feature f, and $\{\Delta_f\}$ is the associated set of TD errors; $\neg f$ indicates the corresponding entities that do not exhibit feature f.

Splitting a visual class s according to the presence of a feature f^* results in two new visual classes, at least one of which will generally exhibit lower TD errors than the original s. However, there is the possibility that such a split turns out to be useless because the observed lack of convergence was due to the stochastic nature

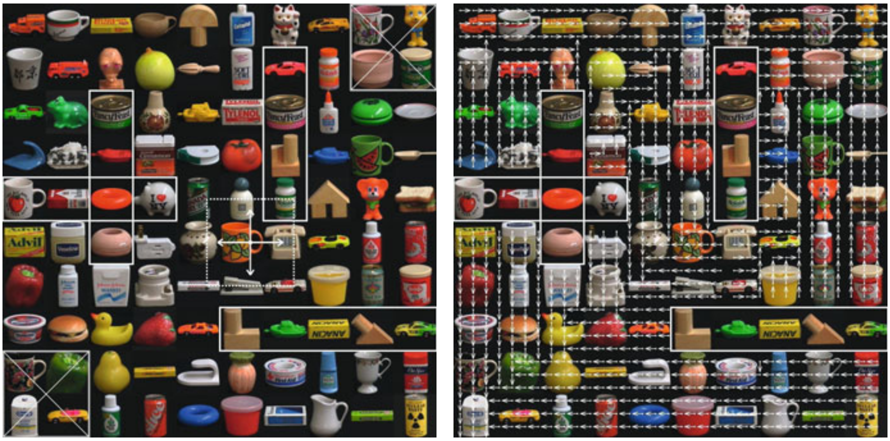


Fig. 3 A continuous, noisy navigation task. The exits of the maze are marked by crossed boxes. Transparent obstacles are identified by solid rectangles. The agent is depicted near the center of the left-hand figure. Each of the four possible moves is represented by an arrow, the length of which corresponds to the resulting move. The sensor returns a picture that corresponds to the dashed portion of the image. The right-hand figure shows an optimal policy learned by RLVC, sampled at regularly-spaced points.

of the environment rather than perceptual aliasing. RLVC partially addresses this by splitting a state only if the resulting distributions of TD errors are significantly different according to a Student's t test.

2.4 Experiments

We evaluated our system on an abstract task that closely parallels a real-world, reactive navigation scenario (Fig. 3). The goal of the agent is to reach one of the two exits of the maze as fast as possible. The set of possible locations is continuous. At each location, the agent has four possible actions: Go up, right, down, or left. Every move is altered by Gaussian noise, the standard deviation of which is 2% of the size of the maze. Invisible obstacles are present in the maze. Whenever a move would take the agent into an obstacle or outside the maze, its location is not changed.

The agent earns a reward of 100 when an exit is reached. Any other move generates zero reinforcement. When the agent succeeds at escaping the maze, it arrives in a terminal state in which every move gives rise to a zero reinforcement. The discount factor γ was set to 0.9. Note that the agent is faced with the delayed-reward problem, and that it must take the distance to the two exits into consideration when choosing the most attractive exit.

The raw perceptual input of the agent is a square window centered at its current location, showing a subset of a tiled montage of the COIL-100 images [15]. There is no way for the agent to directly locate the obstacles; it is obliged to identify them implicitly as regions of the maze where certain actions do not change its location.

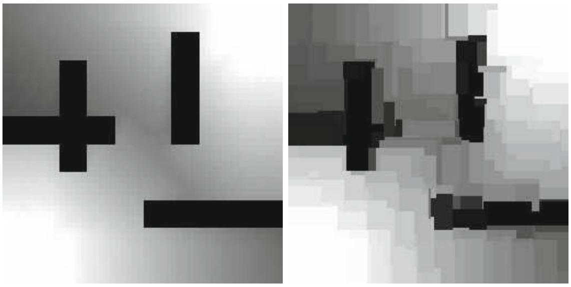


Fig. 4 Left: The optimal value function, when the agent has direct access to its (x,y) position in the maze and when the set of possible locations is discretized into a 50×50 grid. The brighter the location, the greater its value. Right: The final value function obtained by RLVC.

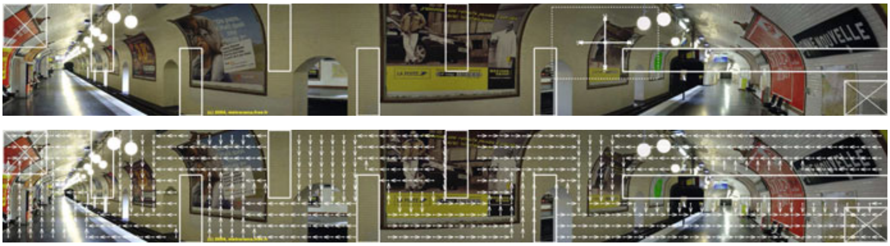


Fig. 5 Top: a navigation task with a real-world image, using the same conventions as Figure 3. Bottom: the deterministic image-to-action mapping computed by RLVC.

In this experiment, we used color differential invariants as visual features [7]. The entire tapestry includes 2298 different visual features, of which RLVC selected 200 (9%). The computation stopped after the generation of $k = 84$ image classifiers, which took 35 minutes on a 2.4 GHz Pentium IV using databases of 10,000 interactions. 205 visual classes were identified. This is a small number compared to the number of perceptual classes that would be generated by a discretization of the maze when the agent knows its (x,y) position. For example, a reasonably-sized 20×20 grid leads to 400 perceptual classes. A direct, tabular representation of the Q function in terms of all Boolean feature combinations would have $2^{2298} \times 4$ cells. Figure 4 compares the optimal value function of a regularly-discretized problem with the one obtained through RLVC.

In a second experiment we investigated RLVC on real-word images under identical navigation rules (Fig. 5). RLVC took 101 iterations in 159 minutes to converge using databases of 10,000 interactions. 144 distinct visual features were selected among a set of 3739 possibilities, generating a set of 149 visual classes. Here again, the resulting classifier is fine enough to obtain a nearly optimal image-to-action mapping for the task.

2.5 Further Developments

The basic method described in the preceding sections admits various powerful extensions. First, as described above, the power of RLVC to resolve action-relevant perceptual ambiguities is limited by the availability of precomputed visual features and their discriminative power. This can be overcome by creating new features on the fly as needed [12]. When the state-refinement procedure fails to identify a feature that results in a significant reduction of the TD errors, new features are created by forming spatial compounds of existing features. In this way, a compositional hierarchy of features is created in a task-driven way. Compounds are always at least as selective as their individual constituents.

A second major improvement results from the observation that RLVC, as described above, is susceptible to overfitting because states are only ever split and never merged. It is therefore desirable to identify and merge equivalent states. Here, we say that two states are equivalent if (a) their optimal values $\max_a Q(s,a)$ are similar, and (b) their optimal policies are equivalent, that is, the value of one state's optimal action $\pi^*(s)$ is similar if taken in the other state. A second drawback of basic RLVC is that decision trees do not make optimal use of the available features, since they can only represent conjunctions of features. To address both issues, we modified RLVC to use a Binary Decision Diagram (BDD) [4] instead of a decision tree to represent the state space [9]. To split a state, a new feature is conjunctively added to the BDD as before to the decision tree. Periodically, after running for some number of stages, *compaction* takes place: equivalent states are merged and a new BDD is formed that can represent both conjunctions and disjunctions of features. In the process, feature tests are reordered as appropriate, which may lead to the elimination of some features. We demonstrated that this can result in a drastic reduction in the number of visual features and classes learned, while improving generalization at the same time.

Thirdly, we generalized the concept of visual classes to joint perception-action classes. Our algorithm Reinforcement Learning of Joint Classes (RLJC) applies the principles of RLVC to an adaptive discretization of the joint space of perceptions and actions [10]. The Q function now operates on the joint-class domain encompassing both perceptual and action dimensions. Joint classes are split based on *joint features* that test the presence of either a visual feature or an *action feature*. An action feature (t,i) tests whether the ith component of an action $a \in \mathbb{R}^m$ falls below a threshold t. This relatively straightforward generalization of RLVC results in an innovative addition to the rather sparse toolbox of RL methods for continuous action spaces.

3 Grasp Densities

RLVC, described in the preceding section, follows a minimalist approach to perception-action learning in that it seeks to identify small sets of low-level visual cues and to associate reactive actions to them directly. There is no elaborate image analysis beyond feature extraction, no intermediate representation, no reasoning or planning, and the complexity of the action space that can be handled by RL is limited. In this section we describe a different approach to perception-action learning that is in many ways complementary to RLVC. Its visual front-end builds elaborate representations from which powerful, structured object representations are learned, and multidimensional action vectors are derived via probabilistic inference.

We describe this method in the context of grasping objects [5], a fundamental skill of autonomous agents. The conventional robotic approach to grasping involves computing grasp parameters based on detailed geometric and physical models of the object and the manipulator. However, humans skillfully manipulate everyday objects even though they do not have access to such detailed information. It is thus clear that there must exist alternative methods. We postulate that manipulation skills emerge with experience by associating action parameters to perceptual cues. Then, perceptual cues can directly trigger appropriate actions, without explicit object shape analysis or grasp planning. For example, to drink, we do not have to reason about the shape and size of the handle of a hot teacup to determine where to place the fingers to pick it up. Rather, having successfully picked up and drunk from teacups before, seeing the characteristic handle immediately triggers the associated, canonical grasp.

Our method achieves such behavior by first learning visual object models that allow the agent to detect object instances in scenes and to determine their pose. Then, the agent explores various grasps, and associates the successful parameters that emerge from this grasping experience with the model. When the object is later detected in a scene, the detection and pose estimation procedure immediately produces the associated grasp parameters as well. This system can be bootstrapped in the sense that very little grasping experience is already useful, and the representation can be refined by further experience at any time. It thus constitutes a mechanism for learning grasping skills from experience with familiar objects.

3.1 Learning to Grasp: A Birdseye View

Figure 6 presents an overview of our grasp learning system. Visual input is provided by a computer vision front-end that produces 3D oriented patches with appearance information. On such sparse 3D reconstructions, the object-learning component analyzes spatial feature relations. Pairs of features are combined by new parent features, producing a hierarchically-structured Markov network that represents the object via sparse appearance and structure. In this network, a link from a parent node to a child node represents the distribution of spatial relations (relative pose) between the parent node and instances of the child node. Leaf nodes encode appearance information.

Given a reconstruction of a new scene provided by computer vision, instances of such an object model can be detected via probabilistic inference, estimating their pose in the process.

Having detected a known object for which it has no grasping experience yet, the robot attempts to grasp the object in various ways. For each grasp attempt, it stores

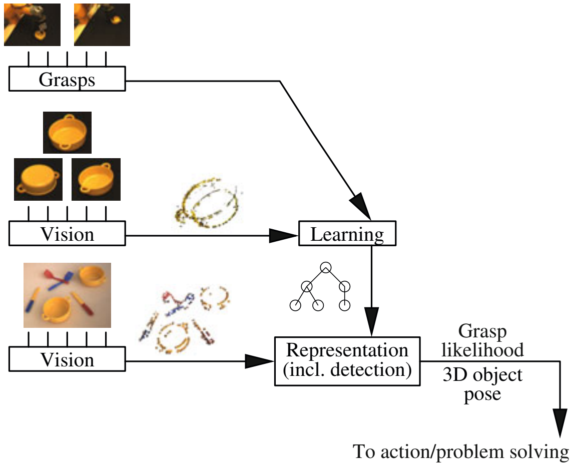


Fig. 6 Overview of learning to grasp. Learning produces graph-structured object representations that combine experienced grasps with visual features provided by computer vision. Subsequently, instances can be detected in new scenes provided by vision. Detection directly produces pose estimates and suitable grasp parameters.

the object-relative pose of the gripper and a measure of success. Object-relative gripper poses are represented in exactly the same way as parent-relative child poses of visual features. In this manner, grasping experience is added to the object representation as a new child node of a high-level object node. From then on, inference of object pose at the same time produces a distribution of gripper poses suitable for grasping the object.

3.2 Visual Front-End

Visual primitives and their location and orientation in space are provided by the Early-Cognitive-Vision (ECV) system by Krüger et al. [14, 17]. It extracts patches – so-called ECV descriptors – along image contours and determines their 3D position and a 2D orientation by stereo techniques (the orientation around the 3D contour axis is difficult to define and is left undetermined). From a calibrated stereo pair, it generates a set of ECV descriptors that represent the scene, as sketched at the bottom of Fig. 6.

Objects can be isolated from such scene representations by motion segmentation. To this end, the robot uses bottom-up heuristics to attempt to grasp various surfaces suggested by combinations of ECV descriptors. Once a grasp succeeds and the robot gains physical control over the grasped structure, the robot can pick it up and turn it in front of the stereo camera. This allows it to segment object descriptors from the rest of the scene via coherent motion, and to complete and refine the object descriptors by structure-from-motion techniques, as illustrated in Fig. 7 [18, 13].

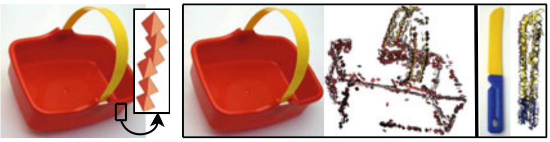


Fig. 7 ECV descriptors. Left: ECV descriptors are oriented appearance patches extracted along contours. Right: object-segmented and refined ECV descriptors via structure from motion.

3.3 Markov Networks for Object Representation

Our object model consists of a set of generic *features* organized in a hierarchy. Features that form the bottom level of the hierarchy, referred to as *primitive features*, are bound to visual observations. The rest of the features are *meta-features* that embody relative spatial configurations of more elementary features, either meta or primitive. At the bottom of the hierarchy, primitive features correspond to local parts that each may have many *instances* in the object. Climbing up the hierarchy, meta-features correspond to increasingly complex parts defined in terms of constellations of lower-level parts. Eventually, parts become complex enough to satisfactorily represent the whole object. Here, a primitive feature represents a class of ECV observations of similar appearance, e.g. an ECV observation with colors close to red and white. Any primitive feature will usually have hundreds of instances in a scene.

Figure 8 shows an example of a hierarchy for a traffic sign. Ignoring the nodes labeled Y_i for now, the figure shows the traffic sign as the combination of two features, a bridge pattern (feature 4) and a triangular frame (feature 5). The fact that the bridge pattern has to be in the center of the triangle to form the traffic sign is encoded in the links between features 4-6-5. The triangular frame is encoded in terms of a single (generic) feature, a short red-white edge segment (feature 3). The link between feature 3 and feature 5 encodes the fact that many short red-white edge segments are necessary to form the triangular frame, and the fact that these edges have to be arranged along a triangle-shaped structure.

Here, a "feature" is an abstract concept that may have any number of instances. The lower-level the feature, the larger generally the number of instances. Conversely, the higher-level the feature, the richer its relational and appearance description.

The feature hierarchy is implemented as a Markov tree (Fig. 8). Features correspond to hidden nodes of the network. When a model is associated to a scene (during learning or instantiation), the pose of feature i in that scene will be represented by the probability density function of a random variable X_i, effectively linking feature i to its instances. Random variables are thus defined over the pose space, which corresponds to the Special Euclidean group $SE(3) = \mathbb{R}^3 \times SO(3)$.

The relationship between a meta-feature i and one of its children j is parametrized by a *compatibility potential* $\psi_{ij}(X_i, X_j)$ that reflects, for any given relative configuration of feature i and feature j, the likelihood of finding these two features in that relative configuration. The (symmetric) potential between i and j is denoted

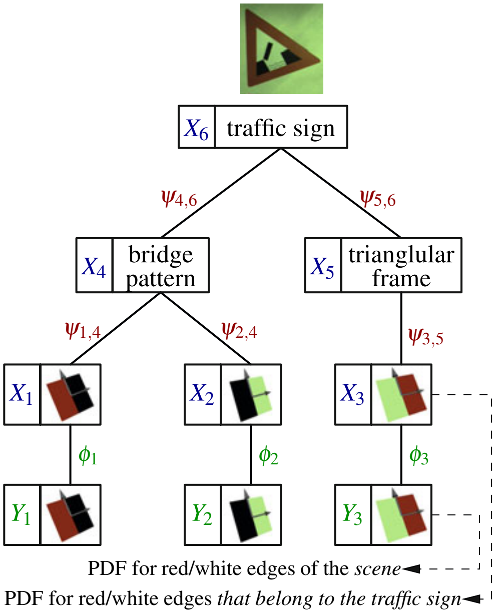


Fig. 8 An example of a hierarchy for a traffic sign. X_1 through X_3 are primitive features; each of these is linked to an observed variable Y_i. X_4 through X_6 are meta-features.

by $\psi_{ij}(X_i,X_j)$. A compatibility potential is equivalent to the spatial distribution of the child feature in a reference frame that matches the pose of the parent feature; a potential can be represented by a probability density over $SE(3)$.

Each primitive feature is linked to an observed variable Y_i. Observed variables are tagged with an appearance descriptor that defines a class of observation appearance. The statistical dependency between a hidden variable X_i and its observed variable Y_i is parametrized by an *observation potential* $\psi_i(X_i,Y_i)$. We generally cannot observe meta-features; their observation potentials are thus uniform.

Instantiation of such a model in a scene amounts to the computation of the marginal posterior pose densities $p(X_i|Y_1,\dots,Y_n)$ for all features X_i, given all available evidence Y. This can be done using any applicable inference mechanism. We use nonparametric belief propagation [20] optimized to exploit the specific structure of this inference problem [6]. The particles used to represent the densities are directly derived from individual feature observations. Thus, object detection (including pose inference) amounts to image observations probabilistically voting for object poses compatible with their own pose. The system never commits to specific feature correspondences, and is thus robust to substantial clutter and occlusions. During inference, a consensus emerges among the available evidence, leading to one or more consistent scene interpretations. After inference, the pose likelihood of the whole object can be read out of the top-level feature. If the scene contains multiple instances of the object, this feature density will present multiple major modes. Figure 9 shows examples of pose estimation results.

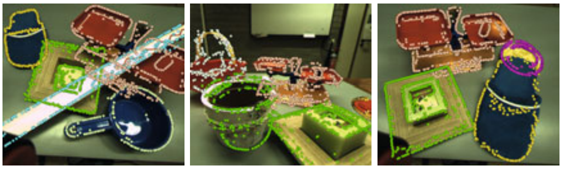


Fig. 9 Cluttered scenes with pose estimates. Local features of object models are back-projected into the image at the estimated pose; false colors identify different objects.

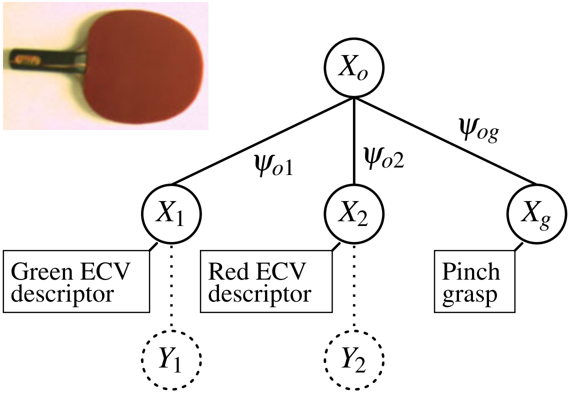


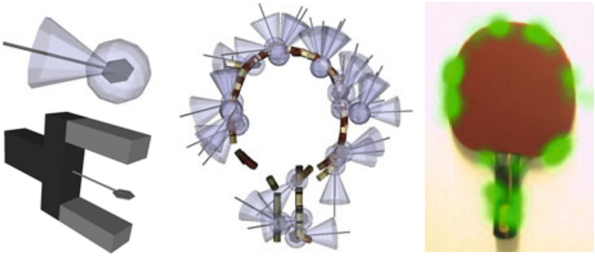
Fig. 10 Visual/affordance model of a table-tennis bat as a 2-level hierarchy. The bat is represented by feature o (top). Feature 1 represents a generic green ECV descriptor. The rectangular configuration of green edges around the handle of the paddle is encoded in ψ_{o1}. Y_1 and Y_2 are observed variables that link features 1 and 2 to the visual evidence produced by ECV. X_g represents a grasp feature, linked to the object feature through the pinch grasp affordance ψ_{og}.

3.4 Grasp Densities

Having described the visual object representation and pose inference mechanism, we now turn to our objective of learning grasp parameters and associating them to the visual object models. We consider parallel-gripper grasps parametrized by a 6D gripper pose composed of a 3D position and a 3D orientation. The set of object-relative gripper poses that yield stable grasps is the *grasp affordance* of the object.

A grasp affordance is represented as a probability density function defined on $SE(3)$ in an object-relative reference frame. We store an expression of the joint distribution $p(X_o, X_g)$, where X_o is the pose distribution of the object, and X_g is the grasp affordance. This is done by adding a new "grasp" feature to the Markov network, and linking it to the top feature (see Fig. 10). The statistical dependency of X_o and X_g is held in a compatibility potential $\psi_{og}(X_o, X_g)$.

When an object model has been aligned to an object instance (i.e. when the marginal posterior of the top feature has been computed from visually-grounded bottom-up inference), the grasp affordance $p(X_g \mid Y)$ of the object *instance*, given



(a) (b) (c)

Fig. 11 Grasp density representation. The top image of Fig. (a) illustrates a particle from a nonparametric grasp density and its associated kernel widths: the translucent sphere shows one position standard deviation, the cone shows the variance in orientation. The bottom image illustrates how the schematic rendering used in the top image relates to a physical gripper. Figure (b) shows a 3D rendering of the kernels supporting a grasp density for a table-tennis paddle (for clarity, only 30 kernels are rendered). Figure (c) indicates with a green mask of varying opacity the values of the location component of the same grasp density along the plane of the paddle.

all observed evidence Y, is computed through top-down belief propagation, by sending a message from X_o to X_g through $\psi_{og}(X_o, X_g)$:

$$p(X_g \mid Y) = \int \psi_{og}(X_o, X_g) p(X_o \mid Y) \, \mathrm{d}X_o \tag{8}$$

This continuous, probabilistic representation of a grasp affordance in the world frame we call a *grasp density*. In the absence of any other information such as priors over poses or kinematic limitations, it represents the relative likelihood that a given gripper pose will result in a successful grasp.

3.5 Learning Grasp Densities

Like the pose densities discussed in Sect. 3.3, grasp densities are represented nonparametrically in terms of individual observations (Fig. 11). In this case, each successful grasp experience contributes one particle to the nonparametric representation. An unbiased characterization of an object's grasp affordance conceptually involves drawing grasp parameters from a uniform density, executing the associated grasps, and recording the successful grasp parameters.

In reality, executing grasps drawn from a 6D uniform density is not practical, as the chances of stumbling upon successful grasps would be unacceptably low. Instead, we draw grasps from a highly biased *grasp hypothesis density* and use importance sampling techniques to properly weight the grasp outcomes. The result we call a *grasp empirical density*, a term that also communicates the fact that the density is generally only an approximation to the true grasp affordance: The number of

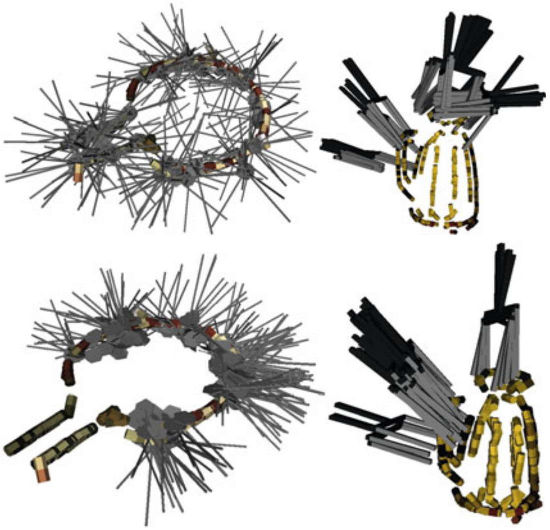


Fig. 12 Particles supporting grasp hypothesis (top) and empirical (bottom) densities. Hypothesis densities were derived from constellations of ECV observations (left) or from human demonstrations (right).

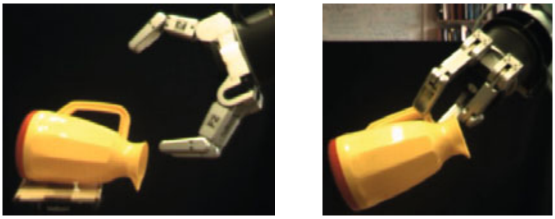


Fig. 13 Barrett hand grasping the toy jug.

particles derived from actual grasp experiences is severely limited – to a few hundred at best – by the fact that each particle is derived from a grasp executed by a real robot. This is not generally sufficient for importance sampling to undo the substantial bias exerted by the available hypothesis densities, which may even be zero in regions that afford actual grasps.

Grasp hypothesis densities can be derived from various sources. We have experimented with *feature-induced* grasp hypotheses derived from ECV observations. For example, a pair of similarly-colored, coplanar patches suggests the presence of a planar surface between them. In turn, this plane suggests various possible grasps [13]. Depending on the level of sophistication of such heuristics, feature-induced grasp hypotheses can yield success rates of up to 30% [16]. This is more than sufficient for effective learning of grasp hypothesis densities.

Another source of grasp hypothesis densities is human demonstration. In a pilot study, we tracked human grasps with a Vicon$^{\mathrm{TM}}$ motion capture system [5]. This can be understood as a way of teaching by demonstration: The robot is shown how to grasp an object, and this information is used as a starting point for autonomous exploration.

Illustrations of some experimental results are shown in Figs. 12 and 13. A large-scale series of experiments for quantitative evaluation is currently underway.

4 Discussion

We described two complementary methods for associating actions to perceptions via autonomous, exploratory learning. RLVC is a reinforcement-learning method that operates on a perceptual space defined by low-level visual features. Remarkably, its adaptive, task-driven discretization of the perceptual space allows it to learn policies with a number of interactions similar to problems with much smaller perceptual spaces. This number is nevertheless still greater than what a physical robot can realistically perform. In many practical applications, interactions will have to be generated in simulation. Among the extensions to RLVC, one particularly interesting avenue for further research is RLJC with its uniform treatment of continuous perceptual and action spaces.

RLVC does not learn anything about the world besides task-relevant state distinctions and the values of actions taken in these states. In contrast, the grasp-densities framework involves learning object models that allow the explicit computation of object pose. Object pose is precisely the determining factor for grasping; it is therefore natural to associate grasp parameters to these models. Beyond this association, however, no attempt is made to learn any specifics about the objects or about the world. For example, to grasp an object using grasp densities, the visibility of the contact surfaces in the scene is irrelevant, as the grasp parameters are associated to the object as a whole.

Notably, both learning systems operate without supervision. RL methods require no external feedback besides a scalar reward function. Learning proceeds by trial and error, which can be guided by suitably biasing the exploration strategy. Learning grasp-densities involves learning object models and trying out grasps. Again, the autonomous exploration can be – and normally will need to be – biased via the specification of a suitable grasp hypothesis density, by human demonstration or other means. The Cognitive Vision Group at the University of Southern Denmark, headed by N. Krüger, has put in place a robotic environment that is capable of learning grasp densities with a very high degree of autonomy, requiring human intervention only in exceptional situations. Like a human infant, the robot reaches for scene features and "plays" with objects by attempting to grasp them in various ways and moving them around.

Learning object-relative gripper poses is only the opening chapter of the grasp-density story. The principle can be extended to learning multiple grasp types such as palmar grasps and various multi-fingered pinches, associating hand pre-shapes and approach directions by learning parameters for motor programs, etc. These and other avenues will be pursued in future work.

Acknowledgements. This work was supported by the Belgian National Fund for Scientific Research (FNRS) and the EU Cognitive Systems project PACO-PLUS (IST-FP6-IP-027657). We thank Volker Krüger and Dennis Herzog for their support during the recording of the human demonstration data.

References

1. Bellman, R.: Dynamic programming. Princeton University Press, Princeton (1957)
2. Bertsekas, D., Tsitsiklis, J.: Neuro-Dynamic Programming. Athena Scientific, Belmont (1996)
3. Breiman, L., Friedman, J., Stone, C.: Classification and Regression Trees. Wadsworth International Group (1984)
4. Bryant, R.: Symbolic Boolean manipulation with ordered binary decision diagrams. ACM Computing Surveys 24(3), 293–318 (1992)
5. Detry, R., Başeski, E., Popović, M., Touati, Y., Krüger, N., Kroemer, O., Peters, J., Piater, J.: Learning Object-specific Grasp Affordance Densities. In: International Conference on Development and Learning (2009)
6. Detry, R., Pugeault, N., Piater, J.: A Probabilistic Framework for 3D Visual Object Representation. IEEE Transactions on Pattern Analysis and Machine Intelligence 31(10), 1790–1803 (2009)
7. Gouet, V., Boujemaa, N.: Object-based queries using color points of interest. In: IEEE Workshop on Content-Based Access of Image and Video Libraries, Kauai, HI, USA, pp. 30–36 (2001)
8. Jodogne, S., Piater, J.: Interactive Learning of Mappings from Visual Percepts to Actions. In: 22nd International Conference on Machine Learning, pp. 393–400 (2005)
9. Jodogne, S., Piater, J.: Learning, then Compacting Visual Policies. In: 7th European Workshop on Reinforcement Learning, Naples, Italy, pp. 8–10 (2005)
10. Jodogne, S., Piater, J.: Task-Driven Discretization of the Joint Space of Visual Percepts and Continuous Actions. In: Fürnkranz, J., Scheffer, T., Spiliopoulou, M. (eds.) ECML 2006. LNCS (LNAI), vol. 4212, pp. 222–233. Springer, Heidelberg (2006)
11. Jodogne, S., Piater, J.: Closed-Loop Learning of Visual Control Policies. Journal of Artificial Intelligence Research 28, 349–391 (2007)
12. Jodogne, S., Scalzo, F., Piater, J.: Task-Driven Learning of Spatial Combinations of Visual Features. In: Proc. of the IEEE Workshop on Learning in Computer Vision and Pattern Recognition, Workshop at CVPR, San Diego, CA, USA (2005)
13. Kraft, D., Pugeault, N., Başeski, E., Popović, M., Kragić, D., Kalkan, S., Wörgötter, F., Krüger, N.: Birth of the Object: Detection of Objectness and Extraction of Object Shape through Object Action Complexes. International Journal of Humanoid Robotics 5, 247–265 (2008)
14. Krüger, N., Lappe, M., Wörgötter, F.: Biologically Motivated Multimodal Processing of Visual Primitives. Interdisciplinary Journal of Artificial Intelligence and the Simulation of Behaviour 1(5), 417–428 (2004)
15. Nene, S., Nayar, S., Murase, H.: Columbia Object Image Library (COIL-100). Tech. Rep. CUCS-006-96, Columbia University, New York (1996)

16. Popović, M., Kraft, D., Bodenhagen, L., Başeski, E., Pugeault, N., Kragić, D., Krüger, N.: An Adaptive Strategy for Grasping Unknown Objects Based on Co-planarity and Colour Information (submitted)
17. Pugeault, N.: Early Cognitive Vision: Feedback Mechanisms for the Disambiguation of Early Visual Representation. Vdm Verlag Dr. Müller (2008)
18. Pugeault, N., Wörgötter, F., Krüger, N.: Accumulated Visual Representation for Cognitive Vision. In: British Machine Vision Conference (2008)
19. Samuel, A.: Some Studies in Machine Learning Using the Game of Checkers. IBM Journal of Research and Development 3(3), 210–229 (1959)
20. Sudderth, E., Ihler, A., Freeman, W., Willsky, A.: Nonparametric Belief Propagation. In: Computer Vision and Pattern Recognition, vol. I, pp. 605–612 (2003)
21. Sutton, R., Barto, A.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
22. Vapnik, V.: Statistical Learning Theory. Wiley, New York (1998)
23. Watkins, C.: Learning From Delayed Rewards. Ph.D. thesis, King's College, Cambridge, UK (1989)

Learning Mobile Robot Motion Control from Demonstrated Primitives and Human Feedback

Brenna Argall, Brett Browning, and Manuela Veloso

Abstract. Task demonstration is one effective technique for developing robot motion control policies. As tasks become more complex, however, demonstration can become more difficult. In this work we introduce a technique that uses corrective human feedback to build a policy able to perform an undemonstrated task from simpler policies learned from demonstration. Our algorithm first evaluates and corrects the execution of motion primitive policies learned from demonstration. The algorithm next corrects and enables the execution of a larger task built from these primitives. Within a simulated robot motion control domain, we validate that a policy for an undemonstrated task is successfully built from motion primitives learned from demonstration under our approach. We show feedback to both aid and *enable* policy development, improving policy performance in success, speed and efficiency.

1 Introduction

The appropriate selection of actions is a fundamental challenge within mobile robotics. The development of a robust *policy*, or mapping from world states to robot actions, is complicated by noisy observations and action execution uncertainty. Policy development furthermore is often specific to a particular robot platform and application, and policy reuse for other platforms or application tasks is rare.

One field of effective development approaches has the robot *learn* a policy, from training data or execution experience. Unlike traditional techniques that model world dynamics by hand, an implemented policy learning algorithm may be reused

Brenna Argall · Brett Browning
Robotics Institute, Carnegie Mellon University, USA
e-mail: bargall@ri.cmu.edu,brettb@cs.cmu.edu

Manuela Veloso
Computer Science Department, Carnegie Mellon University, USA
e-mail: mmv@cs.cmu.edu

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 417–432.
springerlink.com

to learn another policy, though the policy itself typically is still platform or application specific. *Learning from Demonstration (LfD)* is a technique that derives a policy from example executions of a target behavior by a teacher. This approach has seen success on a variety of robotics applications, and has the attractive characteristics of being an intuitive means for human teacher to robot learner knowledge transfer, as well as being an accessible policy development technique for non-robotics-experts.

As tasks become more complex, however, demonstration can become more difficult. One practical extension of the LfD approach is to incorporate simpler behaviors learned from demonstration into larger tasks, especially if such tasks are too complex to demonstrate in full. Though scale-up techniques of this nature have been explored within other policy development approaches, the topic remains largely unaddressed within the LfD paradigm. Moreover, the ability to reuse and incorporate existing policies is a practical feature for any approach given the challenge of developing robust control policies. In this work, we contribute an algorithm that builds a more complex policy from existing behaviors learned from demonstration.

How to effectively incorporate existing behaviors into a new policy is a key design decision for this work. We take the approach of aiding this process with human feedback offered in multiple forms, the most notable of which is continuous-valued corrections on student executions. Our framework for providing feedback does not require revisiting states in need of improvement, and thus offers an alternative to the more typical LfD policy improvement approaches that provide further teacher demonstrations. This feature is particularly attractive given our consideration of complex tasks for which full demonstration may be inconvenient or infeasible.

We introduce *Feedback for Policy Scaffolding (FPS)* as an algorithm that builds and refines a complex policy from component behaviors learned from demonstration and teacher feedback. The FPS algorithm operates by first refining multiple policies learned from demonstrated motion primitives. A single complex policy is derived from these primitives, and execution with the complex policy on a novel, undemonstrated behavior is then evaluated. By providing corrections on this execution, the FPS algorithm develops a policy able to execute the more complex behavior, without ever requiring a full demonstration of the novel behavior.

We validate our algorithm within a simulated motion control domain, where a robot learns to drive on a racetrack. A policy built from demonstrated motion primitives and human feedback is developed and able to successfully execute a more complex, undemonstrated task. Feedback is shown to improve policy performance when offered in response both to motion primitive executions as well as novel behavior executions, and moreover the policy developed under this technique is found to perform well within the novel domain. Finally, comparisons to an exclusively demonstration-based approach show the FPS algorithm to be more concise and effective in developing a policy able to execute the more complex behavior.

The following section overviews the related work that supports this approach. In Section 3 the scaffolding algorithm is presented. Section 4 details the experimental implementation, including results and discussion. In the final section we conclude.

2 Background and Related Work

In this section we first present related work on policy development and improvement within demonstration learning, followed by the details of building policies from behavior primitives and teacher feedback.

We formally define the world to consist of states S and actions A, with a probabilistic transition function $T(s'|s,a) : S \times A \times S \rightarrow [0,1]$ describing the mapping between states by way of actions. As we do not assume that state is fully observable, the learner has access to observed state Z through the mapping $M : S \rightarrow Z$. A policy $\pi : Z \rightarrow A$ selects actions based on observations of the world state.

2.1 Learning from Demonstration

Learning from Demonstration (LfD) is a policy development technique in which teacher executions of a desired behavior are recorded and a policy is subsequently derived from the resulting dataset. Formally, we represent a teacher demonstration $d_j \in D$ as t_j pairs of observations and actions such that $d_j = \{(\mathbf{z}^i_j, \mathbf{a}^i_j)\} \in D, \mathbf{z}^i_j \in Z, \mathbf{a}^i_j \in A, i = 0 \cdots t_j$. The set D of these demonstrations are provided to the learner.

When recording and executing demonstrations the issue of *correspondence*, where teacher demonstrations do not directly map to the robot learner due to differences in sensing or motion [9], is key. *Teleoperation* is a demonstration technique whereby the passive learner platform records from its own sensors while being controlled by the teacher during execution. Since the recorded data maps directly to the learner platform, this demonstration technique best minimizes the introduction of correspondence issues into an LfD system. Examples of successful teleoperated LfD systems include both real [10] and simulated [7] robot applications.

Policy derivation amounts to building a predictor that will reproduce the actions $\mathbf{a}^t \in D$ from the observations $\mathbf{z}^t \in D$. Many approaches exist within LfD to derive a policy from the demonstration data [3], the most popular of which either directly approximate the underlying function mapping observations to actions or approximate a state transition model and then derives a policy using techniques such as *Reinforcement Learning*. Our work derives a policy under the first approach, with function approximation being performed via regression since our target application of low-level motion control has a continuous action space.

A wealth of regression approaches exist, and any are compatible with the FPS algorithm. The specific technique used in our implementation is a form of Locally Weighted Learning [4]. In particular, given observation $\mathbf{z}^t$, action $\mathbf{a}^t$ is predicted through an averaging of datapoints in D, weighted by their kernelized distance to $\mathbf{z}^t$. While a more sophisticated regression technique would likely improve the performance of our implementation, the focus of this work is not how to better use *existing* demonstration data, but rather how to use teacher feedback to produce *new* data and thus refine policy performance and build new behavior.

To have a robot learn from its execution performance, or *experience*, is a valuable policy improvement tool, and there are LfD approaches that incorporate learning from experience into their algorithms. For example, execution experience is used to update state transition models [1] and reward-determined state values [13]. Other approaches provide more demonstration data, driven by learner requests for more data [7, 8] as well as more teacher-initiated demonstrations [6].

Our approach similarly provides new example state-action mappings, but the source for these mappings is *not* more teacher demonstration. There are some LfD limitations that more teacher demonstrations cannot address, for instance correspondence discrepancies between the teacher and learner. Moreover, the need to visit states in order to provide execution information is a drawback if certain world states are difficult to reach or dangerous to visit, for example that lead to a rover falling over a cliff. Our technique for policy improvement *synthesizes* new example state-action mappings from teacher feedback and learner executions [2], without requiring state re-visitation by the teacher to provide appropriate behavior information.

2.2 Behavior Primitives and Teacher Feedback

Our approach builds a policy from the demonstration of simpler behavior primitives and teacher feedback, rather than demonstrate a complex task in full. One motivation is that as behaviors become more complex, demonstrating the behavior in full can become more difficult. In this case, the teacher may be able to demonstrate behavior primitives for a task but not the task in full, or provide higher quality demonstrations for subsets of the behavior. A second motivation is that reaching all states encountered during task execution can become increasingly difficult as tasks become more complex. States may be infeasible, inconvenient or undesirable to reach for a variety of reasons that only compound as task complexity increases. A final motivation is that the demonstrated motion primitives may provide a base for multiple complex behaviors. Through the reuse of these primitives, the effort required to develop policies for the complex behaviors reduces.

Within LfD, hand-coded behavior primitives are used to build larger tasks learned from demonstration [11], demonstrated tasks are decomposed into a library of primitives [5, 13] and behavior primitives are demonstrated and then explicitly combined into a new policy by a human [12]. Closest to our work is that of Bentivegna [5], where a robotic marble maze and humanoid playing air hockey reuse learned primitives, and furthermore refine the policy with execution experience. The work demonstrates the full task and then extractes behavior primitives using hand-written rules. Policy improvement is accomplished through an automatic binary reward signal for task failure, used to adjust regression weights on the policy prediction. By contrast, our approach does not demonstrate the full task, and instead demonstrates the primitives individually. Policy improvement is accomplished by generating new examples, from human feedback on practice executions of the primitives and full task.

3 Algorithm

This section presents our *Feedback for Policy Scaffolding (FPS)* algorithm. Under FPS, teacher feedback is used to enable and improve policy behavior at transition points that link demonstrated primitives. In doing so, it enables expression of the full task behavior, without requiring its full demonstration.

Feedback is provided through the framework *Focused Feedback for Mobile Robot Policies (F3MRP)* [3]. The F3MRP framework operates at the stage of low-level motion control, where actions are continuous-valued and sampled at high frequency. A visual presentation of the 2-D ground path of the mobile robot execution serves as an interface through which the teacher selects segments of an execution to receive feedback, which simplifies the challenge of providing feedback to policies sampled at a high frequency. Visual indications of data support during an execution futhermore assist the teacher in the selection of execution segments and feedback type.

Execution *corrections* are offered through a language termed *advice-operators*, first introduced with the *Advice-Operator Policy Improvement (A-OPI)* algorithm [2]. Advice-operators are commonly defined between the student and teacher, and function by performing mathematical computations on the observations or actions of executed data points. In this manner, they provide continuous-valued corrections on a learner execution, without requiring the teacher to provide the exact *value* for the corrections. Instead, the teacher need only select from a finite list of operators, and indicate the portion of the execution requiring improvement.

3.1 Algorithm Execution

Execution of the FPS algorithm occurs in two phases, presented respectively in Algorithms 1 and 2. The first phase develops a policy π_{ξ_j} for each primitive behavior $\xi_j \in \Xi, j = 1 \dots n$, producing a set of policies Π. The second phase develops a policy for a more complex behavior, building on the primitive policies in Π.

3.1.1 Phase 1: Primitive Policy Development

The development of each primitive policy begins with teacher demonstration. Example observation-action pairs recorded during demonstration of primitive behavior ξ_j produce the initial dataset $D_{\xi_j} \in \mathbf{D}$ and policy $\pi_{\xi_j} \in \Pi$. This initial policy is then refined during practice runs consisting of learner executions and teacher feedback.

During the learner execution portion of a practice run (Alg. 1, lines 7-11), the learner first executes the task. At each timestep the learner observes the world, predicting action $\mathbf{a}^t$ according to policy π_{ξ_j} (line 8). Action $\mathbf{a}^t$ is executed and recorded in the prediction trace d, with observation $\mathbf{z}^t$ (line 10). The information recorded in the trace d will be incorporated into the policy update. The global position x^t, y^t and heading θ^t of the mobile robot, and data support τ^t (discussed in Section 3.2.2) of the regression prediction, are recorded into the execution trace tr, for use by the F3MRP framework when visually presenting the ground path taken by the robot.

Algorithm 1. *Feedback for Policy Scaffolding: Phase 1*

```
 1: Given $\mathbf{D}$
 2: initialize $\Pi \leftarrow \{ \}$
 3: for all behavior primitives $\xi_j \in \Xi$ do
 4:   initialize $\pi_{\xi_j} \leftarrow \texttt{policyDerivation}\Big(D_{\xi_j})\Big)$, $D_{\xi_j} \in \mathbf{D}$
 5:   while practicing do
 6:     initialize $d \leftarrow \{\}$, $tr \leftarrow \{\}$
 7:     repeat
 8:       predict $\{ \mathbf{a}^t, \tau^t \} \leftarrow \pi_{\xi_j}(\mathbf{z}^t)$
 9:       execute $\mathbf{a}^t$
10:       record $d \leftarrow d \cup (\mathbf{z}^t, \mathbf{a}^t)$, $tr \leftarrow tr \cup (x^t, y^t, \theta^t, \tau^t)$
11:     until done
12:     advise $\{ \mathcal{F}, \boldsymbol{\Phi} \} \leftarrow \texttt{teacherFeedback}(tr)$
13:     apply $\hat{d}_{\boldsymbol{\Phi}} \leftarrow \texttt{applyFeedback}(\mathcal{F}, \boldsymbol{\Phi}, d)$
14:     update $D_{\xi_j} \leftarrow D_{\xi_j} \cup \hat{d}_{\boldsymbol{\Phi}}$
15:     rederive $\pi_{\xi_j} \leftarrow \texttt{policyDerivation}(D_{\xi_j})$
16:   end while
17:   add $\Pi \leftarrow \Pi \cup \pi_{\xi_j}$
18: end for
19: return $\Pi$
```

During the teacher feedback portion of the practice run, the teacher first indicates a segment $\boldsymbol{\Phi}$ of the learner execution trace requiring improvement (line 12). The teacher further indicates feedback $\mathcal{F}$, which takes the form either of a binary credit to indicate areas of good performance, or an advice-operator to correct the execution within this segment. The application of $\mathcal{F}$ across all points recorded in d and within the indicated subset $\boldsymbol{\Phi}$ generates new data, $\hat{d}_{\boldsymbol{\Phi}}$, which is added to dataset D_{ξ_j} (lines 13,14). Policy π_{ξ_j} for primitive ξ_j is then rederived from this set (line 15).

3.1.2 Phase 2: Policy Scaffolding

The development of the complex policy builds on the primitive policies developed during Phase 1 of the algorithm. Complex policy development therefore does *not* begin with teacher demonstration of the complex task. Two distinguishing features of the second phase of the FPS algorithm are the (i) selection between the action predictions of multiple policies and (ii) selection of a dataset to receive any new synthesized data. Figure 1 presents a schematic of our scaffolding approach, where dashed lines indicate execution cycles that are performed multiple times.

Phase 2 begins with the initialization of more demonstration datasets. Specifically, n empty datasets are generated, each associated with one primitive policy. Notationally, let new data set $D_{\xi_{i+n}}$ be associated with existing primitive dataset D_{ξ_i}, resulting in a total of $2n$ datasets $D_{\xi_j} \in \mathbf{D}$, $j = 1 \cdots 2n$. Colloquially, call dataset D_{ξ_i+n} the *feedback dataset* associated with primitive dataset D_{ξ_i}. Some of the new data generated during learner practice will be added to these feedback datasets (further details are provided in Sec. 3.2.2). The policies derived from the feedback

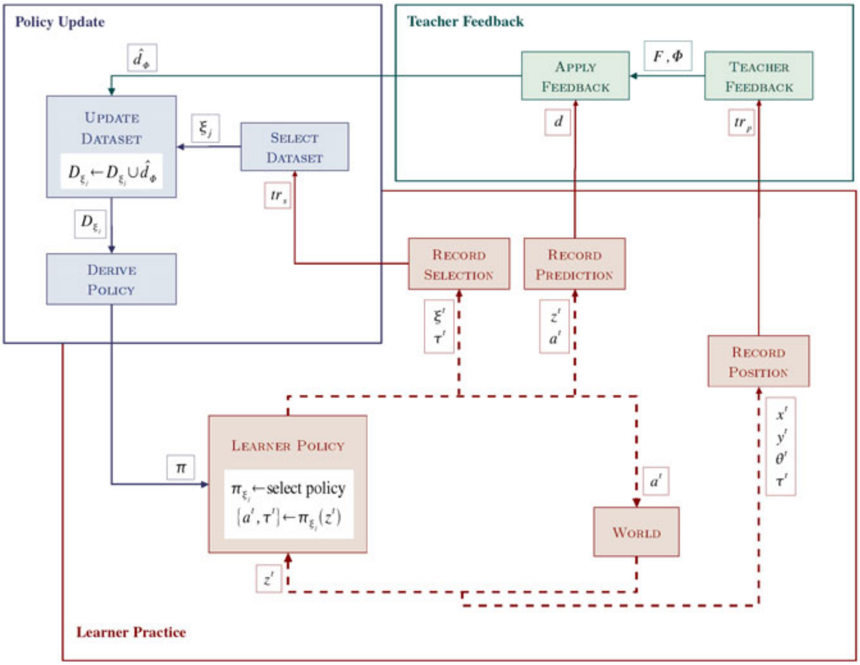


Fig. 1 Policy derivation and execution under the Feedback for Policy Scaffolding algorithm.

datasets are considered, along with the primitive policies, for selection during execution of the more complex policy.

Refinement of the complex policy proceeds with learner execution (Alg. 2, lines 5-10) and teacher feedback (lines 11-17) as in Phase 1, but with the following distinguishing characteristics. The learner now executes with the more complex policy, whose operation proceeds in two steps. The first step is to select between all contributing policies π_{ξ_j} based on observation $\mathbf{z}^t$ (line 6); the details of this selection are provided in Section 3.2.1. The second step is to predict action $\mathbf{a}^t$ according to $\pi_{\xi_j}(\mathbf{z}^t)$, with prediction support τ^t (line 7). After the application of teacher feedback, datasets are individually selected to receive each feedback-modified datapoint. For each point, indexed as $\varphi \in \Phi$, dataset selection (line 14) is determined by the data support τ^φ of $\mathbf{z}^\varphi$ when predicted by policy ξ^φ (recorded in tr_s, line 9); the details of this selection are provided in Section 3.2.2.

3.2 Scaffolding Multiple Policies

Two key factors when building a policy under FPS are how to select between the primitive behaviors, and how to incorporate teacher feedback into the built-up policy. The design of each of these factors within the algorithm is discussed here.

Algorithm 2. *Feedback for Policy Scaffolding: Phase 2*

```
Given $\Pi, \mathbf{D}$
initialize $D_{\xi_i=(n+1)\ldots 2n} \leftarrow \{\ \}$
while practicing do
    initialize $d \leftarrow \{\}$, $tr_s \leftarrow \{\}$, $tr_p \leftarrow \{\}$
    repeat
        select $\pi_{\xi_j} \leftarrow$ policySelection( $\mathbf{z}^t$ ), $\pi_{\xi_j} \in \Pi$
        predict $\{\mathbf{a}^t, \tau^t\} \leftarrow \pi_{\xi_j}(\mathbf{z}^t)$
        execute $\mathbf{a}^t$
        record $d \leftarrow d \cup (\mathbf{z}^t, \mathbf{a}^t)$, $tr_s \leftarrow tr_s \cup (\tau^t, \xi^t = \xi_j)$, $tr_p \leftarrow tr_p \cup (x^t, y^t, \theta^t, \tau^t)$
    until done
    advise $\{F, \Phi\} \leftarrow$ teacherFeedback( $tr_p$ )
    for all $\varphi \in \Phi$, $(\mathbf{z}^\varphi, \mathbf{a}^\varphi) \in d, (\tau^\varphi, \xi^\varphi) \in tr_s$ do
        apply $\hat{d}_\varphi \leftarrow$ applyFeedback( $F, \mathbf{z}^\varphi, \mathbf{a}^\varphi$ )
        select $D_{\xi_\iota} \leftarrow$ datasetSelection( $\tau^\varphi, \xi^\varphi$ ), $D_{\xi_\iota} \in \mathbf{D}$
        update $D_{\xi_\iota} \leftarrow D_{\xi_\iota} \cup \hat{d}_\varphi$
        rederive $\pi_{\xi_\iota} \leftarrow$ policyDerivation($D_{\xi_\iota}$), $\pi_{\xi_\iota} \in \Pi$
    end for
end while
return $\Pi$
```

3.2.1 Selecting Primitive Policies

Primitive selection under FPS assumes that primitives occupy nominally distinct areas of the observation-space. This assumption relies on a state observation formulation that captures aspects of the world that are unique to the demonstrations of each primitive policy. For example, two primitives developed for our validation domain are *turn left* and *turn right*. Observations are formulated to incorporate a notion of track curvature, and so demonstrations in left- versus right-curving areas of the track occupy distinct areas of the observation space.

Primitive selection then is treated as a classification problem. For each primitive ξ_j, a kernelized distance $\phi(\mathbf{z}^t, \mathbf{z}_i)$ between query point $\mathbf{z}^t$ and each point $\mathbf{z}_i \in D_{\xi_j}$ is computed.[1] A weight for policy ξ_j is produced by summing the k largest kernel values $\phi(\mathbf{z}^t, :)$; equivalent to selecting the k nearest points in D_{ξ_j} to query $\mathbf{z}^t$ $(k = 5)$. The policy with the highest weight is then selected for execution.

3.2.2 Incorporating Teacher Feedback

A variety of options exist for how to incorporate synthesized data into the *multiple* underlying datasets of the primitive policies that contribute to the complex behavior

[1] In our implementation the distance computation is Euclidean and the kernel Gaussian, and so $\phi(\mathbf{z}^t, \mathbf{z}_i) = e^{|\mathbf{z}_i - \mathbf{z}^t|\Sigma^{-1}|\mathbf{z}_i - \mathbf{z}^t|}$. The parameter Σ^{-1} is a constant diagonal matrix that scales each observation dimension and embeds the bandwidth of the Gaussian kernel, and is tuned through 10-folds Cross Validation to optimize the least-squared-error on primitive label classification.

execution. To begin, let us establish two ideas. First, this work assumes that in state-space areas covered by the dataset of a particular primitive, the behavior of this primitive matches the intended behavior of the more complex policy. If this is not the case, and the two behaviors conflict, then that primitive should not be incorporated into the complex policy in the first place. Second, both feedback forms produce new data. The new data derives from learner executions, such that every new datapoint $\hat{d}_\varphi = (\hat{\mathbf{z}}^\varphi, \hat{\mathbf{a}}^\varphi)$ derives from an execution point $(\mathbf{z}^\varphi, \mathbf{a}^\varphi)$. Each learner execution point is predicted by a single primitive policy, as discussed in the previous section.

Two factors determine into which dataset a new datapoint $\hat{d}_\varphi$ is added: the policy ξ^φ that predicted the execution point $(\mathbf{z}^\varphi, \mathbf{a}^\varphi)$, and the measure of data support τ^φ for that prediction. In particular, if the policy that made the prediction is a primitive policy, the point is added to its dataset *if* the prediction had strong data support. Otherwise, the point is added to the *feedback* dataset associated with primitive ξ^φ (Sec. 3.1.2). By contrast, data generated from execution points predicted by a feedback policy are automatically added to its dataset, regardless of dataset support.

Prediction support is determined in the following manner. For a given dataset, the 1-Nearest Neighbor Euclidean distance between all points in the set are modelled as a Poisson distribution, parameterized by λ, with mean $\mu = \lambda$ and standard deviation $\sigma = \sqrt{\lambda}$. The threshold on strong prediction support is set by hand, based on empirical evidence ($\tau_{\xi_j} = \mu_{\xi_j} + 5\sigma_{\xi_j}$). Thus a prediction made by policy π_{ξ_j} for query point $\mathbf{z}^t$ with distance $\ell_{\mathbf{z}^t}$ to the nearest point in D_{ξ_j} is classified as strongly supported if $\ell_{\mathbf{z}^t} < \tau_{\xi_j}$ and weakly supported otherwise.

The motivation behind this approach is to avoid adding data to a primitive dataset that conflicts with the behavior of that primitive. Given our assumption that the associated actions of nearby observations express similar behaviors, points that were close enough to the dataset to be strongly supported during prediction therefore are assumed to express behavior similar to that of the primitive.

4 Empirical Implementation

This section presents the experimental details, results and discussion of the application of algorithm FPS to a simulated motion control domain.

4.1 Experimental Setup

The validation domain consists of a simulated differential drive robot performing a racetrack driving task. Robot motion is propagated by simple differential drive simulation of the robot position (1% Gaussian noise), limited in speed and acceleration. The robot observes the world through a monocular camera and wheel encoders; the camera is forward facing and observes track borders (1% Gaussian noise) within its field of view ($130°, 5m$) as a set of points, each of which corresponds to a single image pixel projected into the ground plane. The robot computes a local track representation at each time step ($30Hz$) by fitting a 3-degree polynomial to recently observed track border points. Policy observations are 6-dimensional: current rotational and translational speeds, and the 4 coefficients of the local track polynomial. The actions are 2-dimensional: target rotational and translational speeds.

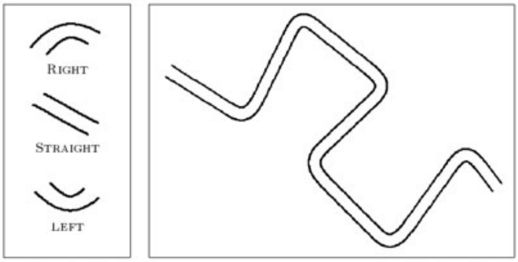


Fig. 2 Primitive subset regions (left) of the full racetrack (right).

The demonstrated motion primitives are: *turn right* (ξ_R), *go straight* (ξ_S) and *turn left* (ξ_L). Demonstrations are performed via human teleoperation, by decreasing or increasing the translational and rotational speeds as the robot moves along the racetrack. The robot has no a priori map of the track, nor does it attempt to build up a map during execution; the aim of the developed policy is to reactively drive on a racetrack. The following steps are taken during policy development:

Demonstrate the motion primitives and derive initial policies. Teacher demonstration of each primitive is performed 3 times on an appropriate track subset (Fig. 2, left). From each dataset a policy is derived, referred to collectively as the set PD_I.

Provide feedback on the motion primitive policies' performance. Learner execution with each policy in PD_I on its respective track subset is observed by the teacher, and feedback-generated data is added to the executing policy's dataset. This observation-*feedback*-update cycle constitutes a single *practice run*, continues to the satisfaction of the teacher and results in final feedback versions of the primitive policies, referred to collectively as the set PF_F.

Derive an initial scaffolded policy from the resultant primitive policies. An initial scaffolded policy SF_I, that selects between the primitive policies in PF_F, is built.

Provide feedback on the scaffolded policy performance. Learner executions with SF_I on the full track are observed by the teacher, and feedback-generated data is added to either the executing policy's dataset *or* its associated feedback dataset (as per Sec. 3.2.2). The observation-feedback-update cycle continues to the satisfaction of the teacher, and results in the final feedback scaffolded policy SF_F.

For comparative purposes, we also evaluate providing more demonstrations. The approach closely follows the policy development steps just outlined, but the teacher provides more teleoperation demonstrations instead of feedback. The result is *demonstration+* versions of a final set of primitives policies (PD_F), initial baseline scaffolded policy (SD_I) and final scaffolded policy (SD_F).

Each of the primitive policies (in sets PD_I, PF_F, PD_F) is evaluated on the track subset appropriate to their respective primitive behavior (Fig. 2, left). Each of the scaffolded policies (SF_I, SF_F, SD_I, SD_F) is evaluated on the full track (Fig. 2, right). Executions end when the robot either runs off the track or reaches the finish line.

Policy performance is measured according to the following metrics. *Success* indicates the ability of the robot to stay on the track, and is measured by the percentage of the track subset (for primitive policy executions) or full track (for scaffolded policy executions) completed. *Speed* is measured by the average translational execution speed. *Efficiency* is measured as the execution time, and is governed jointly by speed and the execution ground path.[2]

4.2 Results

The FPS algorithm successfully learned motion control primitives through a combination of demonstration and teacher feedback, as well as a policy built from these primitives to execute a more complex, *undemonstrated*, behavior. Teacher feedback was found to be critical to the development and performance improvement of all policies, which far outperformed those that received more teacher demonstrations.

4.2.1 Motion Primitives Learned from Demonstration

The three motion primitives were successfully learned in the first phase of the FPS algorithm (Tbl. 1, average of 50 executions, 1-standard deviation).[3]

The initial policies in PD_I were unable to complete either of the *turn right* or *turn left* behaviors. The initial *go straight* primitive behavior was able to complete the task, however execution proceeded extremely slowly.

All policies resulting after Phase 1 development of the FPS algorithm (in PF_F) were able to complete their respective primitive behaviors. Furthermore, executions with these policies were much faster on average than those in PD_I, as summarized in Figure 3 (green bars; also in Tbl. 1). Of particular note is the *go straight* primitive policy, whose *average* speed over the executions approaches the maximum speed of the robot ($3.0\frac{m}{s}$), all without compromising the success of the executions. Aggressive speeds at times negatively impacted the *turn left* policy however, whose more efficient executions came at the cost of occasional incomplete executions.

In contrast to the FPS policy, the *turn right* policy resulting from more teleoperation demonstrations (in PD_F) was not able to complete the task, or even to improve upon the performance or speed of the initial policy. Furthermore, the policy was developed with significantly more practice runs and training data (36 *vs.* 23 practice runs, $2,846$ *vs.* 561 new datapoints). The *go straight* demonstration policy (Fig. 3, blue bar) was able to improve execution speed over the baseline policy, but not as

[2] Efficiency is computed only for successful executions, that by definition do not abort early.

[3] The figures and tables of this section label the primitive policy sets intuitively: *Baseline* refers to PD_I, *Feedback* refers to PF_F and *More Demonstration (More Demo)* refers to PD_F.

Table 1 Execution performance of the primitive policies.

Policy	Success (%)	Speed, Transl [mean] ($\frac{m}{s}$)	Efficiency (s)
Baseline, Right	47.97 ± 1.45	0.61 ± 0.01	-
Feedback, Right	**97.61 ± 12.0**	**1.67 ± 0.02**	1.93 ± 0.07
MoreDemo, Right	51.79 ± 8.54	0.65 ± 0.01	-
Baseline, Straight	**100.0 ± 0.0**	0.60 ± 0.00	5.67 ± 0.13
Feedback, Straight	**100.0 ± 0.0**	**2.74 ± 0.05**	**1.26 ± 0.03**
MoreBaseline, Straight	**100.0 ± 0.0**	1.73 ± 0.34	3.11 ± 0.62
Demo, Left	**99.21 ± 1.31**	0.97 ± 0.01	2.76 ± 0.05
Feedback, Left	91.28 ± 19.30	**1.47 ± 0.39**	**1.80 ± 0.41**
MoreDemo, Left	43.76 ± 8.21	0.60 ± 0.02	-

dramatically as the FPS policy, and again with more training data and practice runs (36 *vs*. 27 practice runs, 1,630 *vs*. 426 new datapoints). The *turn left* policy actually *decreased* the performance of the initial policy, both in success and speed (and with 12 *vs*. 8 practice runs, 965 *vs*. 252 new datapoints). More demonstrations in this case likely created ambiguous areas for the policy, a complication that would perhaps clear up with the presentation of more disambiguating demonstrations.

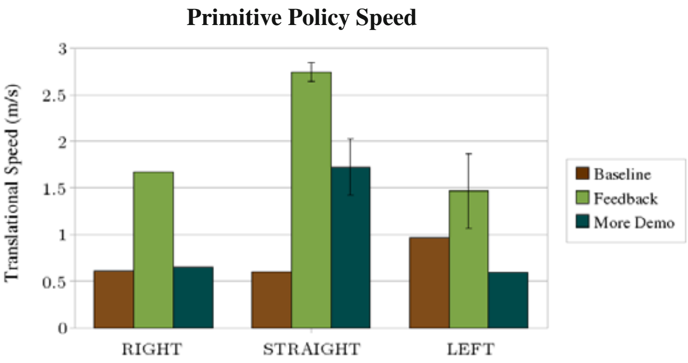


Fig. 3 Average translational execution speed with each of the primitive behavior policies.

4.2.2 Undemonstrated Task Learned from Primitives and Feedback

A policy able to execute a more complex, *undemonstrated*, behavior was successfully developed through the scaffolding of the learned primitive policies, plus the incorporation of teacher feedback. Before any practice runs with teacher feedback, the complex policy, derived solely from selection between the developed feedback primitive policies PF_F, was unable to execute this task in full. Performance improvement over 160 practice runs is presented in Figure 4 (left). Each practice run produces a new iterative policy. Each plot point represents an average of 10 track executions with a given iterative policy, and a regularly sampled subset of the iterative policies were evaluated in this manner (sampled every 10 policies, 17 iterative policies evaluated in total). This constitutes Phase 2 of the FPS algorithm, after which the learner was able to consistently execute the complex task in full.

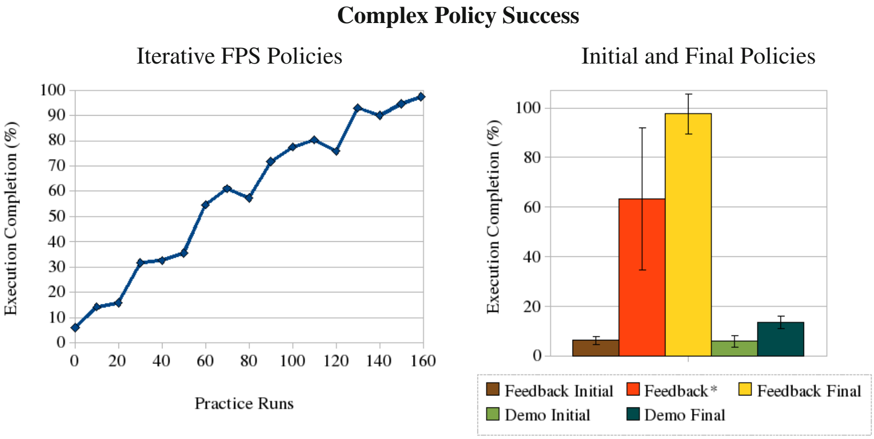


Fig. 4 Percent task completion during (left) and after (right) complex policy practice.

4.2.3 Improvement in Complex Task Performance

Beyond the development of a policy *able* to perform the more complex task, FPS furthermore enabled performance *improvement* such that executions became faster and more efficient (Tbl. 2, average of 50 executions, 1-standard deviation error bars).

Figure 4 (right) summarizes the percent completed execution of multiple policies on the full track task (also in Tbl. 2). The final policy SF_F that resulted after Phase 2 of the FPS algorithm was able to consistently execute the task successfully (*Feedback Final*); as noted above, the initial scaffolded FPS policy SF_I was not able to complete this task (*Feedback Initial*). By contrast, the policy SD_F that resulted from more teleoperation demonstrations (*Demo Final*) was not able to complete the task, though it did improve upon the performance its initial policy (*Demo Initial*).

Table 2 Execution performance of the scaffolded policies.

Policy	Success (%)	Speed, Transl [mean, max] ($\frac{m}{s}$)	Speed, Rot [max] ($\frac{rad}{s}$)
Feedback Initial	6.32 ± 1.72	0.42 ± 0.21 , 1.51 ± 0.36	0.88 ± 0.3
*Feedback**	63.32 ± 28.49	2.24 ± 0.18 , 3.04 ± 0.17	2.14 ± 0.2
Feedback Final	$\mathbf{97.51 \pm 7.94}$	$\mathbf{2.34 \pm 0.03}$, $\mathbf{3.07 \pm 0.01}$	$\mathbf{2.57 \pm 0.06}$
Demo Initial	5.95 ± 0.17	0.58 ± 0.00 , 0.66 ± 0.13	0.15 ± 0.10
Demo Final	13.69 ± 2.36	1.01 ± 0.13 , 1.51 ± 0.56	0.98 ± 0.14

The final FPS policy SF_F however was more extensively developed than the demonstration policy SD_F, whose extremely slow rate of policy improvement prompted the teacher to abort policy development (159 *vs*. 74 practice runs). The above comparison between the final FPS and demonstration policies therefore is not a fair one, and so the results from an *iterative* FPS policy are also provided (*Feedback**). This policy is not the final FPS policy, but rather the result of development after only 74 practice runs, the same number of practice runs as the final demonstration policy. These runs produced $2,448$ new datapoints; far fewer than the 74 runs of the demonstration policy, which produced $8,520$ new points. Even so, against this iterative policy the final demonstration policy also did not measure well. The iterative policy (*Feedback**) significantly outperformed the final demonstration policy (*Demo Final*) on the success measure, though it does not yet perform as successfully or as consistently as the final FPS policy (*Feedback Final*).

The speed performance results closely resemble those of success performance (Tbl. 2). Namely, the final FPS policy far outperformed both the initial FPS policy (*Feedback Initial*) as well as the final demonstration policy (*Demo Final*). The final demonstration policy did offer some improvement over its initial policy (*Demo Initial*), but not nearly as much as the iterative FPS policy provided for comparison (*Feedback**). Interesting to note is that the iterative FPS policy (*Feedback**) produced similar speeds to the final FPS policy, but with larger standard deviations (Tbl. 2), suggesting that performance *consistency*, in addition to execution success, also motivated the teacher to continue development beyond this iterative policy.

4.3 Discussion

This section highlights some noteworthy gains provided by the FPS algorithm, including policy reuse and more focused datasets.

4.3.1 Reuse of Primitives Learned from Demonstration

These empirical results confirm that FPS was able to successfully build a policy for an undemonstrated task, from existing primitive policies learned from demonstration. We identify two crucial gains to such an approach.

The first gain is that the multiple motion primitive policies were developed from *demonstration*. Demonstration has many attractive features as a medium for knowledge transfer from human teacher to robot learner [3]. Moreover, this demonstration technique was aided with teacher *feedback*, provided under the F3MRP framework. Without this feedback, the learned primitive policies are less successful, less efficient, slower, and in some cases even unable to complete the target behavior. This is true not just of the initial primitive policies derived from demonstration, but also of the policies provided with more demonstrations in response to learner execution performance. The FPS algorithm therefore provides a more efficient and effective LfD technique for the development of these motion primitive policies.

Even with the advantages secured through demonstration and teacher feedback however, policy development typically is still a non-trivial task. The second gain of the FPS approach therefore is the ability to *reuse* the primitives within another policy. The full track task was shown to be sufficiently complex that the improvements afforded by demonstrations of the full task behavior were significantly smaller than those gained through teacher feedback. Moreover, this performance difference between the feedback and more-demonstration techniques was much larger for the complex task than for the simpler primitive policies. These results suggest that the complex task cannot be learned through demonstration exclusively, unless perhaps provided with a very large quantity of demonstration data, again underlining the advantage of simple policy reuse within this complex domain.

4.3.2 More Focused Datasets

One result from the experimental validation of A-OPI in [2] was the development of much smaller datasets with corrective feedback in comparison to more demonstration. The smaller datasets furthermore produced similar or superior performance, prompting the conclusion that the datasets were less redundant and more focused.

The same trend is seen in the FPS datasets, and appears to only magnify with the more complex domain of this work. In particular, the combined size of the three primitive policy datasets developed with more demonstration $(6,137)$ is more than three times the size of the comparable FPS primitive datasets $(1,935)$. The size of the final scaffolded more-demonstration policy dataset $(14,657)$ is more than double the final FPS dataset size $(6,438)$, and this is with far fewer practice runs (74 *vs.* 168).

Moreover, these teleoperation policies never perform similarly to their FPS counterparts and, in contrast to the A-OPI results, instead usually display significantly *inferior* performance. This observation suggests that not only is the added data less redundant, but furthermore includes *relevant* data that is *not* being produced by the demonstration. One issue is that to provide a correction through demonstration can be difficult with motion control tasks, and this difficulty scales with task complexity. Further detrimental is the reality that the teacher often must reproduce suboptimal behavior in order to *reach* the state intended to receive a corrective demonstration. We propose that the value in policy refinement alternatives to state revisitation only grows as tasks and domains become more complex.

5 Conclusions

We have introduced *Feedback for Policy Scaffolding (FPS)* as an algorithm that builds, or scaffolds, a policy from demonstrated component behaviors and corrective human feedback. The complete behavior of the scaffolded policy itself need not be demonstrated. We have validated the FPS algorithm within a simulated robot motion control domain. A policy built from demonstrated motion primitives and human feedback was able to execute a more complex, undemonstrated task, thus confirming successful policy reuse. Moreover, we found that successful execution of the complex behavior was in fact *enabled* by teacher feedback. When compared to providing more teacher demonstrations, FPS was shown to produce better performing policies, from more focused datasets. Policy performance was found to *improve* with feedback, in the measures of success, speed and efficiency, and for the complex as well as primitive behaviors.

Acknowledgements. This research is partly sponsored by the Boeing Corporation under Grant No. CMU-BA-GTA-1, BBNT Solutions under subcontract No. 950008572, via prime Air Force contract No. SA-8650-06-C-7606, and the Qatar Foundation for Education, Science and Community Development. The views and conclusions in this document are solely those of the authors.

References

1. Abbeel, P., Ng, A.Y.: Exploration and apprenticeship learning in reinforcement learning. In: Proceedings of ICML (2005)
2. Argall, B., Browning, B., Veloso, M.: Learning robot motion control with demonstration and advice-operators. In: Proceedings of IROS (2008)
3. Argall, B.D.: Learning Mobile Robot Motion Control from Demonstration and Corrective Feedback. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA (2009)
4. Atkeson, C.G., Moore, A.W., Schaal, S.: Locally weighted learning for control. Artificial Intelligence Review 11, 75–113 (1997)
5. Bentivegna, D.C.: Learning from Observation Using Primitives. PhD thesis, College of Computing, Georgia Institute of Technology, Atlanta, GA (July 2004)
6. Calinon, S., Billard, A.: Incremental learning of gestures by imitation in a humanoid robot. In: Proceedings of HRI (2007)
7. Chernova, S., Veloso, M.: Multi-thresholded approach to demonstration selection for interactive robot learning. In: Proceedings of HRI (2008)
8. Grollman, D.H., Jenkins, O.C.: Dogged learning for robots. In: Proceedings of ICRA (2007)
9. Nehaniv, C.L., Dautenhahn, K.: The correspondence problem, ch. 2. MIT Press, Cambridge (2002)
10. Ng, A., Coates, A., Diel, M., Ganapathi, V., Schulte, J., Tse, B., Berger, E., Liang, E.: Inverted autonomous helicopter flight via reinforcement learning. In: International Symposium on Experimental Robotics (2004)
11. Nicolescu, M.N., Mataric, M.J.: Methods for robot task learning: Demonstrations, generalization and practice. In: Proceedings of AAMAS (2003)
12. Saunders, J., Nehaniv, C.L., Dautenhahn, K.: Teaching robots by moulding behavior and scaffolding the environment. In: Proceedings of HRI (2006)
13. Stolle, M., Atkeson, C.G.: Knowledge transfer using local features. In: Proceedings of ADPRL (2007)

Perceptual Interpretation for Autonomous Navigation through Dynamic Imitation Learning

David Silver, J. Andrew Bagnell, and Anthony Stentz

Abstract. Achieving high performance autonomous navigation is a central goal of field robotics. Efficient navigation by a mobile robot depends not only on the individual performance of perception and planning systems, but on how well these systems are coupled. When the perception problem is clearly defined, as in well structured environments, this coupling (in the form of a cost function) is also well defined. However, as environments become less structured and more difficult to interpret, more complex cost functions are required, increasing the difficulty of their design. Recently, a class of machine learning techniques has been developed that rely upon expert demonstration to develop a function mapping perceptual data to costs. These algorithms choose the cost function such that the robot's planned behavior mimics an expert's demonstration as closely as possible. In this work, we extend these methods to address the challenge of dynamic and incomplete online perceptual data, as well as noisy and imperfect expert demonstration. We validate our approach on a large scale outdoor robot with hundreds of kilometers of autonomous navigation through complex natural terrains.

1 Introduction

The capability of mobile robots to autonomously navigate through unknown environments continues to advance. Especially in structured environments, the effectiveness of both perception and planning systems have been greatly improved. In such environments, the actions that a robot can and cannot take are generally well defined. It is then the task of a perception system to report these definitions, and of a planning system to indicate a series of actions to achieve a desired goal.

However, as environments become less structured, the difficulty of coupling perception and planning systems increases. Under these conditions, the final output of a perception system cannot be binary: a more analog measure of traversability or desirability is required. For example, in outdoor terrain it may be desirable to avoid

David Silver · J. Andrew Bagnell · Anthony Stentz
Robotics Institute, Carnegie Mellon University

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 433–449.
springerlink.com

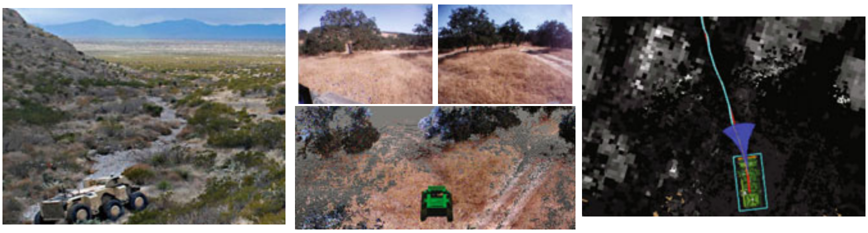


Fig. 1 **Left:** The Crusher platform is capable of long range autonomous navigation through complex terrain. **Center:** Raw perception data from Crusher, in the form of camera images and colorized LiDAR. **Right:** 2D costs derived from perception data. Brighter pixels indicate higher cost.

tall grass in favor of short grass, but it may also be desirable to avoid a bush in favor of either, and to avoid a tree in favor of anything. The computation of such an analog ordering has a drastic effect on the final performance of a mobile robot, and is often a barrier to truly robust navigation.

This paper proposes the use of imitation learning to derive the proper coupling between a mobile robot's perception and planning systems in order to improve autonomous performance. The presented approach is based on extension of the Maximum Margin Planning (MMP) [12, 13] framework, and makes use of expert demonstration of desired navigation behavior. Existing techniques are adapted to account for the unknown and dynamic nature of online perceptual data, as well as to account for noisy and imperfect demonstration. The application of this approach to the Crusher autonomous platform [18] (Figure 1) is then presented. Experimental results are gathered over hundreds of kilometers of autonomous traverse through complex, natural terrains.

2 Perceptual Interpretation for Autonomous Navigation

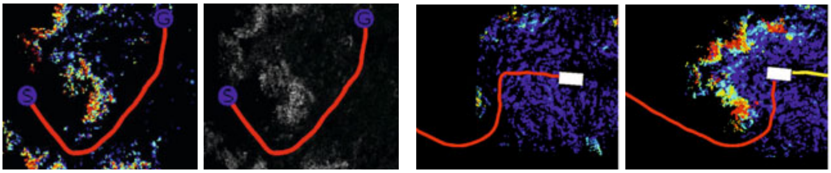
The canonical task of an autonomous navigation system is to safely guide a robot from a start to a goal location. Knowledge of the environment comes from a combination of any prior data available, and data collected by the robot's onboard sensors. Due to map resolution limitations and post mapping changes, the environment is typically not fully known before the robot navigates. Since the robot's knowledge of its environment evolves over time, an onboard planning system must continually make decisions geared towards achieving its goal. Depending on the complexity of the navigation task, the overall planning system may consist of multiple individual planners. It is also common for such systems to operate in a discretized state space that allows perceptual data to be associated with each state.

A planning system attempts to produce the optimal sequence of actions that will lead the robot to its goal. This naturally raises the question of how to determine what is "optimal". In well structured environments, this may be well defined: for example, in an indoor environment, the optimal path may be the shortest collision free path to the goal. The perceptual challenge in this scenario is to properly classify sources of collisions; after that, defining optimality is trivial.

However, in complex unstructured environments, defining optimality is more difficult (along with the rest of the perception task). In such environments it is no longer easily definable what is and is not traversable; for example it may or may not be possible to drive over a small bush, or to drive through a large patch of mud without becoming stuck. Traversability must be considered as a probability or some other analog measure. Further, maximizing traversability may not be the only consideration; there might be a desire to minimize energy usage or time taken, maximize the field of view of onboard sensors, or minimize exposure to external sensors. In these scenarios, an analog ordering of preferences and tradeoffs is required, as opposed to a binary classification of passable regions. This in turn creates the requirement for planners that can make use of such orderings. One of the most field proven approaches is to use A* style grid planners, either separately or in conjunction with more continuous local motion planners [7, 15, 17, 18].

The process of converting a perception system's description of an environment to an ordering of preferences is often called costing, with a cost function mapping perceptual data associated with discrete states to costs. In turn, it is then the function of the planning system to attempt to minimize the accrued cost during navigation. By concretely defining relative preferences and tradeoffs, the cost function has a large impact on a mobile robot's behavior. Often, it is chosen in an attempt to force a robot to approximate some intuitive metric (distance traveled, time taken, energy expended, mission risk, etc.) or combination of such. Whether the robot's behavior actually reflects its designed intent is heavily dependent on the mapping from perceptual features to costs. If the perceptual features are sufficiently descriptive of the environment, the cost function can be seen as an encoding of the desired behavior of the robot.

Unfortunately, the more complex the environment and the desired behavior, the more difficult the task of defining the cost function becomes. Human engineering is often the first approach to tackling this problem; it is also potentially quite time consuming. Complex environments necessitate full featured and high dimensional descriptions, often on the order of dozens of features per discrete state. Worse still, there is often not a clear relationship between features and cost. Therefore, engineering a cost function by hand is akin to manually solving a high dimensional optimization problem using local gradient methods. Evaluating each candidate function then requires validation through either actual or simulated robot performance. This tedious process must be repeated whenever the input perceptual features are modified, or incorrect behavior is observed in a novel environment. Despite these drawbacks, manual engineering has been popular for lack of alternatives [4, 5, 7, 17]. Engineering or learning more intuitive features (with respect to costing) simplifies but does not solve this problem (while creating additional work elsewhere in the perception pipeline). Self-supervised approaches that learn to interpret an environment online through proprioception [3, 8, 19] can also provide a more intuitive feature space; however the requirement of defining relative tradeoffs remains.



(a) (b)

Fig. 2 **(a)** An example behavior (in red) overlayed on a single perceptual feature (obstacle density), and the cost function learned to reproduce the example behavior **(b)** Example behavior from time t (left) and $t+1$ (right), overlayed on a single perceptual feature (obstacle height). Future behavior is inconsistent at time t, but makes sense at time $t+1$ given additional perceptual data.

As opposed to simply reducing the complexity of defining desired behavior, the imitation learning approach seeks to learn a direct mapping to behaviors. Much previous work in this field [9, 11] has been focused on action prediction: given the current robot state and its associated perceptual features, learn to predict what action an expert would perform, based on examples of expert behavior. However, this approach is inherently myopic, as it assumes that all necessary information is contained in the current state. While such an approach may work for reactive planning systems, it is generally ineffective for more deliberative, goal oriented systems.

Recently, a new set of approaches have been developed for learning from demonstration based on the concept of *Inverse Optimal Control* [6]. Rather than learn a mapping from perceptual features to actions, these approaches seek to learn a mapping from perceptual features to costs, such that a planner minimizing said costs will achieve the expert demonstrated behavior (Figure 2(a)). These methods take advantage of the fact that while it is difficult for an expert to define an ordering of preferences, it is easy for an expert to demonstrate the desired behavior.

Our work makes use of the Maximum Margin Planning (MMP) [12] framework, as opposed to the Inverse Reinforcement Learning approach of [2, 1]. The primary benefits of the MMP approach are that it does not require a mixture of policies, and explicitly seeks to reproduce expert behaviors. Further, MMP has been extended to allow for nonlinear cost functions. Previous work [15] has demonstrated the applicability of the MMP framework to autonomous navigation using static perceptual data. The next section summarizes this approach, and extends it to the dynamic setting where perceptual representations evolve over time.

3 Maximum Margin Planning for Dynamic Perceptual Interpretation

This section first describes previous work in the MMP framework, and specifically the LEARCH (LEArning to seaRCH) algorithm [14], for learning a cost function to reproduce expert demonstrated behavior. Extension of LEARCH to handle dynamic, unknown environments is then presented. The MMP framework was developed in the context of Markov Decision Processes, and therefore can handle a cost function defined over state-action pairs. For the sake of simplicity and clarity, the following introduction and subsequent derivations only consider A* style planners, and costs defined over states (this is consistent with how Crusher's autonomy system operates). However, this is not a limitation of the MMP framework itself, which can be applied to any planning-based decision loop. For a full derivation, see [14].

3.1 LEARCH Algorithm

Consider a state space $\mathscr{S}$, and a feature space $\mathscr{F}$ defined over $\mathscr{S}$. That is, for every $x \in \mathscr{S}$, there exists a corresponding feature vector $F_x \in \mathscr{F}$. C is defined as a cost function over the feature space, $C : \mathscr{F} \to \mathbb{R}^+$. The cost of a state x is $C(F_x)$. A path P is defined as a sequence of states in $\mathscr{S}$ that lead from a start state s to a goal state g. The cost of P is simply the sum of the costs of all states along it.

If an example path P_e is provided, then MMP defines the following constraint on cost functions: the cost of P_e must be lower than the cost of any other path from s_e to g_e. The structured maximum margin approach [12] encourages good generalization and prevents trivial solutions (e.g. the everywhere 0 function) by augmenting the constraint to includes a margin: the demonstrated path must be BETTER than another path by some amount. The size of the margin is dependent on the similarity between the two paths, encoded in a loss function L_e. In this context, we define loss by how many states the two paths share. Learning a cost function then involves constrained optimization of an objective functional over C

$$\min \mathscr{O}[C] = \textsc{Reg}(C)$$

subject to the constraints

$$\sum_{x \in \hat{P}} (C(F_x) - L_e(x)) \; - \; \sum_{x \in P_e} (C(F_x)) \geq 0$$

$$\forall \hat{P} \; s.t. \; \hat{P} \neq P_e, \;\; \hat{s} = s_e, \;\; \hat{g} = g_e$$

$$L_e(x) = \begin{cases} 1 \text{ if } x \in P_e \\ 0 \text{ otherwise} \end{cases} \tag{1}$$

where REG is a regularization operator that encourages generalization in the cost function C. There are typically an exponentially large (or even infinite) number of constraints, each corresponding to an alternate path. However, it is not necessary to enumerate these constraints. For every candidate cost function, there is a minimum cost path between two waypoints; at each step it is only necessary to enforce the constraint on this path. Further, it may not always be possible to achieve all constraints, and thus a "slack" penalty is added. Since the slack variable is tight, we may write an "unconstrained" problem that captures the constraints as penalties:

$$\min \mathscr{O}[C] = \; \textsc{Reg}(C) + \sum_{x \in P_e} C(F_x) - \; \min_{\hat{P}} \sum_{x \in \hat{P}} (C(F_x) - L_e(x)) \tag{2}$$

For linear cost functions (and convex regularization) $\mathscr{O}[C]$ is convex, and can be minimized using (sub-)gradient descent. However, linear cost functions may be insufficient. Therefore, we consider the (sub-)gradient of the objective in the space of cost functions [13, 14], given by

$$\bigtriangledown \mathscr{O}_F[C] = \sum_{x \in P_e} \delta_F(F_x) \ - \ \sum_{x \in P_*} \delta_F(F_x) \tag{3}$$

where P_* is the current minimum cost plan, and δ is the dirac delta at the point of evaluation. Simply speaking, the functional gradient is positive at values of F that the example path pass through, and negative at values of F that the planned path pass through. The magnitude of the gradient is determined by the frequency of visits. If the paths agree in their visitation counts at F, the functional gradient is zero.

Applying gradient descent in this space of cost functions directly would involve an extreme form of overfitting: defining a cost at every value of F encountered and involving no generalization. Instead we take a small step in a limited set of "directions" using a hypothesis space of functions mapping features to a real number. The resulting function R belongs to a chosen family of functions (linear, decision trees, neural networks, etc.). The choice of hypothesis space in turn controls the complexity of the resulting cost function. We must then find at each step the element R_* that maximizes the inner product $\langle - \bigtriangledown \mathscr{O}_F[C], R_* \rangle$ between the functional gradient and elements of the space of functions under consideration. Maximizing the inner product between the functional gradient and R_* can be interpreted as a learning problem:

$$R_* = \arg\max_R \sum_{x \in P_e \cup P_*} \alpha_x y_x R(F_x) \tag{4}$$

$$\alpha_x = | \bigtriangledown \mathscr{O}_{F_x}[C] | \quad y_x = -\mathrm{sgn}(\bigtriangledown \mathscr{O}_{F_x}[C])$$

In this form, it can be seen that finding the projection of the functional gradient essentially involves solving a weighted classification problem. Performing regression instead of classification adds an additional regularization to each projection. Intuitively, the regression targets are positive in places the planned path visits more than the example path, and negative in places the example path visits more. The weights on each regression target are the difference in visitation counts.

Gradient descent can be understood as encouraging functions that are "small" in the l_2 norm. If instead, we consider applying an *exponentiated functional gradient descent* update as described in [14] we encourage functions that are "sparse". Thus the final cost function is of the form

$$C(F) = e^{\sum \eta_i R_i(F)} \tag{5}$$

naturally resulting in cost functions that map to $\mathbb{R}^+$. Regularization is achieved implicitly through the learning rate η_i.

With the LEARCH algorithm, multiple expert examples can be provided. Each example behavior between a single start and end waypoint defines its own objective function, and in turn its own regression targets. These targets can be tallied and a new cost function computed for each example one at a time, or all at once with a single update to the cost function.

3.2 LEARCH for Unknown, Dynamic Environments

Previous implementations of LEARCH for mobile robot navigation [15] have only considered the scenario where the mapping from states to features is static and fully known a priori. In this section, we build on [13] and extend the LEARCH algorithm to the scenario where neither of these assumptions holds, such as when features are generated from a mobile robot's perception system. The limited range inherent in onboard sensing means a great deal of the environment may be unknown; for truly complex navigation tasks, the distance between waypoints is generally at least one or two orders of magnitude larger than the sensor range. Further, changing range and point of view from environmental structures means that even once an object is within range, its perceptual features are continually changing. Finally, there are the actual dynamics of the environment: objects may move, lighting and weather conditions can change, sensors may be noisy, etc.

Since the perceptual inputs are no longer static, the robot's current plan must also be continually recomputed. The original MMP constraint must be altered in the same way: rather than enforcing the optimality of the entire example behavior once, the optimality of all example behavior must be continually enforced as the current plan is recomputed. Formally, we add a time index t to account for dynamics. F_x^t represents the perceptual features of state x at time t. P_e^t represents the example behavior starting from the current state at time t to the goal. The objective becomes

$$\min \mathscr{O}[C] = \textsc{Reg}(C) + \sum_t \left(\sum_{x \in P_e^t} C(F_x^t) - \min_{\hat{P}^t} \sum_{x \in \hat{P}^t} \left(C(F_x^t) - L_e^t(x) \right) \right) \qquad (6)$$

with the extra summation over time carried into the functional gradient and its projection in Equation 4. The cost function C does not have a time index: the optimization is searching for the single cost function that best reproduces example behavior over an entire time sequence.

It is important to clarify what P_e^t represents. Until now, the terms *plan* and *behavior* have been interchangeable. This is true in the static case since the environment never evolves; as long as a plan is sufficiently followed, it does not need to be recomputed. However, in the dynamic case, an expert's plan and behavior are different notions: the plan is the currently intended future behavior, and the behavior is the result of previous plans. Therefore, P_e^t would ideally be the expert's *plan* at time t, not example behavior from time t onwards.

However, this information is generally not available: it would require the recording of an expert's current plan at each instant in time. Even if a framework for such a data collection were to be implemented, it would turn the collection of training examples into an extremely tedious and expensive process. Therefore, in practice we approximate the current plan of an expert P_e^t with the expert's behavior from t onwards. Unfortunately, this approximation can potentially create situations where the example at certain timesteps is suboptimal or inconsistent. The consequences of this inconsistency are further discussed in Section 4 (see Figure 2(b)).

Once dynamics have been accounted for, the limited range of onboard sensing can be addressed. At time t, there may be no perceptual features available corresponding to the (potentially large) section of the example path that is outside of current perceptual range. In order to perform long range navigation, a mobile robotic system must already have some approach to planning through terrain it has not directly sensed. Solutions include the use of prior knowledge [15], extrapolation from recent experience [10], or simply to assume uniform properties of unknown terrain.

Therefore, we define the set of visible states at time t as $\mathscr{V}^t$. The exact definition of visible depends on the specifics of the underlying robotic system's data fusion: $\mathscr{V}^t$ should include all states for which the cost of state x at time t is computed with the cost function currently being learned, C. For all other states $\bar{\mathscr{V}}^t$, we can assume the existence of some alternate function for computing cost, $C_{\bar{\mathscr{V}}}(x)$. The objective functional and gradient become

$$\begin{aligned}
&\min \mathscr{O}[C] = \text{REG}(C) + \text{C}_\text{E} - \text{C}_\text{P} \\
&\text{C}_\text{P} = \sum_t \min_{\hat{P}^t} \left(\sum_{x \in \hat{P}^t \cap \mathscr{V}^t} (C(F_x^t) - L_e^t(x)) + \sum_{x \in \hat{P}^t \cap \bar{\mathscr{V}}^t} C_{\bar{\mathscr{V}}}(x) \right) \\
&\text{C}_\text{E} = \sum_t \left(\sum_{x \in P_e^t \cap \mathscr{V}^t} C(F_x^t) + \sum_{x \in P_e^t \cap \bar{\mathscr{V}}^t} C_{\bar{\mathscr{V}}}(x) \right)
\end{aligned} \tag{7}$$

$$\triangledown \mathscr{O}_F[C] = \sum_t \left(\sum_{x \in P_e \cap \mathscr{V}^t} \delta_F(F_x^t) - \sum_{x \in P_* \cap \mathscr{V}^t} \delta_F(F_x^t) \right) \tag{8}$$

Since the gradient is computed with respect to C, it is only nonzero for visible states. The projection of the functional gradient onto the hypothesis space becomes

$$R_* = \arg\max_R \sum_t \sum_{x \in (P_e \cup P_*) \cap \mathscr{V}^t} \alpha_x^t y_x^t R(F_x^t) \tag{9}$$

Although the functional gradient is zero over $\bar{\mathscr{V}}^t$, $C_{\bar{\mathscr{V}}}$ still factors into the planned behavior. Just as LEARCH learns C to recreate desired behavior when using a specific planner, it learns C to recreate behavior when using a specific $C_{\bar{\mathscr{V}}}$. However, if the example behavior is inconsistent with $C_{\bar{\mathscr{V}}}$, it will be more difficult for the planned behavior to match the example. Such an inconsistency could occur if the expert has different prior knowledge than the robot, or interprets the same knowledge differently (a problem addressed in [15]). Inconsistency can also occur due to the previously discussed mismatch between expert plans and expert behavior. Solutions to inconsistent examples are discussed in Section 4.

Contrasting the final form for R_* with that of (4) helps to summarize the changes that result in the LEARCH algorithm for dynamic environments. Specifically, a single expert demonstration from start to goal is discretized by time, with each timestep serving as an example of what behavior to plan given all data to that point in time. For each of these discretized examples, only visitation counts in visible states are used. The resulting algorithm is presented in Figure 3.

```
for $i = 1...N$ do
    foreach $P_e^t$ do
        $\mathscr{M}^t =$ buildCostmap($C_{i-1}, s_e^t, g_e^t, \mathscr{F}^t$);
        $P_*^t =$ planPath($s_e^t, g_e^t, \mathscr{M}^t$);
        $U_-^{e,t} = \{P_e^t \bigcap \bar{P}_*^t, -1\}$, $U_+^{e,t} = \{P_*^t \bigcap \bar{P}_e^t, +1\}$;
    $R_i =$ trainRegressor($\mathscr{F}, U_+ \bigcup U_-$);
    $C_i = C_{i-1} * e^{\eta_i R_i}$;
```

Fig. 3 The dynamic LEARCH algorithm

4 Imperfect and Inconsistent Demonstration

The MMP framework makes the assumption that the provided training data are examples of optimal behavior. Generally, this is not the case, as humans rarely behave in a truly optimal manner. Fundamentally, there will always be noise in human demonstration. Further, multiple examples from different environments and experts may be inconsistent with each other, due either to inconsistency in human behavior, or an incomplete perceptual description of the environment by the robot. Finally, sometimes experts are flat out wrong, and produce poor demonstration.

While MMP is not broken by poor training data, it does suffer degraded overall performance and generalization (in the same way that supervised classification performance is degraded but not broken by mislabeled training data). Attempting to have an expert sanitize the training input is disadvantageous for two reasons: it creates an additional need for human involvement, and assumes that an expert can detect all errors. Instead, this section describes modifications to the LEARCH algorithm that can increase robustness and improve generalization in the face of noisy or poor expert demonstration.

4.1 Unachievable Example Behaviors

Experts do not necessarily plan their example behavior in a manner consistent with a robot's planner: this assumption is not part of the MMP framework. However, what is assumed is that there exists some cost function that will cause the robot's planner to reproduce the behavior. This is not always the case: it is possible for an example to be *unachievable*. For example, an expert may give an inconsistently wide berth to obstacles, or make wider turns than are necessary. The result is that example paths often take slightly longer routes through similar terrain than are optimal [15].

The effect of such an unachievable example is to drive costs towards zero on the terrain in question, since this would result in any path being optimal. However, since costs are constrained to $\mathbb{R}^+$, this will never be achieved. Instead an unachievable example will have the effect of unnecessarily lowering costs over a large section of the feature space, and artificially reducing dynamic range.

This effect can be counteracted by performing a slightly different regression or classification when projecting the gradient. Instead of minimizing the weighted error, the balanced weighted error is minimized; that is, both positive and negative regression targets (corresponding to states on the planned and example path) make an equal sum contribution. Formally, R_* is replaced with R_*^B defined as

$$\begin{aligned} R_*^B &= \arg\max_R \sum_t \left(\sum_{y_x^t > 0} \frac{\alpha_x^t R(F_x^t)}{N_+} - \sum_{y_x^t < 0} \frac{\alpha_x^t R(F_x^t)}{N_-} \right) \\ N_+ &= \sum_t \sum_{y_x^t > 0} \alpha_x^t \quad N_- = \sum_t \sum_{y_x^t < 0} \alpha_x^t \end{aligned} \tag{10}$$

This new projection will zero out the contributions of the unachievable cases described above. In the general case, the effect of balancing can be observed by rewriting the final classification in terms of the planned and example visitation counts, U_+ and U_-, and carrying the balancing through to the inputs.

$$\begin{aligned} R_* = \arg\max \langle R, U_+ - U_- \rangle \quad & R_*^B = \arg\max \langle R, \frac{U_+}{N_+} - \frac{U_-}{N_-} \rangle \\ \langle U_+ - U_-, \frac{U_+}{N_+} - \frac{U_-}{N_-} \rangle &= \frac{\langle U_+, U_+ - U_- \rangle}{N_+} + \frac{\langle U_-, U_- - U_+ \rangle}{N_-} \\ &= \frac{|U_+|^2}{N_+} + \frac{|U_-|^2}{N_-} - (\frac{1}{N_+} + \frac{1}{N_-}) \langle U_+, U_- \rangle \end{aligned}$$

The similarity between inputs to the projections is negatively correlated to the overlap of the positive and negative visitation counts, $\langle U_+, U_- \rangle$. By the Cauchy-Schwarz inequality, $\langle U_+, U_- \rangle$ is bounded by $|U_+||U_-|$, and is only tight against this bound when the visitation counts are perfectly correlated, which implies

$$\langle U_+, U_- \rangle = |U_+||U_-| \iff U_- = \pm \kappa U_+ \implies |U_-| = \kappa |U_+| \,, \, N_- = \kappa N_+$$

for some scalar κ. By substitution

$$\begin{aligned} \frac{|U_+|^2}{N_+} + \frac{|U_-|^2}{N_-} - (\frac{1}{N_+} + \frac{1}{N_-}) \langle U_+, U_- \rangle &\geq \frac{|U_+|^2}{N_+} + \frac{|U_-|^2}{N_-} - (\frac{1}{N_+} + \frac{1}{N_-}) |U_+||U_-| \\ &= \frac{|U_+|^2}{N_+} + \frac{\kappa^2 |U_+|^2}{\kappa N_+} - (\frac{1}{N_+} + \frac{1}{\kappa N_+}) \kappa |U_+||U_+| \\ &= \frac{|U_+|^2}{N_+} + \frac{\kappa |U_+|^2}{N_+} - \frac{|U_+|^2}{N_+} - \frac{\kappa |U_+|^2}{N_+} \;=\; 0 \end{aligned}$$

therefore

$$\langle U_+ - U_-, \frac{U_+}{N_+} - \frac{U_-}{N_-} \rangle \geq 0$$

When there exists clear differentiation between what features should have their costs increased and decreased, the projection inputs will be similar. As the example and current planned behaviors travel over increasingly similar terrain, the inputs diverge; the contribution of the balanced projection to the current cost function will level out, while that of the unbalanced projection will increase in the direction of the longer path. This effect is observed empirically in Section 5.

4.2 Replanning with Corridor Constraints

A balanced projection can account for large scale suboptimality in demonstration. However, suboptimality can also occur at a smaller scale. It is unreasonable to ever expect a human to drive the exact perfect path; it is often the case that a plan that travels through neighboring or nearby states would be a slightly better example. What is needed is an approach that smoothes out small scale noise in expert demonstration, producing a better training example. Such a smoothed example can be derived from expert demonstration by redefining what the example represents: instead of example behavior being interpreted as the exact optimal behavior, it can be interpreted as a behavior that is close to optimal. The exact definition of close depends on the state space; the loss function will always provide at least one possible metric.

For example, if the state space is $\mathbb{R}^n$, then Euclidean distance is a natural metric. Rather than an example defining the exact optimal path, it would define a corridor in which the optimal path exists. A new desired behavior can be derived from the example at each learning iteration by choosing the current optimal behavior that is sufficiently close. Formally, $\mathrm{C_E}$ in (7) is redefined as

$$\begin{aligned}\mathrm{C_E} =& \sum_t \min_{\hat{P}^t_e \in \mathcal{N}^t_e} \left(\sum_{x \in \hat{P}^t_e \cap \mathcal{V}^t} C(F^t_x) + \sum_{x \in \hat{P}^t_e \cap \bar{\mathcal{V}}^t} C_{\bar{\mathcal{V}}}(x) \right) \\ \mathcal{N}^t_e =& \{P \mid \ ||(P \bigcap \mathcal{V}^t) - (P^t_e \bigcap \mathcal{V}^t)|| < \beta\} \end{aligned} \tag{11}$$

where β is a threshold on the metric over paths. The result of this modification is that the example path is replanned at each iteration of the LEARCH algorithm; however, only paths within $\mathcal{N}^t_e$ are considered during replanning.

It should be noted that $\mathcal{N}^t_e$ only defines closeness over $\mathcal{V}^t$. Behavior outside of $\mathcal{V}^t$ does not directly affect the gradient, but does affect the difference in cost between the current (replanned) example and planned behavior. Therefore, by performing a replanning step (even with $\beta = 0$), example behavior can be made consistent with $C_{\bar{\mathcal{V}}}$ without compromising its effectiveness as an example within $\mathcal{V}^t$.

4.3 Filtering for Inconsistent Examples

By always performing a replanning step, it can be ensured that any remaining inconsistency is due to either a mismatch between an expert's planned and actual behavior at time t, a disagreement between the expert's and the robot's underlying perception of the world, or expert error. Figure 2(b) provides a simple example: at time t, the expert likely planned to drive straight, but was forced to replan at time $t+1$ as new obstacles were observed. The assumption that the expert behavior from time t onward matches the plan is false, and results in behavior at time t being inconsistent with the behavior exhibited at other timesteps. While tiny changes in an expert's plan can be corrected by replanning the example, there is no way to correct for a drastic case such as a cul de sac.

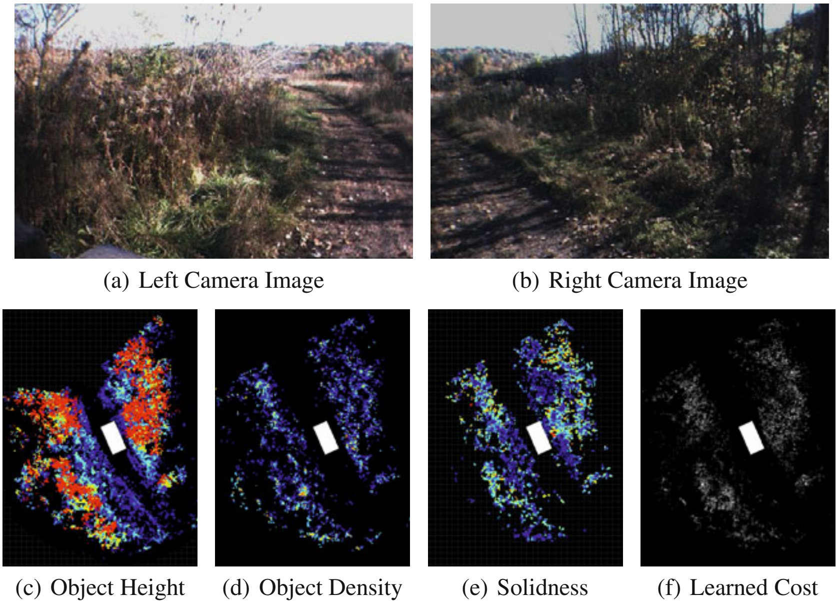
However, the inconsistency of such timesteps provides a basis for their filtering and removal. Specifically, it can be assumed that a human expert will plan in a fairly consistent manner during a single example traverse. If the behavior from a single timestep or small set of timesteps is inconsistent with the demonstrated behavior at other timesteps, then these inconsistent timesteps can be filtered. Inconsistency can be quantitatively defined by observing each timestep's contribution to the objective functional (its slack penalty). If the penalty at a single timestep is a statistical outlier from other timesteps, then that timestep can be seen as inconsistent with the constraints implied by the rest of the example behavior.

Therefore, the following filtering heuristic is proposed. First, attempt to learn a cost function over all timesteps of a single example behavior, using a more complex hypothesis space than intended for the final cost function, and identify timesteps whose penalties are statistical outliers. As these timesteps are inconsistent within an overly complex hypothesis space, there is evidence that the inconsistency is in the example and not for lack of expressiveness in the cost function. Therefore, these timesteps can be removed. This process can be repeated for each example behavior, and only remaining timesteps used in the final training. Experimental results of this filtering approach are presented in the next section.

5 Experimental Results

Our imitation learning approach was implemented for the Crusher autonomous platform (Figure 1) [18]. Crusher's base perception system consists of 6 LiDAR scanners and 4 sets of cameras which are processed to provide a discretized set of geometric and semantic features at a range of up to 20 m (Figure 4). Training data in the form of expert example behaviors was gathered by having Crusher's safety operator teleoperate the vehicle through sets of waypoints. Different training examples were collected over a period of months in varying locations and weather conditions, and with 3 different operators at one time or another. During data collection, all raw sensor data was logged along with the example path. Perceptual features were then produced offline by feeding the raw sensor data through Crusher's perception software. In this way, the base perception system and its features can be modified and improved without having to recollect new training data; the raw data is just reprocessed, and a cost function learned for the new features. Figure 4 provides a simple example of both perceptual features and the resulting (learned) cost.

This set of examples was used to perform a series of offline experiments to validate additions to the MMP framework. The average loss over a path (as defined in Equation 1) was used to evaluate the performance of different variations of the LEARCH algorithm on a large validation set of example behaviors. These results are presented in Figure 5. Fig. 5(a) demonstrates the effectiveness of performing balanced as opposed to unbalanced regression, as the balanced version has superior validation performance. Fig 5(b) demonstrates the benefit of replanning with corridor constraints. With a small corridor size, the algorithm is able to smooth out some of the noise in demonstrated behavior, and improve generalization. As the corridor



(a) Left Camera Image

(b) Right Camera Image

(c) Object Height

(d) Object Density

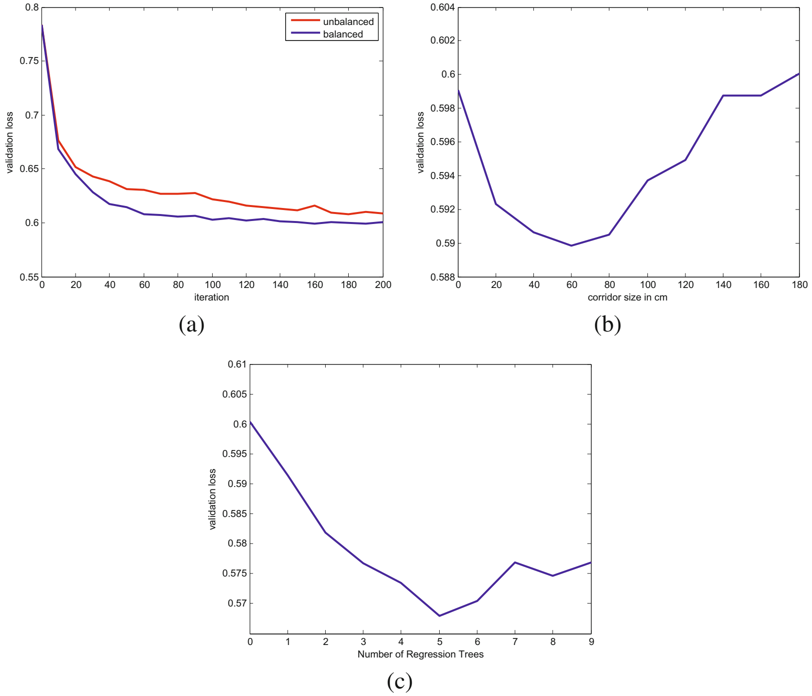
(e) Solidness

(f) Learned Cost

Fig. 4 An example of learned perception cost. A simple scene depicted in (a),(b) results in perceptual features (a small subset shown in (c) - (e)) that are mapped into cost (h).

size increases, the algorithm begins to oversmooth, resulting in decreasing validation performance. In this way, validation data can be used to automatically determine the optimal corridor size. An experiment was also performed to assess the effectiveness of filtering. A single expert behavior was used to learn a cost function, first with no filtering, and then with approximately 10% of its timesteps automatically filtered. Without filtering, the training loss (computed only over unfiltered timesteps) and validation loss were 0.532 and 0.596 respectively. With filtering, these numbers improved to 0.493 and 0.582, indicating further enhanced generalization.

As cost evaluations must be performed online in real time, the computational cost of an evaluation is an important consideration. As a sum of linear functions is another linear function, a linear hypothesis space is computationally advantageous. However, this negates one of the primary benefits of LEARCH (nonlinear cost functions). Therefore, additional experiments were performed with the approach used in [15]: linear regression was used to increase efficiency and generalization, but a feature learning phase was added. Occasional projections of the functional gradient are performed with simple regression trees; these regression trees are then used as a new feature instead of being added directly to the cost function. In this way, future learning iterations can control the contribution of each regression tree feature, and the computational cost of an evaluation is minimized. This approach is similar to the way cost functions are often engineered, with specific rules added to handle special cases. Figure 5(c) shows validation loss as a function of the number of added regression tree features. At first, additional features improve the validation performance; eventually too many features results in overfitting.



(a) (b)

(c)

Fig. 5 Results of offline experiments. **(a)** Validation Loss during learning for the balanced and unbalanced functional gradient projection **(b)** Validation Loss as a function of the replanning corridor size **(c)** Validation Loss as a function of the number of regression trees.

The training set was also used to learn a cost function to run onboard Crusher, which uses a hybrid global and local planning system similar to [7]. Details of the implementation of LEARCH with this planning system can be found in [16]. Originally, the function mapping perceptual features to costs onboard Crusher was hand engineered. This cost function was developed and continually retuned by multiple students, staff, and faculty; this process required hundreds of man-hours over a period of more than 3 years to achieve and maintain a high level of autonomous performance [18]. Much of this maintenance involved retuning the cost function in newly encountered regions of the perceptual feature space observed when first testing in novel terrain (while still trying to maintain good performance in previously encountered terrain). In contrast, the entirety of the training set used for imitation learning involved less than 1 hour of expert driving examples, over a total distance of less than 3 km. This seemingly small amount of human demonstration is sufficient due to the numerous constraints implied by each example behavior: a few kilometers of demonstration provides hundreds of thousands of examples of states to traverse, and millions more examples of states that were avoided. Additionally, if testing in a novel environment demonstrates the existing training set to be insufficient, adapting

Table 1 Per-waypoint averages of various metrics of autonomous performance, demonstrating the difference between operating with an engineered and a learned cost function. The difference in metrics related to efficiency of autonomous traverse are statistically significant, while those related to autonomous vehicle safety are not. Statistics for additional metrics are provided in [16].

Statistic for Comparison	Engineered	Learned	P-value
Avg. Distance Per Waypoint(m)	130.7	124.3	0.07
Avg. Cmd. Vel.($\frac{m}{s}$),Ang. Vel.($\frac{\circ}{s}$)	3.24,6.55	3.39,5.08	0.15,0.0
Avg. Lateral Vel./Accel.($\frac{m}{s^2}$)	0.181/1.34	0.17/1.31	0.0/0.0
Avg. Roll/Pitch($\circ$)	4.05/2.21	3.90/2.18	0.99/0.57
Avg. Slip Ratio	1.131	1.129	0.81
Interventions (Per Waypoint)	8 (0.027)	10 (0.034)	0.48

to the new terrain requires only the collection of a few additional minutes of expert demonstration, and then no further human involvement.

The effectiveness of learned cost functions was validated during hundreds of kilometers of autonomous traverse through highly varying terrains across the continental U.S. During these trials, Crusher used learned cost functions to interpret both onboard and prior overhead perceptual data [15], and reliably navigate between widely spaced waypoints. Additionally, a large set of direct comparison trials were performed to evaluate the online performance of a learned cost function in contrast to the engineered one. During nearly 40 km of autonomous navigation with each cost function, a variety of statistics describing each run were produced (unlike many other comparisons of mobile robot performance, cost itself cannot be used). Table 1 lists some of these statistics and the significance of the difference between systems (treating each waypoint to waypoint traverse as an independent trial). More detailed statistics from these experiments can be found in [16].

The biggest difference in performance when using the two cost functions was with respect to turning: the engineered system was more apt to turn harder to avoid perceived hazards, resulting in a slower traverse and more lateral motion. The effect of this difference on the number of safety interventions and other proprioceptive measures of incurred hazards was not statistically significant (implying that the engineered function may produced more false positives). Therefore, we claim a similar high level of autonomous performance between the two costing approaches, but with orders of magnitude less human involvement in the learned system.

6 Conclusion

This paper addresses the task of interpreting perceptual data for use in autonomous navigation. We have presented a dynamic imitation learning approach that achieves equivalent performance to human engineering with far less manual interaction. The approach easily adapts to new perceptual features, without the need for additional human involvement. Future work will explore the application of this approach to learning cost functions over full state-action pairs. The behavior of an autonomous navigation system is defined not only by which terrain it prefers, but by which motions it prefers (e.g. minimum curvature); by learning all preferences at once from human demonstration, we hope to further improve the robustness of autonomous navigation. Additionally, the use of active learning to solicit expert demonstration will be investigated. Finally, the combination of our approach with self-supervised learning will be explored, to create mobile systems which are trained once and improve with experience.

This work was sponsored by DARPA under contract ”Unmanned Ground Combat Vehicle - PerceptOR Integration” (contract number MDA972-01-9-0005). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government.

References

1. Abbeel, P., Dolgov, D., Ng, A.Y., Thrun, S.: Apprenticeship learning for motion planning with application to parking lot navigation. In: Conference on Intelligent Robots and Systems (2008)
2. Abbeel, P., Ng, A.Y.: Apprenticeship learning via inverse reinforcement learning. In: International Conference on Machine learning (2004)
3. Brooks, C., Iagnemma, K.: Self-supervised classification for planetary rover terrain sensing. In: IEEE Aerospace Conference (2007)
4. Goldberg, S.B., Maimone, M.W., Matthies, L.: Stereo vision and rover navigation software for planetary exploration. In: IEEE Aerospace Conference Proceedings, vol. 5, pp. 2025–2036 (2002)
5. Howard, A., Seraji, H.: Vision-based terrain characterization and traversability assessment. Journal of Robotic Systems 18 (2001)
6. Kalman, R.: When is a linear control system optimal? Trans. ASME, J. Basic Engrg. 86, 51–60 (1964)
7. Kelly, A., Amidi, O., Happold, M., Herman, H., Pilarski, T., Rander, P., Stentz, A., Vallidis, N., Warner, R.: Toward reliable autonomous vehicles operating in challenging environments. In: International Symposium on Experimental Robotics (ISER), Singapore (June 2004)
8. Kim, D., Sun, J., Oh, S.M., Rehg, J.M., Bobick, A.F.: Traversability classification using unsupervised on-line visual learning. In: IEEE Conference on Robotics and Automation (2006)
9. LeCun, Y., Muller, U., Ben, J., Cosatto, E., Flepp, B.: Off-road obstacle avoidance through end-to-end learning. Advances in Neural Information Processing Systems 18 (2006)
10. Nabbe, B., Kumar, S., Hebert, M.: Path planning with hallucinated worlds. In: Proceedings: IEEE/RSJ International Conference on Intelligent Robots and Systems (2004)
11. Pomerleau, D.: Alvinn: An autonomous land vehicle in a neural network. Advances in Neural Information Processing Systems 1 (1989)
12. Ratliff, N., Bagnell, J., Zinkevich, M.: Maximum margin planning. In: International Conference on Machine Learning (2006)
13. Ratliff, N., Bradley, D., Bagnell, J., Chestnutt, J.: Boosting structured prediction for imitation learning. Advances in Neural Information Processing Systems 19 (2007)

14. Ratliff, N.D., Silver, D., Bagnell, J.A.: Learning to search: Functional gradient techniques for imitation learning. Autonomous Robots 27(1), 25–53 (2009)
15. Silver, D., Bagnell, J.A., Stentz, A.: High performance outdoor navigation from overhead data using imitation learning. In: Proceedings of Robotics Science and Systems (2008)
16. Silver, D., Bagnell, J.A., Stentz, A.: Applied imitation learning for autonomous navigation in complex natural terrain. In: Field and Service Robotics (2009)
17. Singh, S., Simmons, R., Smith, T., Stentz, A., Verma, V., Yahja, A., Schwehr, K.: Recent progress in local and global traversability for planetary rovers. In: IEEE Conference on Robotics and Automation (2000)
18. Stentz, A., Bares, J., Pilarski, T., Stager, D.: The crusher system for autonomous navigation. In: AUVSIs Unmanned Systems (2007)
19. Wellington, C., Stentz, A.: Online adaptive rough-terrain navigation vegetation. In: Proceedings of the IEEE International Conference on Robotics and Automation (2004)

An Inverse Optimal Control Approach to Human Motion Modeling

Katja Mombaur, Jean-Paul Laumond, and Anh Truong

Abstract. In this paper, we present inverse optimal control as a promising approach to transfer biological motions to humanoid robots. Inverse optimal control serves to identify the underlying optimality criteria of human motions from measurements. Based on these results optimal control models are established that can be used to control robot motion. Inverse optimal control problems are hard to solve since they require the simultaneous treatment of a parameter identification problem and an optimal control problem. We propose a bilevel approach to solve inverse optimal control problems which efficiently combines a direct multiple shooting technique for the optimal control problem solution with a derivative free trust region optimization technique to guarantee the match between optimal control problem solution and measurements. We apply inverse optimal control to determine optimality principles of human locomotion path generation to given target positions and orientations, using new motion capture data of human subjects. We show how the established optimal control model can be used to enable the humanoid robot HRP-2 to autonomously generate natural locomotion paths.

1 Introduction

1.1 Inverse Optimal Control: What Is the Optimization Criterion of Human Motion?

It is a very common assumption in bionics and biomechanics that natural structures and processes are optimal. This is also true for many forms of human and

Katja Mombaur · Jean-Paul Laumond · Anh Truong
LAAS-CNRS, Université de Toulouse, 7 ave du Colonel Roche, 31077 Toulouse, France
e-mail: {kmombaur,jpl,atruong}@laas.fr

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 451–468.
springerlink.com

animal motion such as locomotion [Alexander (1984), Alexander (1996)]. However, the specific optimization criterion applied to a particular motion is very often not known. But is is possible to observe the results of this natural optimization process by measurements, such as motion capture, EMG etc.

From a mathematical perspective, the generation of motions of animals and humans can be formulated as optimal control problem. Optimal control problems are a special type of optimization problems where the unknown variables are not represented by a simple *n*-dimensional vector, but by *n* unknown functions in time, more specifically, unknown input or control functions, and unknown state functions. A dynamical model establishing the relationship between control and state functions represents a constraint of the optimal control problem. The objective function (which is also calles the cost function) may depend on both, control and state functions, as well as on time. For a classical (forward) optimal control problem the full problem formulation including objective function, model etc. is known and the solution has to be determined.

But as stated above, we are often facing the opposite problem, namely that the exact objective function is not known, but instead we know the solution to this problem, or at least its observable part, from measurements. This type of problem is called an inverse optimal control problem (compare fig. 1). These problems are much harder to solve than (forward) optimal control problems. Inverse optimal control problems are also much more difficult than standard identification problems since optimization and data fitting have to be handled simultaneously.

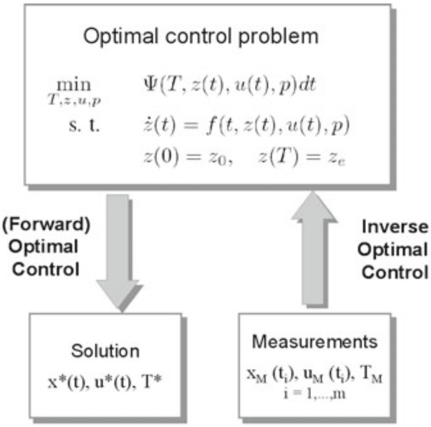

Optimal control problem
$\min_{T,z,u,p} \quad \Psi(T, z(t), u(t), p)dt$
s. t. $\dot{z}(t) = f(t, z(t), u(t), p)$
$z(0) = z_0, \quad z(T) = z_e$
(Forward) Optimal Control
Inverse Optimal Control
Solution
$x^*(t), u^*(t), T^*$
Measurements
$x_M(t_i), u_M(t_i), T_M$
$i = 1,\dots,m$

Fig. 1 (Forward) Optimal control problems vs. Inverse optimal control problems

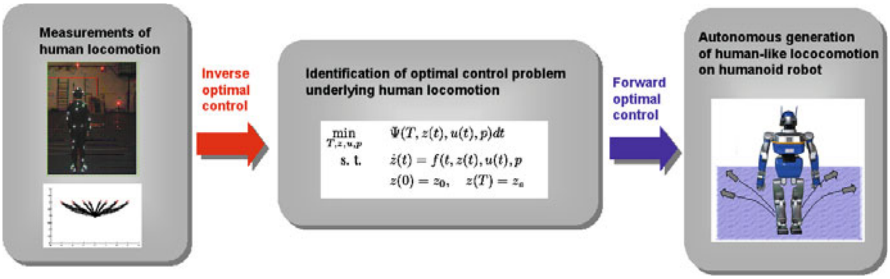

Measurements of human locomotion
Inverse optimal control
Identification of optimal control problem underlying human locomotion
$\min_{T,z,u,p}$ $\Psi(T, z(t), u(t), p)dt$
s. t. $\dot{z}(t) = f(t, z(t), u(t), p$
$z(0) = z_0, \quad z(T) = z_e$
Forward optimal control
Autonomous generation of human-like lococomotion on humanoid robot

Fig. 2 The inverse optimal control approach helps (a) to understand optimality of human locomotion and (b) to generate natural humanoid locomotion

1.2 Natural Humanoid Locomotion by Inverse Optimal Control

We consider the understanding of the optimality principles of human locomotion as one of the keys to generate biologically inspired locomotion on autonomous robots. In fig. 2, we give an overview of the inverse optimal control approach that we propose in this paper. It basically consists of three steps: (a) identification of human optimality criteria for locomotion by inverse optimal control from motion capture measurements, (b) formulation of the full (forward) optimal control model, and (c) implementation and solution of optimal control problem on humanoid robot. This approach enables a humanoid robot to autonomously generate its natural locomotion trajectory to any requested target.

Human and humanoid locomotion can be investigated on different levels. Most research on humanoid robot locomotion aims at generating trajectories on the joint level. The straight or bent path on the floor along which the humanoid robot is supposed to move, is prescribed for this purpose. The study of the selection and optimal generation of this overall path has however been widely neglected in humanoid robotics so far. If humans are asked to walk towards a given end position and orientation in an empty space with no obstacles, they will select a very specific path, out of an infinite number of possibilities. In the attempt to control humanoids in a biologically inspired manner, it would be desirable to understand and imitate that behavior of humans.

In this paper, we show how the inverse optimal control approach is used to generate natural overall locomotion trajectories from an initial rest position and orientation to a given target rest position and orientation. For this purpose, we are not interested in studying the individual trajectories of all joints. Instead, the locomotor system can be described by its overall position and orientation in the plane. However, inverse optimal control problems are prevalent and can basically be found everywhere in natural sciences, and the proposed approach is very general. Consequently, the approach can also be used to analyze motions on joint level.

1.3 Related Work

Classical imitation problems have been widely studied for humanoid robots, leading to impressive results (e.g. [Nakazawa et al (2002)], [Ikeuchi (2009)], [Suleiman et al (2008)] and [Billard and Mataric (2001)]). The task here consists in reproducing a human movement within the kinematic and dynamic ranges of a robot, using different approaches for model identification, such as learning techniques or optimization, but the underlying optimality principles of the motions are not at all investigated. However, the simultaneous treatment of an imitation problem and an identification of optimality principles - i.e. the ”inverse optimal control problem” discussed above - is much more difficult and has not yet been extensively investigated.

[Liu et al (2005)] present a realistic generation of character motion by physics-based models. In their case, the objective function is assumed to be known (minimization of joint torques), but they identify other unknown model parameters from measurement sequences by a nonlinear inverse optimization technique. Heuberger provides a detailed overview of inverse optimization for the different class of combinatorial problems [Heuberger (2004)]. Inverse optimal control problems formulated as bilevel problems can also be treated as MPEC (Mathematical programs with equilibrium constraints). Here the optimal control problem is replaced by the corresponding first order optimality conditions which become constraints of the parameter estimation problem (see the book [Luo et al (1996)]). There is theoretical research on MPECs and corresponding optimality conditions and constraints qualifications (e.g. [Ye (2005)]) but the approach is very difficult to implement in practice. In a recent thesis, it has been applied to very simple dynamical models [Hatz (2008)].

For mobile wheeled robots, the problem of generating the overall path (i.e. the trace on the floor) has been extensively studied (see e.g. [Latombe (1991)], [Laumond (1998)], [LaValle (2006)]). The focus here was on finding a feasible - not an optimal - path in the presence of many obstacles, and no biological inspiration was required in this case. A mobile robot is generally performing nonholonomic movements, i.e. the direction of motion depends on the orientation of the robot or of its wheels.

In humanoid robot research, several authors have studied real time path planning and adaption based on sensor information, looking at the same time at the shape of the path and an appropriate choice of footholds (e.g. [Stasse et al (2006), Chestnutt et al (2005), Gutmann et al (2005), Yoshida et al (2008)]). The problem of natural off-line locomotion path planning has not yet received much attention. In [Mombaur et al (2008)], we have proposed a heuristic optimal control model

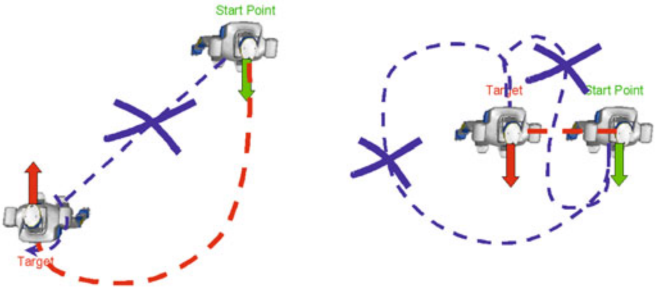


Fig. 3 Natural locomotion paths for humanoid robots: Examples of realistic and unrealistic paths (red dashed lines vs. blue dashed lines) for two different targets

to autonomously generate naturally shaped locomotion paths for humanoid robots. [Choi et al (2003), Pettré et al (2003), Brogan and Johnson (2003)] have studied offline planning for biped locomotion in computer graphics.

From a biological perspective, the shape of human locomotion paths has been investigated, e.g. [Hicheur et al (2007)]. In particular it has been shown that human locomotion in many cases is nonholonomic just as wheeled motion, i.e. people tend to move in forward direction rather than sidewards ([Arechavaleta et al (2008b), Arechavaleta et al (2008a), Laumond et al (2007)]. This general preference may easily be understood from the human anatomy. On the other hand, there are certain situations in which humans naturally tend to abandon the nonholonomic behavior and to include sideward or oblique steps in the locomotion, i.e. move in a holonomic way. This obviously occurs when obstacles must be avoided, but also in the case of very close goals without obstacles (compare fig. 3, right part). In [Mombaur et al (2008)], we have made a first attempt to establish a model that continuously selects between holonomic and nonholonomic locomotion in a realistic way.

1.4 Contribution of This Article

The first contribution of this paper is to propose inverse optimal control as a general approach to transfer biological motions to robots. Inverse optimal control not only helps to understand the underlying optimization objectives of recorded biological motion. It also leads to the generation of mathematical forward optimal control models that can be applied to control humanoid robot motions in a natural way. In this paper, we describe the general form of inverse optimal control problems as well as a very flexible numerical technique for their solution.

The second contribution of this article is to present an example of a successful application of inverse optimal control: a unique optimal control model of the overall locomotion path generation to close targets - i.e. to given final position and orientation, while zero speed is requested at start and end time. In our previously mentioned research [Mombaur et al (2008)], a qualitative model was created and parameters were selected by manual tuning. In contrast to this, the goal in the present paper was to truly identify the weights of the proposed optimal control model from human motion capture data. In addition, based on the inverse optimal control results, we could even simplify the previous formulation by establishing a unique model with constant weight factors that is valid for a whole domain of targets.

This paper is organized as follows: In section 2, we present the general inverse optimal control problem formulation as well as a general numerical solution technique. In section 3, we show how inverse optimal control has been applied to generate natural human-like locomotion paths by outlining the whole sequence from motion capture experiments over model identification to implementation on the humanoid robot HRP-2. In the final section, we summarize results and discuss future research.

2 Inverse Optimal Control: A General Approach to Understand Natural Processes

The goal of inverse dynamic problems is to determine the formulation of an optimal control problem - and in particular its cost function - that is able to best reproduce the available experimental data. In this section, we present the problem statement of inverse optimal control problems, as well as a newly developed numerical solution technique.

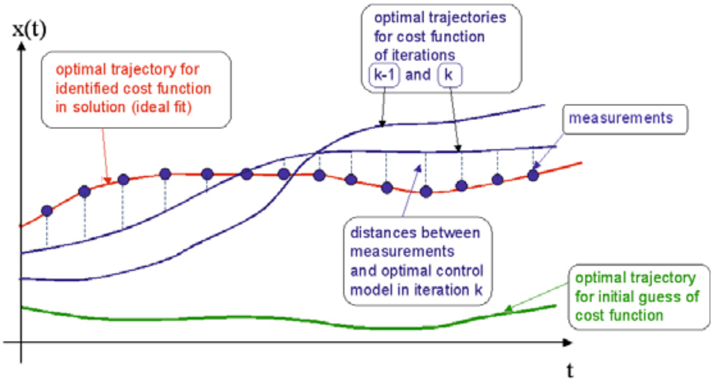

x(t)
optimal trajectory for identified cost function in solution (ideal fit)
optimal trajectories for cost function of iterations k-1 and k
measurements
distances between measurements and optimal control model in iteration k
optimal trajectory for initial guess of cost function
t

Fig. 4 Goal of inverse optimal control: identify cost function that best approximates measured data

2.1 Formulation of Inverse Optimal Control Problem

An inverse optimal control problem consists in determining the function $\Phi(x(t),u(t))$ in the objective function (1) of the following optimal control problem

$$\min_{x(\cdot),u(\cdot),T} \int_0^T \Phi(x(t),u(t))dt \tag{1}$$

$$\text{s. t.}\quad \dot{x} = f(t,x(t),u(t)) \tag{2}$$

$$x(0) = x_0, \quad x(T) = x_e \tag{3}$$

when its solution is known. $x(t)$ are the state variables and $u(t)$ the control variables. The dynamic model (2), and initial and final conditions (3) are assumed to be known. We assume that the solution $x^*(t) \in \mathbb{R}^{n_x}$, $u^*(t) \in \mathbb{R}^{n_u}$ is not known continuously, but only at m evenly space points. In many practical cases, not the full solution is observable, and only some components of the optimal states and controls $x^*_{red}(t) \in \mathbb{R}^{n_{xr}}$ and $u^*_{red} \in \mathbb{R}^{n_{ur}}$ (where $0 < n_{xr} < n_x$ and $0 < n_{ur} < n_u$) are known at m discrete points.

The inverse optimal control problem consists in determining the exact objective function $\Phi(\cdot)$ that produces the best fit to the measurements in the least squares sense (compare fig. 4). We make the basic assumption that the objective function can be expressed as a weighted sum of a series of base functions $\phi_i(t)$ with corresponding weight parameters α_i:

$$\Phi(x(t),u(t),\alpha) = \sum_{i=1}^{n}\left[\alpha_i \int_0^T \phi_i(x(t),u(t))dt\right] \tag{4}$$

The problem of determining the objective function $\Phi(\cdot)$ resulting in the best approximation thus is transformed into the problem of determining the best weight factors α_i.

The base functions $\phi_i(x(t),u(t))$ describe reasonable potential components of the objective function in the given situation. It is important to choose a non-redundant set of objective functions since different base functions leading to exactly the same behavior would be impossible to identify.

In a combined objective function only the relative and not the absolute size of the weight factors counts. If a weight α_i is large, the corresponding term has a big effect on the overall sum and therefore is more likely to be reduced in the overall context. If α_i is small (in the extreme case zero) the term has little (or no) influence on the objective function and the quantities can become large without doing much harm.

With this parameterization of the objective function (4), we can formulate the inverse optimal control problem as bilevel problem:

$$\min_{\alpha} \quad \sum_{j=1}^{m} ||z^*(t_j;\alpha) - z_M(t_j)||^2 \tag{5}$$

where $z^*(t;\alpha)$ is the solution of

$$\min_{x,u,T} \quad \int_0^T \left[\sum_{i=1}^{n} \alpha_i \phi_i(x(t),u(t)) \right] dt \tag{6}$$

$$\text{s. t.} \quad \dot{x} = f(t,x(t),u(t)) \tag{7}$$

$$x(0) = x_0, \quad x(T) = x_e \tag{8}$$

The vector z stands for the full or reduced vector of states and controls, $z(t)^T = (x(t)^T, u(t)^T)$ or $z(t)^T = (x_{red}(t)^T, u_{red}(t)^T)$, depending on the case treated, i.e. the available measurements. z_M denotes the measured values.

2.2 Numerical Solution of Inverse Optimal Control Problems

In this section we present a pragmatic numerical approach to the solution of inverse optimal control problems treated as bilevel problems (compare fig. 5). The upper level handles the iteration over the objective function parameters α such that the fit between measurements and optimal control problem solution is improved. Each upper level iteration includes one call to the lower level where a forward optimal control problem is solved for the current set of α_i. The optimal solution of this problem is then communicated back to the upper level such that the least squares fit between measurements and computations can be evaluated.

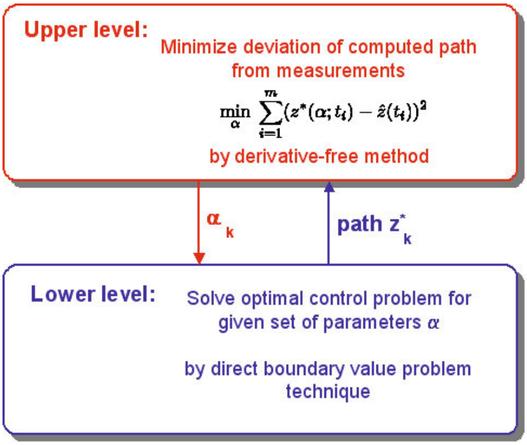

Upper level:
Minimize deviation of computed path from measurements
$\min_{\alpha} \sum_{i=1}^{m} (z^*(\alpha; t_i) - \hat{z}(t_i))^2$
by derivative-free method
α_k
path z^*_k
Lower level:
Solve optimal control problem for given set of parameters α
by direct boundary value problem technique

Fig. 5 Solution of inverse optimal control problem as bilevel optimization problem

As described in [Mombaur (2009)], we have implemented and tested a method to solve inverse optimal control problems on the basis of two powerful numerical techniques. We propose a combination of efficient direct techniques for the solution of the lower level optimal control problem, and of an efficient derivative-free method for the solution of the upper-level least-squares problem. Both techniques that we have combined in our modular software environment will be briefly described in this section.

For the solution of the lower level optimal control problem we have applied the highly efficient direct boundary value problem approach using multiple shooting developed by Bock and co-workers (MUSCOD [Bock and Plitt (1984)] [Leineweber et al (2003)]). The MUSCOD method uses a direct approach (also called a first-discretize-then-optimize approach) to handle control functions. State functions are treated by a multiple shooting technique which transforms the original boundary value problem into a set of initial value problems with corresponding continuity and boundary conditions. The resulting structured nonlinear programming problem (NLP) is solved by a tailored sequential quadratic programming (SQP) algorithm. It is important to note that this approach still includes a simulation of the full problem dynamics on each of the multiple shooting intervals. This is performed simultaneously to the NLP solution using fast and reliable integrators also capable of an efficient and accurate computation of trajectory sensitivity information [Bock (1987)].

For the solution of the upper-level least squares problem, we apply a derivative-free optimization technique, i.e. it only requires function evaluations and does not need derivatives. Derivative-free optimization is always favorable if function evaluations are expensive and noisy and derivative information can therefore not be generated in a reliable manner. In the case of our bilevel problem, each function evaluation of the upper-level problem corresponds to a solution of the lower-level optimal control problem, so it would definitely be difficult to generate numerical derivatives of this function. We only have to handle simple box constraints on the weight parameters in the upper level, all other constraints are handled by the optimal control code in the lower level. We use the newly released derivative-free optimization code BOBYQA [Powell (2008)], which is a very efficient derivative-free optimization technique. It is an extension of Powell's well known code NEWUOA, and can additionally handle simple bounds on the variables. Interpolation-based trust region techniques of derivative-free optimization are used to establish a quadratic polynomial model of the objective function, based on function evaluations only.

As state above, in a combined objective function of an optimization problem only the relative size of parameters matters, not the absolute size. Consequently, the identification of parameters by inverse optimal control is therefore only possible up to a common constant. Our practical way to tackle this issue is to fix one of the parameters a priori to 1.0 and to determine the remaining parameters. If by mistake a parameter that actually should be zero in the solution has been fixed to a nonzero value, a strange behavior of the numerical iterations will be observed, and computations should be repeated with a different parameter fixed.

3 Application of Inverse Optimal Control to Study Human Locomotion

The purpose of this section is to demonstrate how inverse optimal control has been successfully applied to determine optimality criteria of human locomotion. It also describes how this optimal behavior is implemented on a humanoid robot. We are interested in the shape and temporal development of the overall locomotion trajectories, i.e. the traces of the human locomotion on the floor, for given start and end positions and orientations.

3.1 Experiments: Human Locomotion Trajectories

We have performed a series of experiments to capture human locomotion trajectories for given start and end positions and orientations and with zero initial and final speed, in particular to close-by targets in a radius of ~ 3.5 m.

Ten healthy male subjects participated in the experiments, with an average height of $1.77+/-0.06$ m and an average age of $27+/-4$ years. All subjects gave their informed consent to perform the experiments. We have used a Motion Analysis motion capture system with 10 cameras, all with a sampling frequency of 100 Hz.

100 different target scenarios, i.e. different combinations of target positions and orientations, were selected, and randomly ordered, using each scenario twice, resulting in 200 motions preformed by each subject. An arrow on the floor indicated target position and orientation of each trial. The experimental setup is shown in fig. 6.

We were interested in recording the time histories of the overall positions $x(t)$ and $y(t)$ and orientations $\theta(t)$ of the subjects. This could be achieved using two markers on the subjects shoulders, as shown in fig. 6, bottom. The shoulder orientation represents a good simple approximation to the overall orientation of the subject. The subjects were equipped with additional markers, some of which were used to distinguish the two shoulder markers and this correctly identify the forward direction. Since we are interested in measuring the average development of positions and orientations of the subjects, we had to eliminate the natural relative oscillations that occur during a step in forward, sideward and rotational directions. For this, the collected data was filtered to eliminate the step frequency oscillations.

In the experiments we could observe stereotypic behavior of the ten subjects for most of the recorded trajectories. Another publication describing the experiments in more detail and providing a detailed statistical analysis of the collected data is currently in preparation.

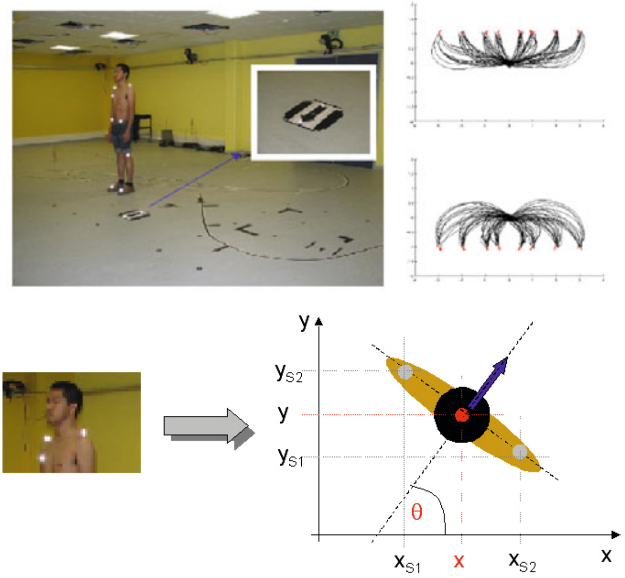


Fig. 6 Motion capture experiments on human locomotion trajectories. Global position and orientation histories of the subjects are determined using markers on the shoulders.

3.2 *An Optimal Control Model of the Human Locomotion Path*

In this section, we present the general formulation of human locomotion as an optimal control model. We give a formulation of the overall locomotion path for rest-to-rest locomotion by differential equations, as well as of a parameterized objective function, using a reasonable set of base functions. The purpose of this optimal control model is not to describe locomotion up to the last detail, but to provide a good description of the essential locomotion objectives.

In this model, we use variables x, y and θ to describe position and orientation of the locomotor system in the global reference frame. For velocities and accelerations, we shift to the human-centered reference frame, since humans do not perceive their movement in a general fixed coordinate system but rather in a local body reference frame. In this system, we can distinguish translational velocities in forward and sideward - called orthogonal - direction, v_{forw} and v_{orth}, as well as rotational velocity ω, and corresponding accelerations which are used as inputs variables u of the optimal control model $u = (u_1, u_2, u_3)^T = (a_{forw}, a_{rot}, a_{orth})^T$.

As described in the introduction, as far as translational motions are concerned, humans in most cases prefer to move in forward direction, and the orthogonal component is zero. Such a motion is called nonholonomic. However in certain situations, orthogonal velocity components appear and locomotion becomes holonomic. For the locomotion trajectories to close-by targets studied here, we expect to observe holonomic motions or at least motion phases.

We therefore use the fully holonomic locomotion model:

$$\begin{aligned}
\dot{x} &= cos\theta\, v_{forw} - \sin\theta\, v_{orth} \\
\dot{y} &= sin\theta\, v_{forw} + \cos\theta\, v_{orth} \\
\dot{\theta} &= \omega \\
\dot{v}_{forw} &= u_1 \\
\dot{\omega} &= u_2 \\
\dot{v}_{orth} &= u_3
\end{aligned} \tag{9}$$

which still contains nonholonomic motions as a special case for $v_{orth} \equiv 0$, i.e. $u_3 \equiv 0$ and $v_{orth}(0) = 0$.

The choice of base functions for the objective function was guided by some intuitive ideas: It is clear that the total time of the path has to be a free variable of the problem and humans will generally prefer faster over slower paths, i.e. tend to minimize total time. Without sudden events, humans tend to perform smooth paths, i.e. large variations of all velocities are avoided, which corresponds to a minimization of accelerations (by magnitude). Motions in forward, orthogonal and rotational direction are clearly judged differently from the subject's perspective and therefore need individual weights. This results in the following basic formulation of the objective function as a combined weighted minimization of total time and the integrated squares of the three acceleration components:

$$\begin{aligned}
\Phi(T, x(t), u(t), p) &= \sum_{i=0}^{3} \left[\alpha_i \int_0^T \phi_i(x(t), u(t)) dt \right] \\
&= \alpha_0 \cdot T + \alpha_1 \int_0^T u_1^2\, dt + \alpha_2 \int_0^T u_2^2\, dt + \alpha_3 \int_0^T u_3^2\, dt
\end{aligned} \tag{10}$$

The objective function is therefore composed of four base functions and has four corresponding weight parameters. In contrast to [Mombaur et al (2008)], where the parameters were determined by manual tuning for a humanoid robot model, we will here use inverse optimal control to properly identify the size of the parameters from human locomotion data. Additionally, in contrast to the qualitative robot model proposed previously, we will show in this paper, that it is not necessary to each time adjust the parameter α_3 according to the distance and orientation change of the target. We will show that is possible to approximate the human behavior in the whole area of close-by targets investigated in the experiments by a unique set of parameters $\alpha_0 - \alpha_3$. The model weights will change for far away targets and long motion segments without any rest position where the motion in general is nonholonomic. So instead of the the continuous model proposed in [Mombaur et al (2008)] we identify here a model which only requires a split into few domains.

3.3 Computational Results: Identification of the Objectives of Human Locomotion

In this section we present computational results of applying inverse optimal control to identify the objective function of problem (4) to match the human locomotion trajectories described in section 3.1. We present numerical evidence to support the hypothesis that locomotion objectives can be approximated by a simple unique model in all of the domain we investigated experimentally.

Concerning the choice of trajectories or trajectory combinations there is of course a wide range of possibilities, due to the large amount of data collected. In this paper we show how five randomly selected locomotion scenarios (a "scenario" is characterized by a target position and orientation) can very well be approximated simultaneously by the same optimal control model, i.e. the same objective function. For each scenario, we use experimental trajectories of five subjects, so the fit was performed over a total of 25 trajectories.

The variable vector z in the bilevel inverse optimal control problem formulation has dimension three: time histories of x, y and θ (i.e. three of the six state variables) are approximated at the same time. Neither velocities (the three remaining state variables) nor accelerations (the three control variables) are directly measured.

As optimal values for the objective function parameters we identified $\alpha^T = (1, 1.139, 0.159, 2.681)$, where α_0 was the parameter fixed a priori. The objective function (4) therefore becomes

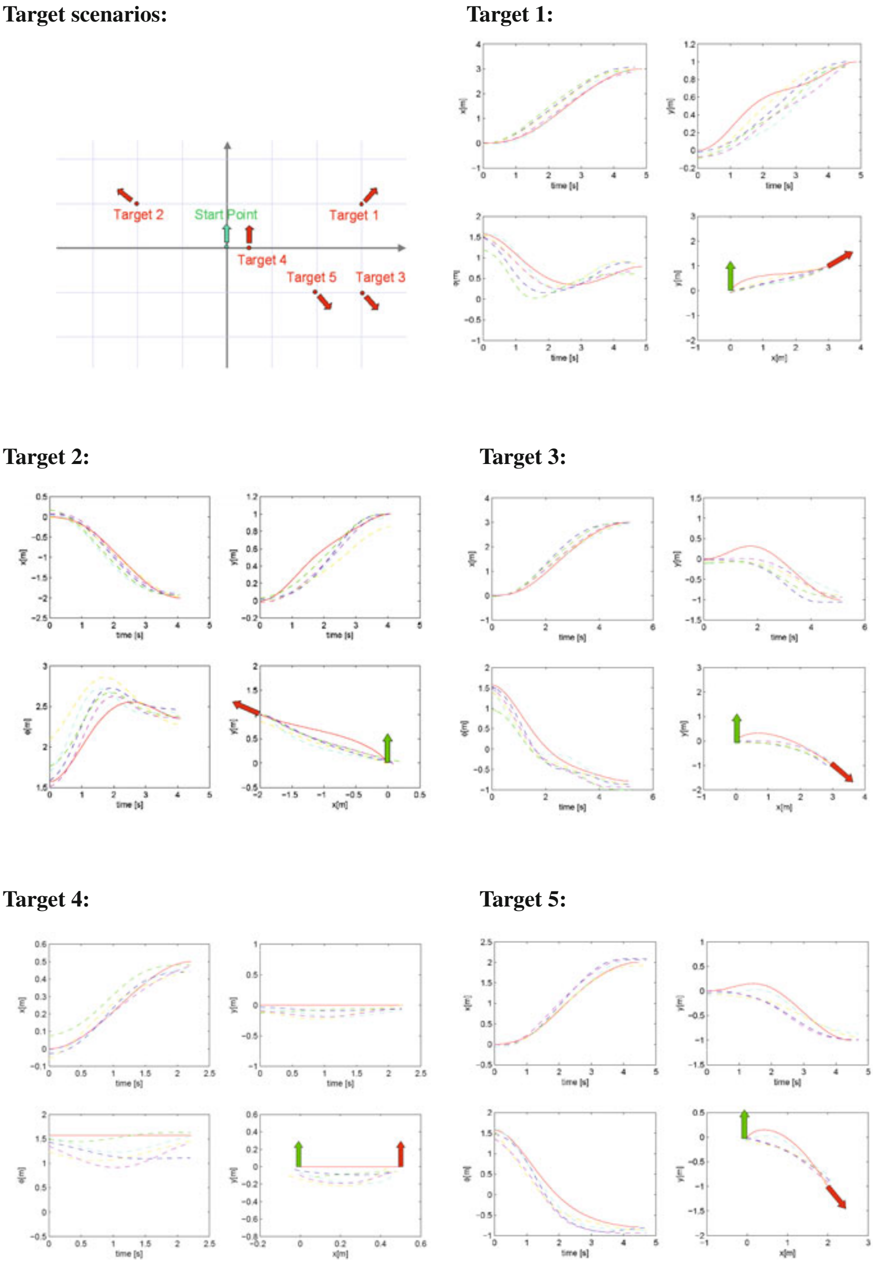
$$\Phi(T, x(t), u(t), p) = T + 1.139 \int_0^T u_1^2 \, dt + 0.159 \int_0^T u_2^2 \, dt + 2.681 \int_0^T u_3^2 \, dt. \quad (11)$$

The weight factor corresponding to the orthogonal direction is about 2.5 the weight factor of the forward direction which leads to a clear preference of forward walking, but leaves the possibility for orthogonal motions whenever they are more efficient in this measure. The weight factor of the rotational term is quite small, i.e. large accelerations in rotational direction are less punished.

The top left part of fig. 7 shows the five arbitrarily chosen scenarios. The other parts of the figure show the results of simultaneous inverse optimal control for all five cases. The red solid line in all sub-figures represents the respective computed optimal trajectory for the identified set of objective function parameters. The five dashed lines denote the measured trajectories of the five subjects used as bases for the computation. The fit in all cases is very good, taking into account that the model equations and optimization functions are always a simplification, and that no perfect fit can be achieved.

3.4 Using Inverse Optimal Control Results to Control Humanoid Robots

In this section we briefly describe how the optimal control model that has been established by inverse optimal control can be used to enable the humanoid robot



Target scenarios:

Target 1:

Target 2:

Target 3:

Target 4:

Target 5:

Fig. 7 Results of inverse optimal control performed simultaneously for five target scenarios. The top left sub-figure presents the five arbitrarily selected scenarios. The other sub-figures show the fit between the measurements (5 dashed lines in each case, representing 5 different subjects) with the respective optimal trajectory (solid red line) produced by the objective function identified by inverse optimal control.

HRP-2 [Kaneko et al (2004)] at LAAS to autonomously generate locomotion trajectories. As described previously, the focus of the presented research is on the generation of bio-inspired overall locomotion trajectories, i.e. the appropriate choice of the trace of the robot on the floor. Our interest here is neither the selection of foot patterns about the path nor the generation of trajectories of all internal joints. For this purpose we rely on existing approaches for the robot HRP-2.

For any given locomotion target to be reached by the humanoid robot, it is now possible to solve the optimal control problem (1) - (3) with the objective function established above in section 3.3. In the optimal control problem formulation, velocity and acceleration bounds are modified to correctly describe the limits of the humanoid robot. The solution of this optimal control problem gives the natural overall path to be followed to the target.

Linear and angular velocities of this computed path are then passed to the pattern generator. We use the walking pattern generator by Kajita et al. [Kajita et al (2003)], which is based on preview control of zero moment point (ZMP) using the table-cart inverted pendulum model, and which produces appropriate footprints and generates a desired ZMP trajectory. Leg joint angles are computed by inverse kinematics from the CoM trajectory and the footprints. The resulting biped walking motion is dynamically stable in the ZMP sense. Fig. 8 shows a visualization of the resulting robot motion for one example, using the humanoid simulator and controller software OpenHRP [Kanehiro et al (2004)] for the humanoid robot HRP-2. Due to identical interfaces of OpenHRP towards simulation and the real robot, the same motions can easily be transferred to the robot.

The computations described above have so far been performed offline. But since computation times are very short - compared to typical delays of humanoid robots - these routines could easily be implemented on the robot and and could be called each time the robot has to autonomously decide about a locomotion trajectory.

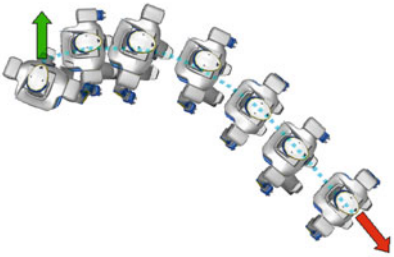


Fig. 8 Implementation of natural locomotion trajectory for target 5 on the humanoid robot HRP-2

4 Conclusion and Future Research

The main purpose of this paper was to present inverse optimal control as a very useful approach to identify underlying optimization objectives of natural processes such as biological motions from experimental data. We have described a flexible numerical approach which allows the solution of inverse optimal control problems for very general problems.

The second purpose of this paper was to propose an optimal control model to describe human locomotion in rest-to-rest motions to close targets. Establishing this simple and unique model was only possible using inverse optimal control. According to our computations it seems to represent a good approximation of the collected locomotion data, and it is very useful to produce natural motions of a humanoid robot.

In any inverse optimal control problem formulation, the selection of appropriate base functions for the objective function is obviously a very crucial element, since any solution of the inverse optimal control problem can only become as good as its base functions permit. For the locomotion study in this paper, we have used a simple objective function based on four elementary functions minimizing total time and accelerations. According to our results this objective function, with properly identified weight factors, seems to be able to explain much of the observed behavior. It can be expected that there are additional components of minor importance which could however slightly improve the fit. Example base functions that we plan to further investigate are terms related to the respective jerks, to the velocity components, or to energy or variation of energy of the motion. We also will establish the model for far away goals for which previous experience has already shown that the resulting motion is mainly nonholonomic. In addition, we are currently extending our research towards the study of locomotion in the presence of fixed and moving obstacles.

The generality of the presented inverse optimal control approach allows its application to a variety of other problems, such as the identification of optimization criteria of locomotion on the joint level. Based on our previous work on forward optimal control of human-like running motions [Schultz and Mombaur (2008)] [Mombaur (2008)] and the multi-body system models developed in this research, we are currently applying inverse optimal control to identify objective functions of different human running motions based on motion capture data.

Acknowledgements. We thank Hans Georg Bock and Johannes Schlöder, Heidelberg, for allowing us to use the optimal control code MUSCOD. We also thank Michael J.D. Powell, Cambridge, for providing the optimization code BOBYQA. Financial support by the French ANR within the project *Locanthrope* is gratefully acknowledged.

A long version of this article has been submitted to *Autonomous Robots*.

References

Alexander (1984). Alexander, R.M.: The gaits of bipedal and quadrupedal animals. International Journal of Robotics Research 3(2), 49–59 (1984)

Alexander (1996). Alexander, R.M.: Optima for Animals. Princeton University Press, Princeton (1996)

Arechavaleta et al (2008a). Arechavaleta, G., Laumond, J.P., Hicheur, H., Berthoz, A.: On the nonholonomic nature of human locomotion. Autonomous Robots 25(1-2) (2008a)

Arechavaleta et al (2008b). Arechavaleta, G., Laumond, J.P., Hicheur, H., Berthoz, A.: An optimality principle governing human walking. IEEE Transactions on Robotics 24(1), 5–14 (2008b)

Billard and Mataric (2001). Billard, A., Mataric, M.: Learning human arm movements by imitation: Evaluation of a biologically inspired connectionist architecture. Robotics and Autonomous Systems 27(2-3), 145–160 (2001)

Bock (1987). Bock, H.G.: Randwertproblemmethoden zur Parameteridentifizierung in Systemen nichtlinearer Differentialgleichungen. In: Bonner Mathematische Schriften, vol. 183, Universität Bonn (1987)

Bock and Plitt (1984). Bock, H.G., Plitt, K.J.: A multiple shooting algorithm for direct solution of optimal control problems. In: Proceedings of the 9th IFAC World Congress, Budapest, International Federation of Automatic Control, pp. 242–247 (1984)

Brogan and Johnson (2003). Brogan, D., Johnson, N.: Realistic human walking paths. In: Proceedings of International Conference on Computer Animation and Social Agents, New-Brunswick, NJ, USA, pp. 94–101 (2003)

Chestnutt et al (2005). Chestnutt, J., Lau, M., Cheung, G., Kuffner, J., Hodgins, J., Kanade, T.: Footstep planning for the Honda ASIMO humanoid. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Barcelona, pp. 629–634 (2005)

Choi et al (2003). Choi, M.G., Lee, J., Shin, S.Y.: Planning biped locomotion using motion capture data and probabilistic roadmaps. ACM Trans. on Graphics 22(2), 182–203 (2003)

Gutmann et al (2005). Gutmann, J.S., Fukuchi, M., Fujita, M.: Real-time path planning for humanoid robot navigation. In: Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI), Edinburgh, UK, pp. 1232–1237 (2005)

Hatz (2008). Hatz, K.: Estimating parameters in optimal control problems. Master's thesis. IWR, Universität Heidelberg (2008)

Heuberger (2004). Heuberger, C.: Inverse combinatorial optimization: A survey on problems, methods and results. Journal of Combinatorial Optimization (2004)

Hicheur et al (2007). Hicheur, H., Pham, Q.C., Arechavaleta, G., Laumond, J.P., Berthoz, A.: The formation of trajectories during goal-oriented locomotion in humans I: A stereotyped behaviour. European Journal of Neuroscience 27 (2007)

Ikeuchi (2009). Ikeuchi, K.: Dance and robotics. In: Digital Human Symposium (2009)

Kajita et al (2003). Kajita, S. et al. : Biped walking pattern generation by using preview control of zero-moment point. In: Proceedings of IEEE International Conference on Robotics and Automation (ICRA), pp. 1620–1626 (2003)

Kanehiro et al (2004). Kanehiro, F., Hirukawa, H., Kajita, S.: OpenHRP: Open architecture humanoid robotics platform. Int. J. of Robotics Research 23(2), 155–165 (2004)

Kaneko et al (2004). Kaneko, K. et al. : The humanoid robot HRP-2. In: Proceedings of IEEE Int. Conf. on Robotics and Automation (ICRA), pp. 1083–1090 (2004)

Latombe (1991). Latombe, J.C.: Robot Motion Planning. Kluwer Academic Publishers, Dordrecht (1991)

Laumond (1998). Laumond, J.P. (ed.): Robot Motion Planning and Control. LNCIS. Springer, Heidelberg (1998)

Laumond et al (2007). Laumond, J.P., Arechavaleta, G., Truong, T.V.A., Hicheur, H., Pham, Q.C., Berthoz, A.: The words of the human locomotion. In: Proceedings of 13th International Symposium on Robotics Research (ISRR 2007), Springer Star Series (2007)

LaValle (2006). LaValle, S.M.: Planning Algorithms. Cambridge University Press, Cambridge (2006)

Leineweber et al (2003). Leineweber, D.B., Bauer, I., Bock, H.G., Schlöder, J.P.: a) An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process optimization - part I: theoretical aspects. Comput. Chem. Engng. 27, 157–166 (2003a)

Liu et al (2005). Liu, C.K., Hertzmann, A., Popovic, Z.: Learning physics-based motion style with inverse optimization. ACM Transactions on Graphics (SIGGRAPH 2005) 24 (2005)

Luo et al (1996). Luo, Z.Q., Pang, J.S., Ralph, D.: Mathematical Programs with Equilibrium Constraints. Cambridge University Press, Cambridge (1996)

Mombaur (2008). Mombaur, K.: Using optimization to create self-stable human-like running. Robotica (2008)

Mombaur (2009). Mombaur, K.: A numerical inverse optimal control method to identify optimality criteria of dynamical systems from measurements (2009) (preparation)

Mombaur et al (2008). Mombaur, K., Laumond, J.P., Yoshida, E.: An optimal control model unifying holonomic and nonholonomic walking. In: Proceedings of IEEE Humanoids, Daejon, Korea (2008)

Nakazawa et al (2002). Nakazawa, A., Nakaoka, S., Ikeuchi, K., Yokoi, K.: Imitating human dance motions through motion structure analysis. In: Proceedings of IEEE IROS, vol. 3, pp. 2539–2544 (2002)

Nelder and Mead (1965). Nelder, J.A., Mead, R.: A simplex method for function minimization. Computer Journal 7, 308–313 (1965)

Pettré et al (2003). Pettré, J., Siméon, T., Laumond, J.P.: A 2-stage locomotion planner for digital actors. In: ACM Eurographics Symposium on Computer Animation (2003)

Powell (2008). Powell, M.J.D.: Developments of newuoa for unconstrained minimization without derivatives. IMA Journal of Numerical Analysis 28, 649–664 (2008)

Schultz and Mombaur (2008). Schultz, G., Mombaur, K.: Modeling and optimal control of human-like running (2008) (submitted)

Stasse et al (2006). Stasse, O., Davison, A.J., Sellaouti, R., Yokoi, K.: Real-time 3D SLAM for humanoid robot considering pattern generator information. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Bejing, China, pp. 348–355 (2006)

Suleiman et al (2008). Suleiman, W., Yoshida, E., Kanehiro, F., Laumond, J.P., Monin, A.: On human motion imitation by humanoid robot. In: Proceedings of IEEE ICRA (2008)

Ye (2005). Ye, J.: Necessary and sufficient optimality conditions for mathematical programs with equilibrium constraints. Journal of Mathematical Analysis and Applications 307, 350–369 (2005)

Yoshida et al (2008). Yoshida, E., Belousov, I., Esteves, C., Laumond, J.P., Sakaguchi, T., Yokoi, K.: Planning 3-d collision-free dynamic robotic motion through iterative reshaping. IEEE Transactions on Robotics 24(5), 1186–1198 (2008)

Towards Motor Skill Learning for Robotics

Jan Peters, Katharina Mülling, Jens Kober, Duy Nguyen-Tuong,
and Oliver Krömer

Abstract. Learning robots that can acquire new motor skills and refine existing one has been a long standing vision of robotics, artificial intelligence, and the cognitive sciences. Early steps towards this goal in the 1980s made clear that reasoning and human insights will not suffice. Instead, new hope has been offered by the rise of modern machine learning approaches. However, to date, it becomes increasingly clear that off-the-shelf machine learning approaches will not suffice for motor skill learning as these methods often do not scale into the high-dimensional domains of manipulator and humanoid robotics nor do they fulfill the real-time requirement of our domain. As an alternative, we propose to break the generic skill learning problem into parts that we can understand well from a robotics point of view. After designing appropriate learning approaches for these basic components, these will serve as the ingredients of a general approach to motor skill learning. In this paper, we discuss our recent and current progress in this direction. For doing so, we present our work on learning to control, on learning elementary movements as well as our steps towards learning of complex tasks. We show several evaluations both using real robots as well as physically realistic simulations.

1 Introduction

Despite an increasing number of motor skills exhibited by manipulator and humanoid robots, the general approach to the generation of such motor behaviors has changed little over the last decades [1]. The roboticist models the task as accurately as possible and uses human understanding of the required motor skills in order to create the desired robot behavior, as well as to eliminate all uncertainties of the environment. In most cases, such a process boils down to recording a desired

Jan Peters · Katharina Mülling · Jens Kober · Duy Nguyen-Tuong · Oliver Krömer
Max Planck Institute for Biological Cybernetics
Department of Empirical Inference & Machine Learning, 72076 Tübingen, Germany
e-mail: {jrpeters,muelling,kober,duy,oliverkro}@tuebingen.mpg.de

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 469–482.
springerlink.com

trajectory in a pre-structured environment with precisely placed objects. If inaccuracies remain, the engineer creates exceptions using human understanding of the task. Such highly engineered approaches are feasible in highly structured industrial or research environments. However, it is obvious that if robots should ever leave factory floors and research environments, we will need to reduce the strong reliance on hand-crafted models of the environment and the robots. Instead, we need a general framework which allows us to use compliant robots that are designed for interaction with less structured and uncertain environments in order to reach domains outside industry. Such an approach cannot rely solely on human knowledge but instead has to be acquired from and adapted to data generated both by human demonstrations of the skill as well as trial and error of the robot.

The tremendous progress in machine learning over the last decades offers us the promise of less human-driven approaches to motor skill acquisition. However, despite offering the most general methods for data-driven acquisition of motor skills, generic machine learning techniques (which do not rely on an understanding of motor systems) often do not scale into the realt-time domain of manipulator or humanoid robotics due to their high dimensionality. Therefore, instead of attempting to apply a standard machine learning framework to motor skill aquisition, we need to develop approaches suitable for this particular domain. To cope with the complexities involved in motor skill learning, the inherent problems of task representation, learning and execution should be addressed separately in a coherent framework employing a combination of imitation, reinforcement and model learning. The advantage of such a concerted approach is that it allows the separation of the main problems of motor skill acquisition, refinement and control. Instead of either having an unstructured, monolithic machine learning approach or creating hand-crafted approaches with pre-specified trajectories, we are capable of acquiring skills from demonstrations and represented as policies which become refined by trial and error (as discussed in Section 4). Additionally, we can learn how to activate and adapt the task-related parameters in order to achieve more complex tasks as discussed in Section 5. Finally, using learning-based approaches, we can achieve accurate control without accurate analytical models of the complete system as discussed in Section 3.

2 Towards a General Skill Learning Framework

In order to create a motor skill learning framework that is sufficiently general, we need to discuss three basic components for such an approach. For this, *a general representation* is required that can encapsulate elementary and frequently used motions. We need to be able to *learn* these motions efficiently, and a *supervisory module* must be able to use these basic elements. Finally, *execution* is required that can adapt to changes in the environment. The resulting control architecture is shown in Figure 1. Let us now briefly discuss each of these aspects in the remainder of this section.

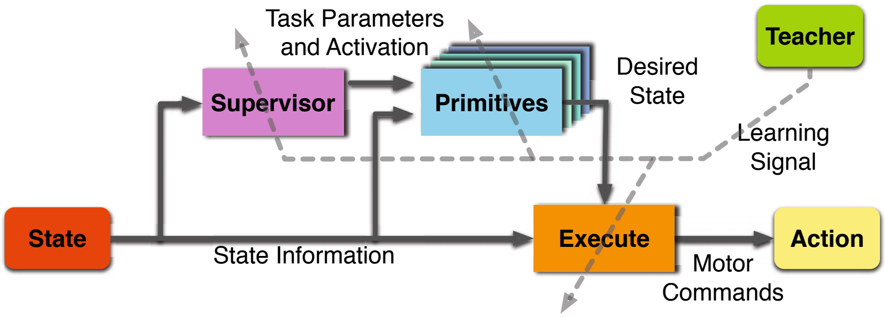


Fig. 1 This figure illustrates the generic components of a motor skill learning system, i.e., the *supervisor system* activates *motor primitives* and sets their task parameters. These elementary movements are *executed* by a learned motor control law. The learning signals are provided with the help of a teacher and may be (a) a demonstration for imitation, (b) a reward or punishment for self-improvement or (c) a model error if it can be observed. In this paper, we focus on learning primitives and execution but also discuss ongoing work on learning to incorporate context in the supervisor layer.

Motor Primitives. For the representation of motor skills, we can rely on the insight that humans, while being capable of performing a large variety of complicated movements, restrict themselves to a smaller amount of primitive motions [3]. As suggested by Ijspeert et al. [4], such primitive movements can be represented by nonlinear dynamic systems. As a result, we may represent elementary tasks by elementary policies of the type[1]

$$\dot{\mathbf{x}}^d = \pi_i(\mathbf{x}^d, \mathbf{x}, t, \rho_i) \tag{1}$$

where $\mathbf{x}^d$ is the internal state of the system, t denotes the time, $i \in \{1,2,\ldots,n\}$ is the index of the motor primitive in a library of movements, and task parameters $\rho_i = [\theta_i, d, \mathbf{g}, \mathbf{A}, \ldots]$ determine the shape of movement primitive i using $\theta_i \in \mathbb{R}^L$, duration d, goal $\mathbf{g}$ and amplitude $\mathbf{A}$, etc, of the motion. The resulting system is linear in the shape parameters θ_i and can therefore be learned efficiently. They are robust towards perturbations and, as they are time-continuous, they are well-suited for control.Both primitives in task-spaces as well as in joint-space can be learned. We have extended Ijspeert et al.'s model [4] so that it may be coupled to additional external variables included in the state $\mathbf{x}$ as discussed in [14]. A key element of the Ijspeert formulation is that the shape is solely determined by θ_i but that it is invariant under changes of duration, goal or amplitude of the movement. Hence, the resulting primitives can be reused efficiently by a higher-level supervisory module.

Supervisor. The supervisory level is an increasingly hot topic for research as it allows the usage of the motor primitive policies π_i in a multitude of novel ways. First, it may reuse a movement primitive with the same shape in various situations by simply modifying the duration, the goal, the amplitude or other task parameters.

[1] Note that Equation (1) is in state-space formulation and, in fact, a second order system.

As we will see in Section 5.1, it is straightforward to *learn subgoal functions* that set the task context variables based on the external state. The supervisory level allows the genereralization of learned movements by creating a *mixture of motor primitives*, i.e., a new movement policy π results from a convex combination of existing movements π_i. In the same context, we can treat the selection of motor primitives. Here, the primitive with the maximal weight is activated while in generalization several primitives using this state-dependent weight. These topics are discussed in Section 5.2. Other tasks of the supervisor are *sequencing* motion primitives as well as *blending* the transitions between them and the *superposition* of different movements.

Execution. The execution of a motor primitive π_i on compliant robot systems, which are safe in the interaction with humans, adds another level of complexity. It requires that we generate motor commands $\mathbf{u} = \eta(\dot{\mathbf{x}}^d, \mathbf{x}^d, \mathbf{x})$ so that the motor primitives get executed precisely while not introducing large feedback gains. If accomplished using hand-crafted control laws, the quality of the analytical models is essential and, low gain control can only be achieved with very accurate models. Hence, in the presence of unmodeled, time-variant nonlinearities resulting from stiction, cable drives, or the hydraulic tubes, it will become essential to learn accurate models and to adapt them online. We are developing efficient real-time regression methods for online model learning based on the state-of-the-art in machine learning, see Section 3.1. If a motor primitive is only acting in a limited subspace, it can often be better to directly learn a mapping from primitives and states to motor command. While learning such an operational space control is no longer a standard regression problem, it can still be solved using a reward-weighted regression when using insights from mechanics.

Learning is required for acquiring and refining the motor primitives discussed before. However, it is also needed for adapting the execution to changes in the environement and to learn the supervisory module, as can be observed in Figure 1. Learning motor primitives is achieved by adapting the parameters θ_i of motor primitive i. The high dimensionality of our domain prohibits the exploration of the complete space of all admissible motor behaviors, rendering the application of many standard machine learning techniques impossible as these require exhaustive exploration. Instead, we have to rely on a combination of imitation and reinforcement learning to acquire motor skills where supervised learning is used to obtain the initialization of the motor skill, while reinforcement learning is used in order to improve it. Therefore, the aquisition of a novel motor task consists out of two phases, i.e., the 'learning robot' attempts to reproduce the skill acquired through supervised learning and then improve the skill from experience by trial-and-error through reinforcement learning. See Section 4 for more details on this part. When learning to execute, we are interested in two topics: learning better models of the robots dynamics in order to improve the model-based control laws of the system (as discussed in Section 3.1), and to directly learn policies that transform task-space motor primitives policies into motor command (see Section 3.2). The supervisory layer poses a variety of learning problems such learning mappings from states to motor primitive task parameters (see Section 5.1), learning activation functions for selection and generalization of motor primitives (see Section 5.2), sequencing, blending and superposition of primitives, as well as parsing longer trajectories into motor primitive automata (see [9]) or determining how many movement primitives might be included in a data set [10].

These components allow us to create a motor skill learning framework in a bottom-up manner wherein we can understand each component well from an analytical robotics point of view.

3 Learning for Control

Bringing anthropomorphic robots into human daily life requires backdrivable robots with compliant control in order to ensure safe interactions with human beings. In contrast, traditional industrial robots employ high control gains which results in an inherent stiffness and, thus, are ill-suited for this aim. To achieve accurate but compliant tracking, it is essential to predict the torques required for the current movement accurately. It is well-known that for sufficiently complex robots (e.g., humanoids, service robots), the standard rigid body dynamics (RBD) models no longer describe the dynamics properly [5], and data-driven approximation methods become a promising alternative. Using modern machine learning techniques has a multitude of advantages ranging from higher precision torque prediction to adaptation with online learning if the dynamics are altered.

In this section, we will discuss two learning-to-control problems, i.e., learning models for control in Section 3.1 and learning operational space control in Section 3.2.

3.1 Learning Models for Control

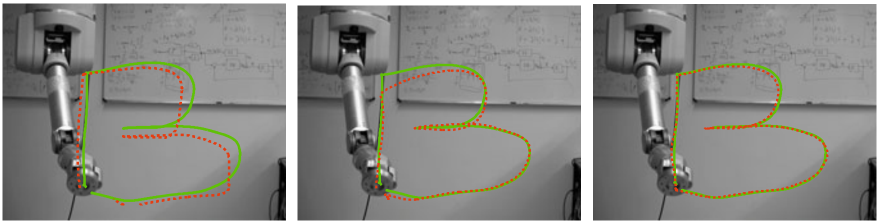
In theory, learning models of the robot dynamics is a straightforward and well-defined regression problem, wherein we can observe joint angles $\mathbf{q}$, joint velocities $\dot{\mathbf{q}}$, joint accelerations $\ddot{\mathbf{q}}$ and motor commands $\mathbf{u}$. We intend to infer the unique mapping $\mathbf{f}$ from state variables $\mathbf{x} = [\mathbf{q}, \dot{\mathbf{q}}]$ and $\dot{\mathbf{x}}$ to motor commands $\mathbf{u}$ of which we have some prior knowledge[2]

$$\mathbf{u} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\dot{\mathbf{q}}, \mathbf{q}) + \mathbf{G}(\mathbf{q}) + \varepsilon(\ddot{\mathbf{q}}, \dot{\mathbf{q}}, \mathbf{q}) = \mathbf{f}(\mathbf{x}, \dot{\mathbf{x}})$$

with mass matrix $\mathbf{M}(\mathbf{q})$, coriolis and centrifugal forces $\mathbf{C}(\dot{\mathbf{q}}, \mathbf{q})$, gravity $\mathbf{G}(\mathbf{q})$ and the unmodeled nonlinearities $\varepsilon(\ddot{\mathbf{q}}, \dot{\mathbf{q}}, \mathbf{q})$.

However, despite being a well-posed problem, and contrary to all progress in machine learning, online learning of robot dynamics still poses a tremendous technical challenge for any learning method. It has to deal with an endless stream of high-dimensional data while learning needs to take place in real-time at sampling rates of

[2] We can in fact straightforwardly use this knowledge as described in [8].



(a) RBD Model (b) Offline Learned Model (c) Online Learned Model

Fig. 2 This figure exhibits the effects of offline and online learning in low-gain control. The green line shows the trajectory of the letter B (previously exhibited by haptic input) as a reference trajectory and the robot is supposed to reproduce this trajectory with reproduction shown as a dashed red line. In (a), a standard control law using an analytical model provided by the manufacturer Barrett is shown. In (b), a full GP has been learned offline from the letter A and now generalizes to the letter B with tracking errors where it lacks data. In (c), local GP (LGP) have been learned based on letter A and improve online while executing letter B. As a result, there is an improved tracking performance.

approximately 100Hz. While modern machine learning approaches such as Gaussian process regression (GPR) and support vector regression (SVR), yield significantly higher accuracy than traditional RBD models, their computational requirements can become prohibitively costly as they grow with number of data points. Thus, it is infeasible to simply use off-the-shelf regression techniques and the development of domain-appropriate versions of these methods is essential in order to make progress in this direction [7].

One possibility for reducing the computational cost is the partitioning of the data such that only the regionally interesting data is included in a local regression and, subsequently, combining these local predictions into a joint prediction. This approach was inspired by LWPR [2], which employs linear models. Using the more powerful Gaussian process models, we can achieve a higher prediction accuracy with less tuning of the algorithm. As a result of the localization and the resulting smaller local models, we can reach a significantly higher learning and prediction speed than for standard kernel regression techniques while having a comparable accuracy. While our approach is not as fast as LWPR, it has a significantly improved prediction accuracy in comparison and requires less manual tuning of the hyperparameters of the algorithm. The resulting method is called *Local GPR* or LGP [6] as it employs Gaussian process regression (GPR) for learning each local model i using

$$\hat{u}_i^i = \mathbf{k}^{iT} \; (\mathbf{K}^i + \sigma_n^2 \mathbf{I})^{-1} \mathbf{U}_i = \mathbf{k}^{i\;T} \alpha_i,$$

where u_j^i is the torque for joint j predicted by model i, $\mathbf{K}^i$ is the kernel matrix with $K_{ml}^i = k(\mathbf{x}_m^i, \mathbf{x}_l^i)$, the kernel vector $\mathbf{k}^i$ with $k_m^i = k(\mathbf{x}, \mathbf{x}_l^i)$ between the new input $\mathbf{x}$ and the stored data points $\mathbf{x}_l$, as kernel k a Gaussian kernel is employed (however, Matern kernels and rigid-body kernels have been used successfully in this context), past actions $\mathbf{U}_i$ and the so-called prediction vector $\mathbf{a}_i$. This prediction vector can be updated incrementally which is computationally feasible as we only have small local models. A weighted average allows the combination of the local models

$$\hat{\mathbf{u}} = \frac{\sum_{i=1}^{n} w_i \hat{\mathbf{u}}_i}{\sum_{i=1}^{n} w_i},$$

where the weights $w_i = \exp(-0.5\sigma_i^{-2} \|\mathbf{x} - \mathbf{c}_i\|^2)$ are used to re-weight the model i in accordance to the proximity of the input $\mathbf{x}$ to the centers of the model $\mathbf{c}_i$.

Due to the reduced computational cost, this approach was successfully implemented on a real Barrett WAM arm where it was able to improve the tracking performance while learning online. When using the learned model in a computed torque setup where the learned model is employed to predict the required torque while stabilized by a linear low-gain control law. It can be shown that the learned model outperforms RBD models and, due to the online improvement, also most global regression techniques. Figure 2 exhibits the difference between these methods. In Figure 2(a), the performance of a low-gain feedback control law with a RBD model is shown for tracking the letter B, Figure 2(b) shows an offline-learned model trained with the letter A tracking letter B and 2(c) shows the improvements due to online-learning. For details on the approach please refer to [6]. Future work will include improvements on the current method, the inclusion of a priori knowledge about the rigid body dynamics into the regression (see [8]) and applications to operational space control, see Section 3.2.

3.2 Learning Operational Space Control

Operational space control (OSC) is one of the most elegant approaches to task control for complex, redundant robots. Its potential for dynamically consistent control, compliant control, force control, and hierarchical control has not been exhausted to date. Applications of OSC range from basic end-effector control of manipulators [18] to balancing and gait execution for humanoid robots [23]. If the robot model is accurately known, operational space control is well-understood and a variety of different solution alternatives are available. However, as many new robotic systems are supposed to operate safely in human environments, compliant, low-gain operational-space control is desired. As a result, the practical use of operational space control becomes increasingly difficult in the presence of unmodeled nonlinearities, leading to reduced accuracy or even unpredictable and unstable null-space behavior in the robot system.

Learning control methods are a promising potential solution to this problem. However, learning methods do not easily provide the highly structured knowledge required in traditional operational space control laws, e.g., Jacobians, inertia matrices, and Coriolis/centripetal and gravity forces, since all these terms are not always instantly observable. They are therefore not suitable for formulating supervised learning as traditionally used in learning control approaches.

We have designed novel approaches to learning operational space control that avoid extracting such structured knowledge as much as ill-posed problems and rather aim at learning the operational space control law directly, i.e., we pose OSC as a direct inverse model learning problem where we acquire an execution policy of the type $\mathbf{u} = \eta(\dot{\mathbf{x}}^d, \mathbf{x}^d, \mathbf{x}, \mathbf{u}_0)$ in which $\mathbf{x}^d = [\dot{\mathbf{p}}^d, \mathbf{p}^d]$ and $\dot{\mathbf{x}}^d$ denote the desired behavior prescribed by the motor primitives in task space while the state $\mathbf{x} = [\dot{\mathbf{p}}, \mathbf{p}, \dot{\mathbf{q}}, \mathbf{q}]$ of the robot is still described by both state-space and task-space components as well as a null-space behavior $\mathbf{u}_0$. Similarly, if we wanted to directly learn the operational space control law as done for model learning in Section 3.1, we would have an ill-posed regression problem as averaging over a non-convex data set is not directly possible. However, the first important insight for this paper is that a physically correct solution to the inverse problem with redundant degrees-of-freedom does exist when learning of the inverse map is performed in a suitable piecewise linear way [19, 20]. The second crucial component for our work is based on the insight that many operational space controllers can be understood in terms of a constrained optimal control problem [18]. The cost function associated with this optimal control problem allows us to formulate a learning algorithm that automatically synthesizes a globally consistent desired resolution of redundancy while learning the operational space controller. From the machine learning point of view, this learning problem corresponds to a reinforcement learning problem that maximizes an immediate reward. We employ an expectation-maximization policy search algorithm in order to solve this problem. Evaluations on a simulated three degrees of freedom robot arm show that the approach always converges to the globally optimal solution if provided with sufficient data [20].

The application to a physically realistic simulator of the anthropomorphic SARCOS Master arm demonstrates feasibility for complex high degree-of-freedom robots. We also show that the proposed method works in the setting of learning resolved motion rate control on a Mitsubishi PA-10 medical robotics arm [22] and a high-speed Barrett WAM robot arm.

The presented approach also allows us to learn hierachies of operational space controllers where a higher level operational space control law i given by $\mathbf{u}_i = \eta(\dot{\mathbf{x}}_i^d, \mathbf{x}_i^d, \mathbf{x}, \mathbf{u}_{i-1})$ is simply fed the output of the next lower-level operational space control law $\mathbf{u}_{i-1}$ as input. This kind of daisy-chaining of learned control laws may in the future allow us to properly solve the problem of superimposing motor primitives.

4 Imitation and Reinforcement Learning with Motor Primitives

Humans and many mammals appear to rely on motor primitives [3] in order to generate their highly agile movements. In many cases, e.g., when learning to play tennis, humans acquire elementary actions from a teacher. This instructor takes the student by the hand and shows him how to perform forehand and backhand swings.

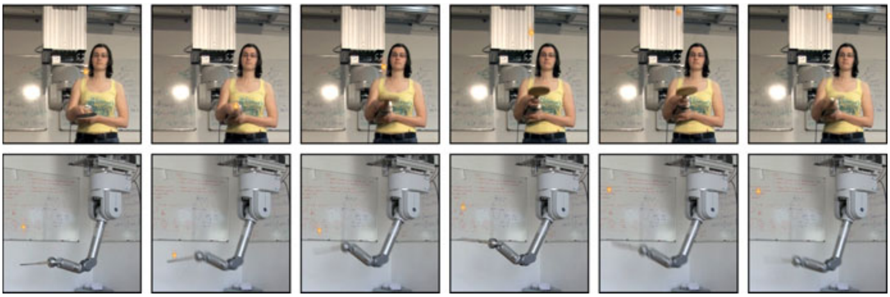


Fig. 3 This figure shows how a ball-on-a-string task can be learned by imitation. The human demonstration presents a rhythmic movement with an initial discrete transient where the generic movement is represented by a rhythmic motor primitive modulated by a discrete motor primitive handling the start-up phase.

Subsequently, the student tries to play by himself and improves as he observes the results of his own successes and failures.

4.1 Imitation with Motor Primitives

When viewed from a probabilistic perspective, imitation learning can be seen as a relatively straightforward problem. When we have observed trajectories $\tau = [\dot{\mathbf{x}}, \mathbf{x}]$ as well as their distribution $p(\tau)$, we will try to reproduce these movements by matching this distribution with a distribution $p_\theta(\tau)$ that is determined by the policy parameters θ. While such a policy can be either deterministic or stochastic, it is often easier to model it as a stochastic policy to take the variation in the data into account.

This policy is represented by a motor primitive modeled by a dynamical system as described by Equation (1). Here, imitation learning reduces to inferring the set of parameters so that the distance $D(p(\tau)||p_\theta(\tau))$ between the observed distribution $p(\tau)$ and the reproduced behavior distribution $p_\theta(\tau)$ is minimized. The Kullback-Leibler divergence is known to be the natural distance measure between probability distributions and is hence employed here.

From this point of view, one can straightforwardly derive regression algorithms such as the ones in [4, 14] to imitate using both the standard formulation of motor primitives [4] as well as the perceptually coupled formulation [14]. As a result, we can learn complicated tasks such as paddling a ball [15] simply by imitation, see Figure 3. This formulation can be made to work both with imitations captured using a VICON setup, see [14], as well as for kinethetic teach-in as in [15].

However, in most real life situations, imitation learning does not suffice and self-improvement is required. E.g., for the Ball-in-a-cup shown in Figure 4, an imitation only suffices for bringing the ball somewhere in the proximity of the cup.

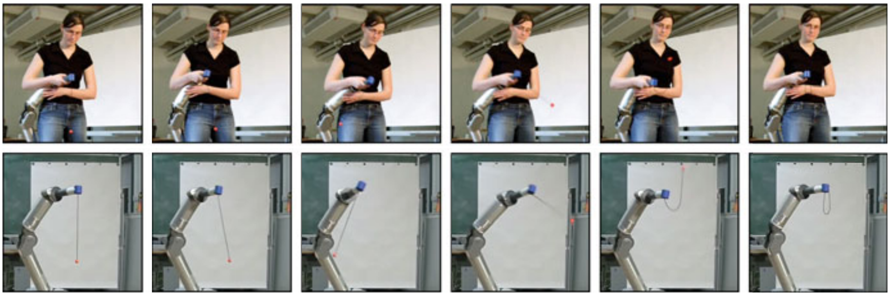


Fig. 4 This figure exhibits the general approach, first, a robot is taught the basic movement which is turned into a motor primitive using imitation learning. Subsequently, reinforcement learning is applied to the problem until the robot obtains a motor primitive policy where it slings the ball perfectly into the cup every single time. The imitation is shown in the upper time series while the optimal learned policy is shown in the lower row.

4.2 Self-improvement by Reinforcement Learning

Reinforcememt learning is in general a much harder problem. Unlike in imitation learning, its focus no longer lies on simply reproducing a presented behavior, but rather on improving a behavior with respect to rewards r. Hence, the system has to try out new actions and, from these actions, infer the policy parameters θ^* that maximizes the expected return

$$J(\theta) = E\left\{\frac{1}{T}R_{1:T}\right\} = E\left\{\frac{\delta t}{d}\sum_{i=1}^{d/\delta t} r_t\right\},$$

where $1/\delta t$ is the sampling rate of the system, d the duration, $T = d/\delta t$ the number of steps and $R_{1:d/\delta t}$ is the return of an episode. In the general setting, reinforcement learning might be an unsolvable problem. Finding a generically optimal policy requires exhaustive try-outs of possible state-action pairs, wherein the number of possibilities grows exponentially with the number of degrees of freedom involved in the task. As anthropomorphic robot exhibit a high dimensionality, they remain beyond the reach of generic reinforcement learning methods.

However, the full reinforcement learning problem appears to be solved rarely in human motor control. For example, olympic high jumper used to refine a variety of different techniques (e.g., straddles, scissor jumps and eastern cut-offs) that all involved running towards the bar and jumping forward. It took until 1968 when the athlete Dick Fosbury accidentally found out that approaching the bar from the side and jumping backwards might be a significantly superior policy. While no reinforcement learning method is in sight that will provide us automatically with such insights, we can design local reinforcement methods that allow us to improve existing policies incrementally. To do so, we rely on obtaining initial parameters θ_0 from an imitation and, subsequently, optimize this policy by self-improvement with respect to the expected return.

Pursuing this type of approach for several years, we have been developing a series of different methods. We originally started out by following the policy gradient approach [12] where the policy improvement is achieved by following the gradient of expected return with respect to its parameters. The resulting update rule can be denoted by

$$\theta_k = \theta_{k-1} + \alpha_k \, \nabla_\theta J(\theta)|_{\theta=\theta_k},$$

where α_k denotes a learning rate at update k and $\nabla_\theta J(\theta)$ is a policy gradient. However, the standard or 'vanilla' policy gradient proved to be suprisingly slow and, thus, not applicable on real robots. It turned out that a covariant or 'natural' policy gradient was able to provide us with the learning speed required for basic motor primitive learning in robotics and we were able to optimize basic movements as well as a T-Ball swing [12]. Nevertheless, the resulting algorithms had open parameter such as the learning rate and the learning process would be too slow for some tasks. As a result, we studied the similarity between expectation-maximization (EM) algorithms and policy gradients. It turned out [11, 19, 20, 13] that as a new cost function we can maximize the distance $D(R(\tau)p(\tau)||p_\theta(\tau))$ between return- or reward-weighted observed path distribution $R(\tau)p(\tau)$ and the new path distribution $p_\theta(\tau)$. This cost function can become part of a lower bound on the expected return $J(\theta)$ and, hence, maximizing it iteratively as in

$$\theta_k = \text{argmax}_\theta \, D(R(\tau)p_{\theta_k}(\tau)||p_\theta(\tau))$$

will at least converge to a locally optimal policy. Such algorithms allow us to show that the problem of policy search can been framed in the parameter estimation setting and, as the similarity to the equations in Section 4.1 makes clear, we have obtained a reward-weighed imitation. At this point, one needs to think about exploration and the type of exploration determines the type of parameter estimation that can be used. For instance, Gaussian exploration with constant variance will result in the reward-weighted regression algorithm [19, 20] and heteroscedastic Gaussian exploration will result in the PoWER algorithm [13].

The PoWER algorithm has been used successfully in a variety of settings, most prominently, it has been able to learn ball-in-a-cup. Here, it started to learn with a policy obtained by imitation that could barely bring the ball into the proximity of the cup. Subsequently, it has learned how to catch the ball in the cup and after less than a hundred trials, it manages to succeed at every trial.

5 Towards Learning the Supervisor

In order to get a step closer to creating complex tasks that require a supervisor, various other topics need to be addressed as already outline in Section 2. We will first discuss two topics where we have made recent progress, i.e., goal learning in Section 5.1, and the mixture of motor primitives in Section 5.2. Further topics for learning the supervisory layer are sequencing, blending and superposition of primitives as well as the parsing of longer trajectories into motor primitive automata (see [9]) or determining how many distinct movement primitives are included in a data set ([10]).

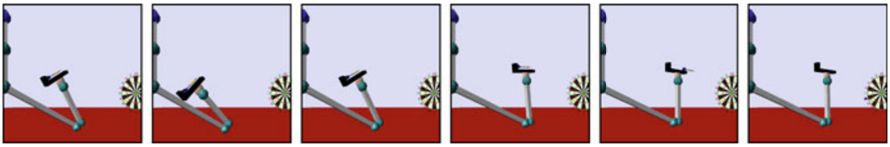


Fig. 5 This figure shows a dart thrown in a physically realistic simulation. Here, the robot is told that the dart should hit a specific target square on the disk and learns to modulate the goal in order to adapt a single motor primitive to many different situations.

5.1 Goal Learning

Previous work in learning for motor primitives has largely focussed on learning the shape parameters θ_i (see Section 4) while duration d, goal $\mathbf{g}$, amplitude $\mathbf{A}$, etc., were simply considered constant parameters optimized along with the shape [12] or set based on an external stimuli [21]. Here, we attempt to learn mappings from the state to these parameters which allow us to take movements of the same shape and use them for various different contexts. Nevertheless, in goal learning, we assume that we have to respond to constantly changing external stimuli, and always adapt the external parameters appropriately. For example, assume that you are playing a dart game where you are told to hit predetermined fields on the dart board in a certain sequence (as in Figure 5). In this case, all movements will simply be slight variations of that same throwing movement and can be represented by the same movement primitive. Hence, the proper way to adapt motor primitive to the square that you intend to hit is by altering its duration d and goal $\mathbf{g}$.

However, in order to learn this dart game faster than can be achieved using the shape parameters, we also need another method. We discovered that this can be achieved using a cost regularized Gaussian process regression.

5.2 Mixture of Motor Primitives

Selection of motor primitives as well as generalization between motor primitives can be achieved using a mixture of motor primitives approach. In such an approach, we have a gating or localization network λ, similar to that in a mixture of experts [16] as part of the supervisor system and activates the right motor primitives. As a result, we obtain a task policy $\mathbf{u} = \pi(\mathbf{x},t)$ that is composed of the n primitives such that

$$\mathbf{u} = \pi(\mathbf{x},t) = \frac{\sum_{i=1}^{n} \lambda_i(\mathbf{x}_0)\pi_i(\mathbf{x},t)}{\sum_{j=1}^{n} \lambda_j(\mathbf{x}_0)}, \tag{2}$$

where $\lambda_i(\mathbf{x}_0)$ denotes the activation of the motor primitive i represented by π_i and $\mathbf{x}_0$ denotes the initial state based upon which of the primitives are activated. A project currently in progress is the learning of table tennis [17] using a mixture of motor primitives (see Figure 6). Here, we currently have achieved already a success rate of 52% of the learned table tennis control law in a ball gun setup and we hope to have a significantly improved setup in the near future.

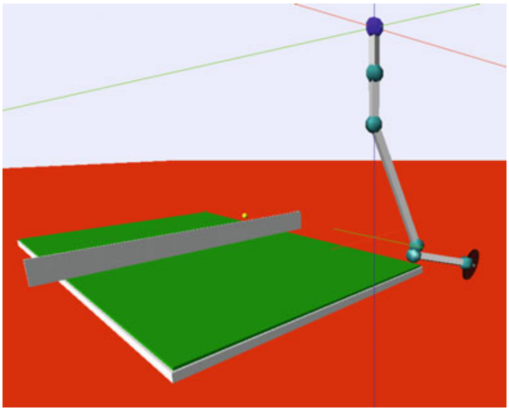


Fig. 6 The mixture of motor primitives is used for the selction and generalization of motor primitives in a table tennis setup.

Using the example of table tennis, we can straightforwardly explain how the mixture of motor primitives is able to generalize between motor primitives. Assume that the system has successfully learned n primitives by imitation observed with different external states $\mathbf{x}_0^i$ (such as a ball position and velocity) and a gating network λ has been obtained. In this case, if a ball is observed at a new initial state $\mathbf{x}_0$, the motor primitives, that resulted in a successful responses to the most similar input, will also be activated and the resulting movement will be a convex combination of the previously successful ones. Selection can be understood in a similar fashion, i.e., if there are both forehands and backhands in the data set, these will be responses to drastically different ball trajectories if viewed in the robot coordinates. Hence, the gating network will discriminate between both types of motor primitives.

6 Conclusion

In this paper, we have presented both past and current progress towards a complete framework for motor skill learning. While an overview paper in its nature, we have given a detailed outline of a general framework for motor skill the ingredients of which we have been investigating. We have presented selected topics in several important areas. In *learning to control*, we have reviewed our work on learning models using local GPs and on learning operational space control. When learning motor primitives, we have discussed both *imitation learning* approaches as well as our progress in *reinforcement learning* for robotics starting from policy gradients and moving towards reward-weighted self-imitation. Current work towards learning the supervisory layer for complex tasks is briefly discussed with a more in-depth focus on learning goal function as well as generalizing and selecting movement primitives. Successful implementations on real robots as well as in simulation underline the applicability of the presented approaches.

References

1. Sciavicco, L., Siciliano, B.: Modeling and control of robot manipulators.MacGraw-Hill, Heidelberg (2007)
2. Schaal, S., Atkeson, C.G., Vijayakumar, S.: Scalable techniques from nonparameteric statistics for real-time robot learning. Applied Intelligence, 49–60 (2002)

3. Schaal, S., Ijspeert, A.J., Billard, A.: Computational approaches to motor learning by imitation. In: Frith, C.D., Wolpert, D. (eds.) The Neuroscience of Social Interaction, pp. 199–218. Oxford University Press, Oxford (2004)
4. Ijspeert, A.J., Nakanishi, J., Schaal, S.: Learning attractor landscapes for learning motor primitives. In: Advances in Neural Information Processing Systems, vol. 15, pp. 1547–1554. MIT Press, Cambridge (2003)
5. Nakanishi, J., Farrell, J.A., Schaal, S.: Composite adaptive control with locally weighted statistical learning. Neural Networks 18(1), 71–90 (2005)
6. Nguyen-Tuong, D., Seeger, M., Peters, J.: Local Gaussian Process Regression for Real Time Online Model Learning and Control. In: Advances in Neural Information Processing Systems (NIPS 2008), vol. 21. MIT Press, Cambridge (2009)
7. Nguyen-Tuong, D., Seeger, M., Peters, J.: Computed torque control with nonparametric regression models. In: Proceedings of the 2008 American Control Conference (ACC 2008) (2008)
8. Nguyen-Tuong, D., Peters, J.: Semi-parametric regression in learning inverse dynamics. In: International Conference on Robotics & Automation (ICRA) (2010) (Submitted)
9. Chiappa, S., Peters, J.: Motion segmentation by detecting in continuous time-series. In: Advances in Neural Information Processing Systems (NIPS 2009), vol. 22. MIT Press, Cambridge (2010) (Submitted)
10. Chiappa, S., Kober, J., Peters, J.: Using Bayesian Dynamical Systems for Motion Template Libraries. In: Advances in Neural Information Processing Systems (NIPS 2008), vol. 21. MIT Press, Cambridge (2009)
11. Dayan, P., Hinton, G.E.: Using expectation-maximization for reinforcement learning. Neural Computation 9(2), 271–278 (1997)
12. Peters, J., Schaal, S.: Reinforcement learning of motor skills with policy gradients. Neural Networks 21(4), 682–697 (2008)
13. Kober, J., Peters, J.: Policy Search for Motor Primitives in Robotics. In: Advances in Neural Information Processing Systems (NIPS 2008), vol. 21. MIT Press, Cambridge (2009)
14. Kober, J., Mohler, B., Peters, J.: Learning Perceptual Coupling for Motor Primitives. In: Proceedings of the IEEE International Conference on Intelligent RObots and Systems (IROS) (2008)
15. Kober, J., Peters, J.: Learning Motor Primitives for Robotics. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (2009)
16. Jordan, M., Jacobs, R.: Hierarchical mixture of experts and the EM algorithm. Neural Computation 6, 181–214 (1994)
17. Muelling, K.: Learning (2009)
18. Peters, J., Mistry, M., Udwadia, F.E., Nakanishi, J., Schaal, S.: A unifying methodology for robot control with redundant DOFs. Autonomous Robots 24(1), 1–12 (2008)
19. Peters, J., Schaal, S.: Reinforcement learning by reward-weighted regression for operational space control. In: Proceedings of the International Conference on Machine Learning (ICML 2007), Oregon, USA, pp. 745–750 (2007)
20. Peters, J., Schaal, S.: Learning to Control in Operational Space. The International Journal of Robotics Research 27(2), 197–212 (2008)
21. Pastor, P., Hoffmann, H., Asfour, T., Schaal, S.: Learning and generalization of motor skills by learning from demonstration. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA) (2009)
22. Peters, J., Nguyen, D.: Real-Time Learning of Resolved Velocity Control on a Mitsubishi PA-10. In: International Conference on Robotics and Automation (ICRA) (2008)
23. Sentis, L., Khatib, O.: Synthesis of whole-body behaviors through hierarchical control of behavioral primitives. International Journal of Humanoid Robotics 2(4), 505–518 (2005)

Learning Landmark Selection Policies for Mapping Unknown Environments

Hauke Strasdat, Cyrill Stachniss, and Wolfram Burgard

Abstract. In general, a mobile robot that operates in unknown environments has to maintain a map and has to determine its own location given the map. This introduces significant computational and memory constraints for most autonomous systems, especially for lightweight robots such as humanoids or flying vehicles. In this paper, we present a universal approach for learning a landmark selection policy that allows a robot to discard landmarks that are not valuable for its current navigation task. This enables the robot to reduce the computational burden and to carry out its task more efficiently by maintaining only the important landmarks. Our approach applies an unscented Kalman filter for addressing the simultaneous localization and mapping problem and uses Monte-Carlo reinforcement learning to obtain the selection policy. In addition to that, we present a technique to compress learned policies without introducing a performance loss. In this way, our approach becomes applicable on systems with constrained memory resources. Based on real world and simulation experiments, we show that the learned policies allow for efficient robot navigation and outperform handcrafted strategies. We furthermore demonstrate that the learned policies are not only usable in a specific scenario but can also be generalized towards environments with varying properties.

1 Introduction

In recent years, there has been a trend towards embedded systems in robotics. A series of such approaches deal with autonomous cars, helicopters, blimps, underwater vehicles, and wheeled or humanoid robots. As embedded systems typically have much higher limitations with respect to the computational power and memory capacity, it is important in the context of embedded systems to develop efficient algorithms that scale with the computational constraints of the underlying hardware.

In robotics, one of the core capabilities needed for the majority of applications is autonomous navigation. For truly autonomous navigation in initially unknown

Hauke Strasdat
Dept. of Computing, Imperial College London, SW7 2AZ, UK

Cyrill Stachniss · Wolfram Burgard
University of Freiburg, Dept. of Computer Science, D-79110 Freiburg, Germany

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 483–499.
springerlink.com

environments, the robot has to solve the so-called simultaneous localization and mapping (SLAM) problem [2, 14, 24, 21]. Solving the SLAM problem, however, is computationally demanding and the memory requirements increase with the number of landmarks that need to be maintained by the robot. In practice, there are many scenarios in which the number of visible landmarks during a navigation task is significantly larger than the number of landmarks which can be processed efficiently using an embedded device. This leads to the question which landmark should be stored and maintained by the robot to optimally solve the navigation task. A landmark is only useful if it contributes to keep an accurate pose estimate of the robot at the right time and in such a way that it is valuable for the navigation task. In this paper, we present an approach for learning a landmark selection policy that optimizes the navigation task carried out by the robot given its computational or memory constraints. It is obvious that the utility of a landmark depends on the type of navigation task. We analyze two types of navigation tasks: A single-goal navigation task and a round-trip navigation task where subgoals are visited more then once. One major advantage of our approach is that the policies are not limited to the environment they were been learned in. Rather, they can also be applied successfully in environments with different properties of the underlying landmark distribution. Furthermore, we present a way to compress learned policies so that they become applicable on systems with reduced memory resources.

This paper is organized as follows. After a discussion of related work, Section 3 briefly introduces the unscented Kalman filter and its application to SLAM as well as reinforcement learning. Section 4 then describes the different navigation tasks considered in this paper. After that, we introduce our approach to learn the optimal landmark selection policy. Finally, we present experimental results carried out in simulation as well as on a real wheeled robot.

2 Related Work

The extended Kalman filter (EKF) [12] or its variants such as the unscented Kalman filter (UKF) [8] belong to the most popular approaches to solving the SLAM problem. With these algorithms, the computational requirements and memory demands increase at least quadratically with the number of landmarks since the full correlation between the position of all landmarks is taken into account. There are several alternative and approximative filtering techniques for SLAM [14, 24], which do not incorporate the full correlation between the landmarks, so that the computational constraints are less restrictive. However, their memory demand increases at least linearly with the number of landmarks used.

Recently, Sala *et al.* [18] presented a graph-theoretic formulation for the selection problem of visual features to perform navigation in known environments. The optimal set of features is defined as the minimal set with which navigation is possible. Zhang *et al.* [26] proposed an entropy-based landmark selection method for SLAM. This method specifies a measure of which visible landmark is best in terms of entropy reduction. However, it only provides a vague guideline for how many features should be selected at a given point in time. Furthermore, Lerner *et al.* [11] presented a quality measure for landmark selection in known environments which is based on the comparison of pose uncertainties. Dissanayake *et al.* [6] suggested a map management which ensures a uniform distribution of landmarks over the traversed area. Apart from landmark selection, other active methods were presented such as maximizing the SLAM estimate by intelligent path planning [5], or increasing the performance of a soccer playing robot by active sensing [10]. Recently, Hornung *et al.* [7] proposed a system for learning acceleration policies in the context of vision-based navigation. Furthermore, they presented a technique to compress the learned policy using a clustering approach. Several other policy/state space compression techniques for reinforcement learning were presented in the past [25][13]. While these techniques mainly focus on gaining a speed-up during learning, our policy compression approach is similar to Hornung *et al.*'s [7] and motivated by the storage problem: How can we represent the learned policy in a most compact way so that it becomes applicable on memory-constrained systems and so that, at the same time, the compression does not lead to a loss of performance?

In this paper, we present a universal approach for landmark selection in unknown environments. The value of a landmark is measured in terms of how well it improves the navigation/localization capabilities of the robot given the targeted navigation task. This is especially important for robots with restricted resources. We learn a landmark selection policy using Monte-Carlo reinforcement learning [3, 20, 23] and k-nearest neighbor regression [19]. We show in real world and simulation experiments that this technique allows for more efficient robot navigation. Furthermore, we demonstrate how the learned policies, which consist of tens of thousands of parameters, can be compressed using neural networks without a loss of performance. In this way, our approach becomes applicable on memory-constrained systems and extends our previous work [22].

3 Preliminaries

3.1 The Unscented Kalman Filter

The unscented Kalman filter (UKF) is a recursive Bayes' filter that estimates the state $\mathbf{x}$ of an dynamical system in discrete time steps given a sequence of actions $\mathbf{u}$ and observations $\mathbf{z}$. The n-dimensional state vector $\mathbf{x}$ is represented by a multivariate Normal distribution with mean μ and covariance matrix $\Sigma^{(n \times n)}$. The dynamics of the system are described by a *state transition function* g plus Gaussian noise $\epsilon_{g,t}$: $\mathbf{x}_t = g(\mathbf{u}_t, \mathbf{x}_{t-1}) + \epsilon_{g,t}$. Measurements are integrated using the *observation function* h: $\hat{\mathbf{z}}_t = h(\mathbf{x}_t) + \epsilon_{h,t}$. Again, Gaussian noise $\epsilon_{h,t}$ is added. Since Kalman filtering is an approach for systems governed by a linear difference equation, special efforts must be made to take the non-linearities in g and h into account.

The key idea of the unscented Kalman filter, which has been introduced by Julier and Uhlmann [8], is to apply a deterministic sampling technique that is known as the unscented transform to select a small set of so-called sigma points around the mean. Then, the sigma points are propagated through the non-linear functions. Afterward, mean and covariance estimates are computed based on the transformed points. The advantage of this technique is that the filter can much better deal with non-linearities, which leads to higher robustness than obtained with the EKF.

3.2 Simultaneous Localization and Mapping

In the context of the SLAM problem, one seeks to simultaneously determine the map of the environment and the pose of the robot. Probabilistic methods seek to estimate the joint probability distribution

$$p(\mathbf{p}_t, \mathbf{l}_1, \ldots, \mathbf{l}_M \mid \mathbf{u}_1, \ldots, \mathbf{u}_t, \mathbf{z}_1, \ldots, \mathbf{z}_t) \tag{1}$$

about the pose $\mathbf{p}_t$ of the robot at time t and the position of the landmarks $\mathbf{l}_1, \ldots, \mathbf{l}_M$ given all previous motions $\mathbf{u}_1, \ldots, \mathbf{u}_t$ and observations $\mathbf{z}_1, \ldots, \mathbf{z}_t$. Various approaches to estimate this posterior have been presented in the literature.

In this paper, we address the SLAM problem using the UKF by representing the joint state $(\mathbf{p}_t^T, \mathbf{l}_1^T, \ldots, \mathbf{l}_M^T)^T$ with $\langle \mu, \Sigma \rangle$. This is a standard approach which has been shown to operate successfully in the past. For convenience, we abbreviate the mean of the robot pose $(\mu_1, \mu_2, \mu_3)^T$ as $(x, y, \theta)^T$. The mean of the j-th landmark location $(\mu_{2j+2}, \mu_{2j+3})^T$ is denoted by $\left(l_x^{[j]}, l_y^{[j]}\right)^T$. Furthermore, we interpret the state transition function h as the robot's motion model. In addition, we assume that range and bearing observations $(\rho, \phi)^T$ are given so that we can define a corresponding observation model g. In our work, we initialize new landmarks in a single step as described by Bailey [2].

Note that our landmark selection framework is not limited to the UKF or other Kalman filter-based approaches. It can rather be applied to arbitrary other methods for addressing the SLAM problem.

3.3 Monte-Carlo Reinforcement Learning

The basic idea of reinforcement learning [23] is to learn by interacting with the environment. We consider a dynamic system consisting of an agent and its environment at discrete time steps τ. At each point in time τ, the world is in state $s_\tau \in \mathcal{S}$ and the agent chooses an action $a \in \mathcal{A}$. Then, the world transforms into a new state $s_{\tau+1}$ and the agent receives a reward $r_{\tau+1} \in \mathcal{R}$. The goal is to maximize the return

$$R_\tau = \sum_{k=\tau+1}^{\mathcal{T}} r_k, \tag{2}$$

where $\mathcal{T}$ is the total number of time steps of one learning episode. The agent is following a policy

$$\pi(s, a) := p(a \mid s) \quad \forall s \in \mathcal{S}, \tag{3}$$

which represents the probability of choosing action a under the assumption of being in state s. Each policy π has a corresponding Q-function

$$Q^{\pi}(s,a) := E_{\pi}\{R_{\tau} \mid s_{\tau} = s, a_{\tau} = a\}, \tag{4}$$

which specifies the expected return R from choosing action a in state s. During the learning process, we would like to approximate the optimal policy $\pi^{*}(s,a)$ that maximizes the expected return. Therefore, we have to approximate the corresponding Q-function simultaneously.

One way of solving the reinforcement learning problem is based on Monte Carlo methods [3, 20]. Here, we estimate the Q-function as the average return over sample episodes. Initially, the Q-function is initialized with a prior R_{prior}. During the training, a *soft policy* should be used. Thus, it should hold that $\pi(s,a) > 0$ for all possible state-action pairs in order to assure that each state is reachable during the training process. One common soft policy is ϵ-greedy which selects with the high probability of $1-\epsilon$ the action

$$a^{*} = \arg\max Q(s,a) \tag{5}$$

that maximizes the expected return and a random action otherwise.

Note that the time index t used in the SLAM setting is not necessarily identical with the discrete time at which the reinforcement learning framework has to make decisions. Therefore, we introduced a second time index τ.

4 Navigation Tasks

4.1 Single-Goal Task

Let us consider the following most basic navigation task (see Fig. 1 (a-c)). The robot is located at position A and is supposed to drive to the goal position B. In this example, the robot's motion is affected by a drift. In addition, N landmarks are distributed randomly over the environment. When the robot perceives a new landmark, it has to decide whether it should integrate this landmark into the UKF or not. The UKF has a landmark capacity of M landmarks with $M << N$. The goal is to choose the landmarks in such a way that the distance of the final position of the robot $(x_T, y_T)^T_{\text{true}}$ and the target position B is minimized. Hence, we define the reward

$$r_{\tau} = \begin{cases} -\left|B - (x_T, y_T)^T_{\text{true}}\right| & \text{if } \tau = \mathcal{T} \\ 0 & \text{else,} \end{cases} \tag{6}$$

as the negative Euclidean distance of the robot's true position to the goal B if the training episode reaches the terminal state $s_{\mathcal{T}}$; intermediate rewards $r_1,\ldots,r_{\mathcal{T}-1}$ are set to zero.

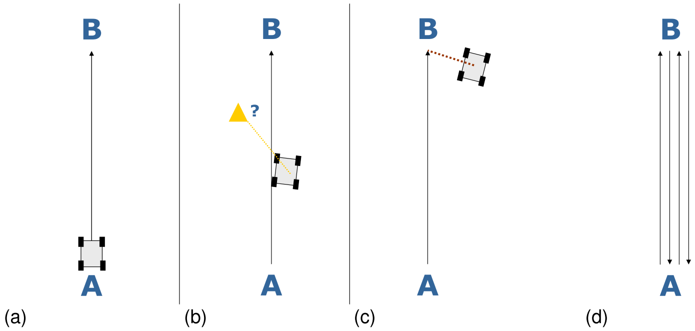


Fig. 1 Illustration of the single-goal navigation task (a-c) and the round-trip task (d).

4.2 Round-Trip Task

In the round-trip task, the robot is supposed to reach several subgoals (see Fig. 1 (d)). First, it starts at A and, is supposed to drive to B, and then back to A. This entire process is repeated twice. A new subgoal is selected as soon as the position estimate of the robot $(x_t, y_t)^T$ is close to it – independent of the robot's true position $(x_t, y_t)^T_{\text{true}}$. In this task, the error in the pose estimate should be minimized over the whole trajectory. For convenience, we specify the return directly as the negative average error over the remaining trajectory,

$$R_\tau = -\frac{1}{|T - t(\tau)|} \sum_{t'=t(\tau)}^{T} \left| \begin{pmatrix} x_{t'} \\ y_{t'} \end{pmatrix}_{\text{true}} - \begin{pmatrix} x_{t'} \\ y_{t'} \end{pmatrix} \right|, \tag{7}$$

whereas $t(\tau)$ specifies the time when the τ-th decision is made and T is the time when the robot reaches its final destination. To simplify things for the second task, landmark selection is only allowed while the robot moves from A to B the first time. The round-trip task is more complex than the previous one. However, it is worth considering since it focuses on the loop-closing problem of SLAM and therefore has a higher practical relevance than the single-goal task.

5 Navigation and Landmark Selection

5.1 Motion Control

The robot is steered towards the subgoals using a straightforward controller. An appropriate translational acceleration $\dot{\omega}_t$ and rotational acceleration $\dot{\upsilon}_t$ is selected based on the current estimate of the robot pose $\mathbf{p}_t$, the translational velocity ω_t and rotational velocity υ_t.

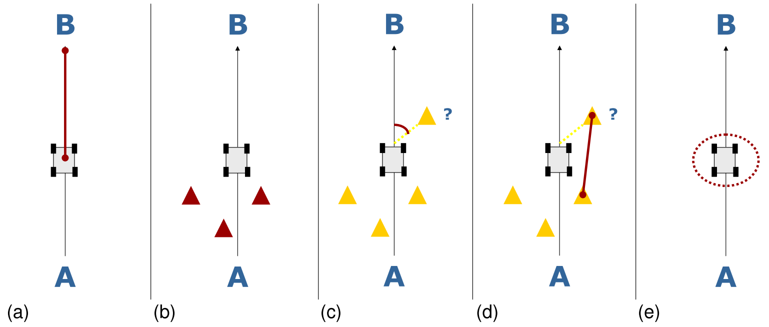


Fig. 2 Illustration of the state space.

5.2 *Learning Landmark Selection Policies*

In order to learn landmark selection policies with Monte Carlo reinforcement learning, we need to define the state space $\mathcal{S}$ and the action space $\mathcal{A}$. In addition to that, we need to find an appropriate representation for the continuous Q-function. First, however, we would like to discuss what makes this problem challenging.

Challenges

Learning a landmark selection policy for navigation tasks is a hard problem. Due to the stochasticity of robot motion, optimal landmark selection can still lead to a significant localization error of the vehicle. At the same time, the robot may reach a desired target location accurately by chance even if no landmarks are selected at all. Thus, the training data used in our learning domain is comparably noisy. In the long run, however, the average localization error reflects the quality of a specific selection strategy. Furthermore, it is essential to note that we perform navigation in *unknown environments*. Hence, it is not the goal to memorize a specific environment, but to learn general principles. For this reason, each learning episode is performed in a different environment. This makes the training data even more noisy: A selection policy which is good for one environment might not be a good policy in general (i.e., concerning the whole space of possible environments).

State Space

The available state information consists of the UKF state $\langle \mu, \Sigma \rangle$ and the current range and bearing observation $(\rho, \phi)^T$. This full information would lead to an high-dimensional state space so that successful learning is impractical. It is therefore desirable to reduce the space while preserving as much as possible of the relevant information. This can be achieved by defining features that summarize the essential information. One basic feature is the position of the potentially new landmark,

$$\begin{pmatrix} l_x^{[\text{new}]} \\ l_y^{[\text{new}]} \end{pmatrix} = \begin{pmatrix} x_t + \rho\cos(\phi + \theta_t) \\ y_t + \rho\sin(\phi + \theta_t) \end{pmatrix}, \tag{8}$$

according to the current range and bearing observation $(\rho, \phi)^T$ and the robot's pose estimate $(x_t, y_t, \theta_t)^T$. Additionally, we define the following five features:

1. Estimated distance d_{est} to subgoal B (see Fig. 2 (a)),

$$d_{\text{est}} = \left| B - (x_t, y_t)^T \right| . \tag{9}$$

2. Number of landmarks m integrated into the UKF (see Fig. 2 (b)),

$$m = \left| \{ j \in M : \Sigma_{2j+2} < \infty \wedge \Sigma_{2j+3} < \infty \} \right| , \tag{10}$$

   where Σ_{2j+2} and Σ_{2j+3} are the variances of the j-th landmark in the x and y direction.
3. Yaw angle ϕ to potentially new landmarks (see Fig. 2 (c)),
4. Distance d_l of the potentially new landmark to the closest landmark already integrated (see Fig. 2 (d)),

$$d_l = \min_{\substack{j \in L \\ \text{with } \Sigma_{2j+2} < \infty \wedge \Sigma_{2j+3} < \infty}} \left| \begin{pmatrix} l_x^{[j]} \\ l_y^{[j]} \end{pmatrix} - \begin{pmatrix} l_x^{[\text{new}]} \\ l_y^{[\text{new}]} \end{pmatrix} \right| . \tag{11}$$

5. Uncertainty of the robot pose $\Sigma^{3\times 3}$ in terms of its entropy (see Fig. 2 (e)),

$$H = \ln(\sqrt{(2\pi e)^3 |\Sigma^{3\times 3}|}). \tag{12}$$

The first of these features summarizes the robot position (x_t, y_t). The landmark positions $\mathbf{l}_1, \ldots, \mathbf{l}_M$ are summarized by the fourth feature while the new observation $(\rho, \phi)^T$ is represented by the third feature as well as fourth one. The covariance Σ_t is comprised by the second feature and the fifth one.

In the following, we will consider three different variants of the learning approaches. The first approach only relies on a two dimensional state space (first and second feature), the second one uses an four dimensional feature space (first to fourth feature), and the third one uses five dimensions (all five features).

Function Approximation

Since the state space of the features is continuous (with the exception of the second dimension), we need to estimate the Q-function with some function approximator. In our current implementation, we use k-nearest neighbor (k-NN) regression [19]. Training points – i.e. state/action values (s, a) which are each labeled with a return R – are efficiently stored in set of kd-trees [4, 1]. The jth kd-tree represents the returns from choosing an action a_j in a given state s. In the kd-tree representation, each dimension of the data points are normalized between zero and one, so that spherical search regions become practicable. If a query (s', a') is performed, the k nearest data points to the query point s' (w.r.t. Euclidean distance) are selected from the appropriate kd-tree. The return is estimated as unweighted average over the corresponding R-values. If less then k_{min} data points are found within a fixed radius around the query point, a prior R_{prior} is returned instead. In our current implementation, we set $k = 50$ to reflect the high amount of noise in our training data (see Sec. 5.2); k_{min} is set to 10. The k-NN regression approach has the advantage over the common grid-based discretization methods that it has a high degree of generalization in areas where the density is low and it is precise in regions where the data points are dense. In contrast to non-linear models such as neural networks [17], no over-fitting occurs. As opposed to other regression techniques, in which the model is also expressed directly in terms of their training data such as Gaussian Processes [15], k-NN regression is very fast. Even with hundreds of thousands of data points, a query can be performed in a few milliseconds. An efficient evaluation is essential for learning in practice, since the regression has to be carried out frequently. In various tests, we could not reveal a significant benefit from using Gaussian process regression over k-NN regression for reinforcement learning in our domain. Due to space restrictions, these experiments are omitted in the experimental section.

Action Selection

In our learning problem, the action is a binary decision:

$$\mathcal{A} = \{a_{\text{reject}}, a_{\text{accept}}\} \tag{13}$$

This means that either the potential new landmark is chosen or not. In order to boost the training, a variant of ϵ-greedy is used:

$$\pi(s) = \begin{cases} \arg\max Q(s,a) & \text{if } Q(s, a_{\text{reject}}) \neq Q(s, a_{\text{accept}}) \\ & \ \text{and } \chi_1 < 1 - \epsilon \\ a_{\text{accept}} & \text{if } [Q(s, a_{\text{reject}}) = Q(s, a_{\text{accept}}) \text{ or } \chi_1 \leq \epsilon] \\ & \ \text{and } \chi_2 < \frac{M}{N_{\text{visible}}} \\ a_{\text{reject}} & \text{else} \end{cases} \tag{14}$$

Here, χ_1 and χ_2 are uniform random samples between zero and one; N_{visible} is the expected number of visible landmarks in one training episode. Using this soft policy in the beginning of the training – when $Q(s, a_{\text{reject}}) = Q(s, a_{\text{accept}}) = R_{\text{prior}}$ in most cases – landmarks are selected with the probability of $\frac{M}{N_{\text{visible}}}$. Thus, it is ensured that landmarks are selected over the whole trajectory. Neither landmarks in the beginning of the episode nor landmarks in the end are preferred. If standard ϵ-greedy is used, a_{accept} and a_{reject} would be chosen with a probability of 0.5 each. Hence, depending on the values for the landmark capacity M and expected number of visible landmarks N_{visible} it could happen that either all landmarks are selected in the beginning of the episode or that considerably fewer landmarks than M are selected. Either would lead to a slow convergence rate.

To sum up, we use a learning approach for landmark selection based on Monte-Carlo reinforcement learning and k-NN regression. The state space is compactly represented by five features and the action is a binary decision.

5.3 Generalization

Until now, we considered an approach to learn a selection policy in unknown environments, but for a specific scenario. However, it is desirable to train a policy in one scenario and then apply this policy in another setting. Important parameters of a training scenario are the number N of landmarks in the environment and the landmark capacity M of the filter. To generalize, it is important to have a scenario-independent state space representation. For instance, instead of the number of landmarks m already integrated into the filter, we need to consider the percentage of landmarks $\frac{m}{M}$.

5.4 Policy Compression

Policies learned using k-NN-regression usually consists of tens of thousands of data points. To apply these policies on memory-constrained systems, we need to produce a compressed representation. One possibility is to compress the Q-function directly [25][13]. However, the Q-function also represents areas of the state/action space which are rarely visited when applying a greedy policy. Additionally, the Q-function maps each state/action value to a return value, whereas we are only interested in the action to apply given a state and not the specific return value. Therefore, we follow the approach of Hornung *et al.* [7] and perform the greedy policy n times (here, $n = 1,000$) and store the occurring decisions. As a result, we get a set of 10,000 state/action pairs. Since we only have two different actions, a_{reject} and a_{accept}, these pairs can be seen as labeled training data of a binary classification task. In order to compress the policy, any supervised classification technique can be applied which has a small number of model parameters. Here, we use a neural network [17] with one hidden layer, Rprop [16] for the weight optimization, and a sigmoid activation function. As this leads to a continuous output between zero and one, we treat values smaller than 0.5 as 0 (corresponding to a_{reject}) and all others as 1 (corresponding to a_{accept}).

6 Experiments

6.1 Single-Goal Task in Simulation

We evaluate the performance of our learning procedure for the single-goal task in a simulated environment. We consider an environment in which N landmarks are

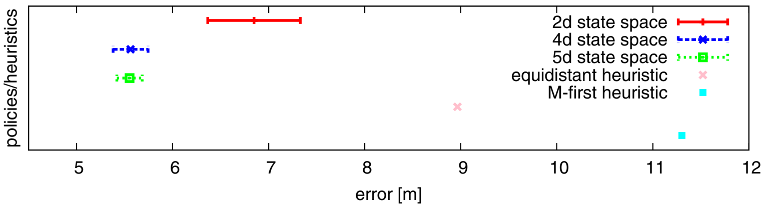


Fig. 3 Average performance of the trained policies and heuristics w.r.t. 1,000 test episodes. For each of the trained policies we show the mean over the ten training runs and the corresponding 95%-confidence interval.

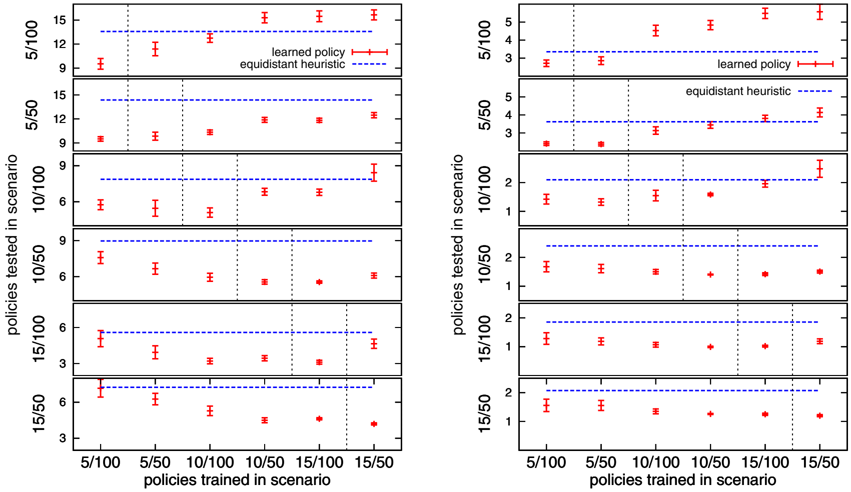
randomly distributed over a $30m$ by $60m$ area. The distance between the start position A and the goal B is set to $44m$. We train our policy for 1,000 episodes. In each episode, landmarks are randomly re-distributed. We compare the trained policies to two heuristics. The first one is the M*-first heuristic* which simply integrates the M first landmarks that are observed. An apparently better policy is the *equidistant heuristic*. With this heuristic, the robot only integrates a new landmark after it has driven a certain distance so that the landmarks are approximately uniformly distributed over the whole trajectory (similar to Dissanayake *et al.* [6]).

For the learning, we use k-NN regression with a two-, a four-, and a five-dimensional state space (see Section 5.2). At first, we consider an UKF with a landmark capacity of $M = 10$ and an environment with $N = 50$ landmarks. For each learning approach, ten training runs are performed. Each trained policy and heuristic is evaluated in 1,000 different environments (see Fig. 3). The one-sample t-test with a significance level of $\alpha = 0.05$ shows that all three learning approaches are significantly better than the equidistant heuristic. Furthermore, it a two-sample t-test revealed that k-NN regression with a four dimensional state space leads to a significantly smaller error than k-NN with two dimensions. Thus, the third feature, which is the distance d_l of a new landmark to the landmarks already integrated, and the fourth one, which is the angle ϕ to the new landmark, seem to include relevant information which are not encoded in the first two dimensions of the state space. Further experiments revealed that indeed both features are essential. However, we were not able to show that there is any benefit from including the fifth feature, the entropy H of the robot's pose. Even at a significance level of $\alpha = 0.25$, the t-test did not reveal a difference between the learning approach using the four-dimensional state space and the one using five dimensions.

In order to evaluate how good the trained policies generalize, we trained and tested a policy in environments with $N = 50$ as well as $N = 100$ landmarks. In addition, we use UKFs with a capacity M of five, ten, and 15 landmarks. Fig. 4 (a) illustrates the high degree of generalization of our learning approach. For instance, if we perform the training in a setting with $N = 50$ and $M = 5$, we see that the trained policy leads to significantly better results than the equidistant heuristic in all six test scenarios. This indicates that our approach generalized over different landmark densities which is similar to environments of different scale and sensor range.

6.2 Single-Goal Task Performed in a Real World Experiment

Furthermore, we evaluated our learning approach in a laboratory environment. We randomly attached visual markers [9] to the ceiling of a corridor in a $2.5m$ by $5m$ area. The robot we used, a Pioneer 2DX-8, is equipped with an upward-looking camera and a SICK laser range scanner (see Fig. 5). Whereas the camera was used to observe landmarks at the ceiling the laser was utilized to estimate the ground truth. Since the odometry of the robot was too accurate in the limited space in which we carried out the experiment, we added a rotational bias of 0.1 rad per meter. Unfortunately, it is impractical to train the policy in the real-world because this would not only require us to perform hundreds of training episodes but also to install different landmark distributions for each training episode. Thus, we trained the policy in simulation and tested it in the real-world setting. We also compared the trained policy to the equidistant heuristic. Both, the trained policy as well as the equidistant heuristic were tested ten times. The trained policy results in an error of 0.50 ± 0.08 whereas the equidistant heuristic leads to an error of 0.66 ± 0.07. Hence, the trained policy is significantly better than the equidistant heuristic (w.r.t. a t-test with $\alpha = 0.05$).



(a) Single-goal task (b) Round-trip task

Fig. 4 High degree of generalization in the single-goal task (a) and the round trip task (b). The mean error over ten training runs and the corresponding standard derivation is shown. All policies below the dashed line are significantly better than the equidistant heuristic (α = 0.05).

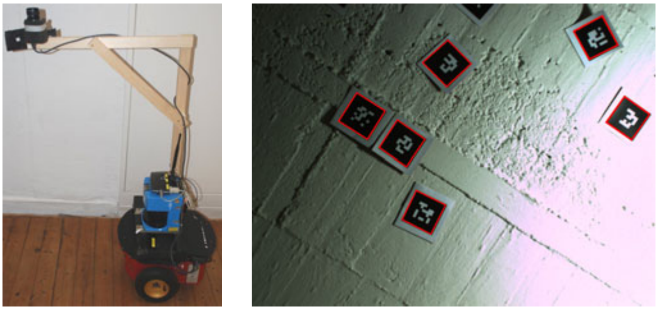


Fig. 5 Pioneer 2-DX8 robot with upward-looking camera and SICK laser range scanner (left) and detected visual landmarks on the ceiling (right).

6.3 Round-Trip Task

The performance of our learning procedure for the round-trip task is evaluated in a simulated environment in a similar way to the single-goal task. It was trained using k-NN regression with a four-dimensional state space over ten training runs. However, the error is defined as the average localization error over the whole trajectory. Again, we compare our learning with the equidistant heuristic. Fig. 4 (b) shows that the learned policy is significantly better than the heuristic. Furthermore, it shows that we are able to generalize over the UKF capacity M as well as the number of landmarks N.

6.4 Policy Compression Using Neural Network

The uncompressed policy for the single goal task is illustrated in Fig. 6. The policy is represented using approximately 20,000 four-dimensional state points which are either labeled with a_{accept} or a_{reject} (see Sec. 5.4). The illustration indicates that there is a large overlap between the two classes. This is most likely due to the noisiness of the training data, i.e., due to the fact that the very same policy can lead to different returns. However, it is important to notice that only *projections* of the four-dimensional state space are shown and that the overlap is not that severe locally. Since the state space of the round-trip task looks similar, we do not show it here.

We compressed our learned policy for the single-goal task and the round-trip task using neural network classification (for M=5 and N=50). Since we have a four-dimensional state space and we want to learn a binary decision, we use a neural network with four input units and one output unit. It turned out that one hidden layer with three units is sufficient for learning. This leads to a model with 19 parameters. Given this network, we were able to compress the learned policies which were approximately represented by 80,000 parameters (20,000 four-dimensional points), using only 19 parameters.

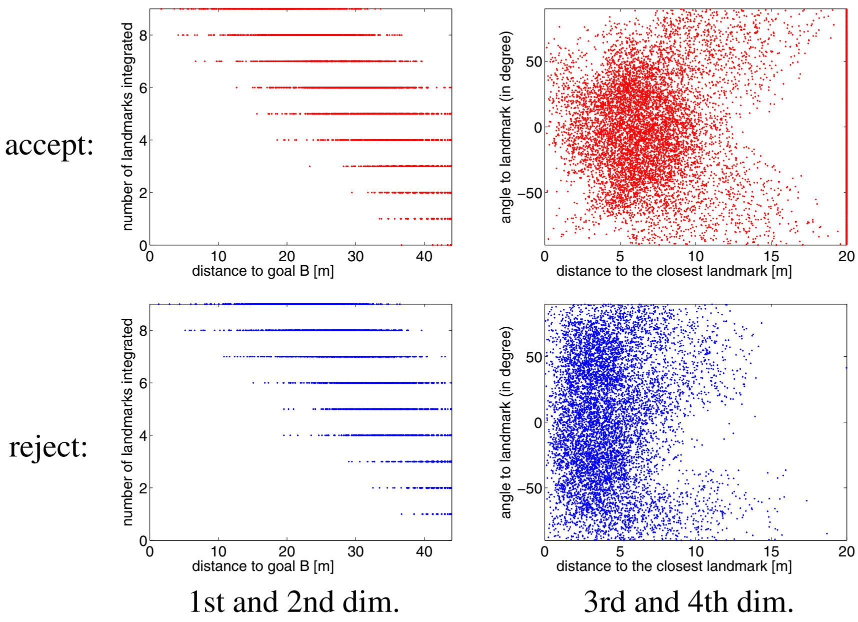


Fig. 6 Uncompressed strategy of the single-goal task. A projection of the four-dimensional state space onto the first and second dimensions (left) as well as onto the third and fourth dimensions (right) is shown.

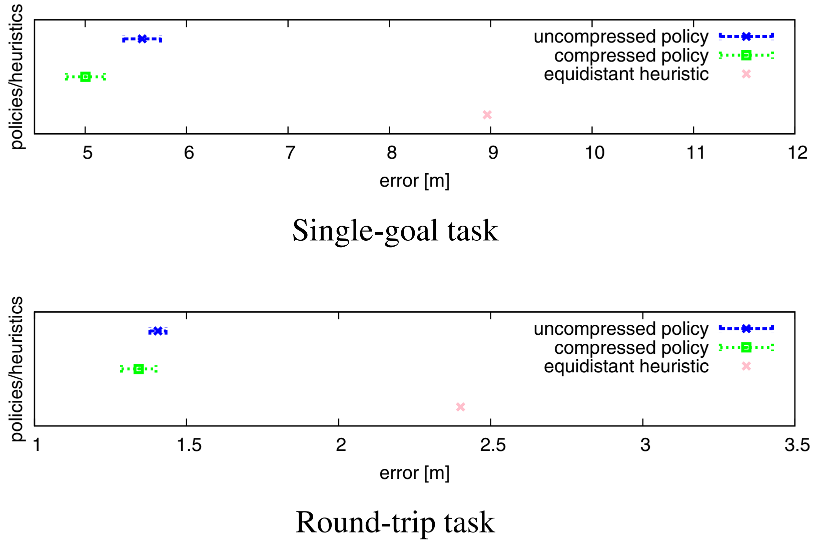


Fig. 7 Comparison of the compressed and uncompressed strategy of the single-goal task (top) and the round-trip task (bottom). For the trained policies, the mean over the ten training runs as well as the corresponding 95%-confidence interval is shown.

For both navigation tasks, we learned policies in ten training runs. Each of the learned policies was compressed using the method describe above. Comparisons between the compressed and the uncompressed policies are shown in Fig. 7. One can see that there is no performance loss using the compressed policies in place of the uncompressed ones. Surprisingly, the compressed policy for the single-goal task is even significantly better than the uncompressed one. This could be explained by the generalization property of neural networks. To analyze this more precisely, it is worth looking at the compressed policy. Since the neural network parameters are hard to interpret, we apply the same approach as before. We apply the compressed policy in 1,000 different environments and store the occurring decisions. The resulting labeled data points are shown in Fig. 8. Although, the policy looks similar to the uncompressed one (Fig. 6), there are differences. For instance the neural network was able to better model the decisions boundaries. Because of its smoothness, it performed better on the unknown test data.

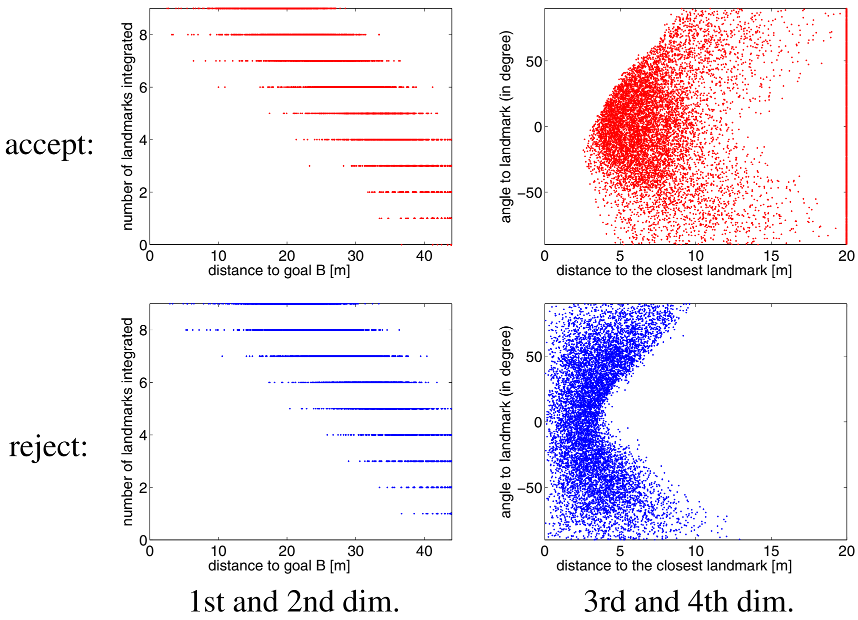


Fig. 8 Compressed strategy of the single-goal task. The figures show a projection of the four-dimensional state space onto the first and second dimensions (left) as well as onto the third and fourth dimensions (right).

7 Conclusion and Future Work

In this paper, we considered the problem of deciding whether to incorporate a landmark into the probabilistic belief of a SLAM approach or to discard it. Such techniques are especially relevant for efficient robot navigation under computational constraints. We presented a novel approach based on reinforcement learning that is able to effectively perform landmark selection in unknown environments. We demonstrated by a series of real world and simulation experiments that the learned policies outperform handcrafted heuristics. Furthermore, we showed that a learned policies have a high degree of generalization since they can be applied in different environments with changed underlying parameters. Finally, we were able to compress the learned policies by means of neural network classification without introducing a performance loss.

Despite these encouraging results, there is space for further improvement. One interesting aspect is the possibility to delete an already incorporated landmark. It should be noted that such a scenario is substantially more complex compared to the scenarios considered in the experimental section of this paper. For instance, the state space must be extended to also include information about the already integrated landmarks. In initial experiments carried out with a variant of our current implementation, we discovered that good strategies keep a set of landmarks fixed for re-localization and perform an incremental pose correction with set of frequently replaced landmarks.

References

1. Arya, S., Mount, D.M.: Algorithms for fast vector quantization. In: Proc. of the IEEE Data Compression Conference (DCC 1993), pp. 381–390 (1993)
2. Bailey, T.: Mobile Robot Localisation and Mapping in Extensive Outdoor Environments. PhD thesis, University of Sydney, 22–23 (2002)
3. Barto, A., Duff, M.: Monte Carlo matrix inversion and reinforcement learning. In: Advances in Neural Information Processing Systems, pp. 687–694 (1994)
4. Bentley, J.L.: K-d trees for semidynamic point sets. In: Proc. of the 6th ACM Symposium on Computational Geometry, pp. 187–197 (1990)
5. Bryson, M., Sukkarieh, S.: Active airborne localisation and exploration in unknown environments using inertial SLAM. In: Proc. of the IEEE Aerospace Conference (2006)
6. Dissanayake, G., Durrant-Whyte, H., Bailey, T.: A computationally efficient solution to the simultaneous localisation and map building (SLAM) problem. In: Proc. of the Int. Workshop on Augmented Reality (IWAR 1999), pp. 1009–1014 (2000)
7. Hornung, A., Strasdat, H., Bennewitz, M., Burgard, W.: Learning efficient policies for vision-based navigation, St. Louis, USA (2009)
8. Julier, S.J., Uhlmann, J.K.: A new extension of the Kalman filter to nonlinear systems. In: Proceedings of the Int. Symp. on Aerospace/Defense Sensing, Simulation and Controls (1997)
9. Kato, H., Billinghurst, M.: Marker tracking and HMD calibration for a video-based augmented reality conferencing system. Proc. of the Int. Workshop on Augmented Reality, IWAR (1999)
10. Kwok, C., Fox, D.: Reinforcement learning for sensing strategies. In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS 2004), Sendai, Japan (2004)
11. Lerner, R., Rivlin, E., Shimshoni, I.: Landmark selection for task-oriented navigation. IEEE Transaction on Robotics 23(3) (2007)
12. Maybeck, P.S.: The Kalman filter: An introduction to concepts. Autonomous Robot Vehicle. Springer, Heidelberg (1990)

13. Menache, I., Mannor, S., Shimkin, N.: Basis function adaptation in temporal difference reinforcement learning. Annals of Operations Research 134(1), 215–238 (2005)
14. Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B.: FastSLAM: A factored solution to the simulaneous localization and mapping problem. In: Proc. of the National Conf. on Artificial Intelligence (AAAI 2002), pp. 593–598 (2002)
15. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. MIT Press, Cambridge (2006)
16. Riedmiller, M., Braun, H.: A direct adaptive method for faster backpropagation learning: The Rprop algorithm. In: Proceedings of the IEEE International Conference on Neural Networks (1993)
17. Rojas, R.: Neural Networks: A Systematic Introduction. Springer, Heidelberg (1996)
18. Sala, P., Sim, R., Shokoufandeh, A., Dickinson, S.: Landmark selection for vision-based navigation. IEEE Transaction on Robotics 22(2) (2006)
19. Shakhnarovich, G., Darrell, T., Indyk, P.: Nearest-Neighbor Methods in Learning and Vision: Theory and Practice. MIT Press, Cambridge (2006)
20. Singh, S.P., Sutton, R.S., Kaelbling, P.: Reinforcement learning with replacing eligibility traces. Machine Learning 22, 123–158 (1996)
21. Stachniss, C., Grisetti, G., Burgard, W., Roy, N.: Evaluation of gaussian proposal distributions for mapping with rao-blackwellized particle filters. In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS 2007), San Diego, CA, USA (2007), http://www.informatik.uni-freiburg.de/ stachnis/pdf/stachniss07iros.pdf
22. Strasdat, H., Stachniss, C., Burgard, W.: Which landmark is useful? Learning selection policies for navigation in unknown environments. In: Proc. of the IEEE Int. Conf. on Robotics and Automation (ICRA 2009), Kobe, Japan (2009)
23. Sutton, R.S., Barto, A.G.: Reinforcement learning: An introduction. MIT Press, Cambridge (1998)
24. Thrun, S., Liu, Y., Koller, D., Ng, A.Y., Ghahramani, Z., Durrant-Whyte, H.: Simultaneous localization and mapping with sparse extended information filters. Int. Journal of Robotics Research 23 (2004)
25. Uther, W.T.B., Veloso, M.: Tree based discretization for continuous state space reinforcement learning. In: Proc. of the National Conf. on Artificial Intelligence (AAAI 1998), pp. 769–774 (1998)
26. Zhang, S., Xie, L., Adams, M.D.: Entropy based feature selection scheme for real time simultaneous localization and map building. In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS 2005 (2005)

Mapping

Cédric Pradalier

Understanding its environment is accepted as a major requirements for most mobile robots. In the following chapters, the authors focus on methods to represent and build a map using mostly images and 3D point clouds. The general consensus in the mapping community is that mapping static 2D environments has been solved and open-source turn-key solution are readily available. Instead, the challenge lies in the processing of visual data or 3D point clouds, the development of scalable representations, and the management of dynamic environments.

- In “Error-Driven Refinement of Multi-scale Gaussian Maps ”, Yguel et al. provides a new insight on the use of shape tensors as the representation of multi-resolution dense map made out of 2D or 3D point clouds. The approach relies on a smart clustering phase to minimise the number of tensors used to represent the map.
- In “Robust 3-D Visual SLAM in a Large-scale Environment ”, Kim et al. proposes an improvement to the solution to the 3D Visual SLAM problem. By separating features into those tracked from key-frame to key-frame and those appearing in the current key-frame, the authors manage to combine a motion estimator and a stochastic filter so as to implement a visual SLAM system more resilient to sudden camera motion.
- In “Towards Large-Scale Visual Mapping and Localization ”, Pollefeys et al. discusses an image processing pipeline for real-time localization and mapping using multiple camera sources on a mobile platform. An important aspect of this paper is its broad overview of all the problems faced when developing a vision-based localization and mapping system. This chapter also highlights solutions to some of these problems.
- In “Place-Dependent People Tracking ”, Luber et al. take another viewpoint on the mapping problem. The main problem addressed by this chapter is mapping the motion pattern of people based on their position in the environment. Such a map could then be used to predict people's motion and integrate this information in a motion planner.
- In “Combining Laser-Scanning Data and Images for Target Tracking and Scene Modeling ”, Zha et al. use a combination of multiple laser scanners and cameras to detect and track objects in an urban environment. In particular, this chapter presents mapping results from static cameras ob-

serving a dynamic environment and a sensor-suite mounted on a car driving around Peking university.

- In “Towards Lifelong Navigation and Mapping in an Office Environment”, Wyeth and Milford present RatSLAM as an alternative to classical cartesian-map-based localisation and mapping. An important aspect of this work is the focus on functional accuracy of the map and on the evaluation of the performance through long-term testing.

A common thread of this part of the ISRR'09 proceedings is the focus on managing high quantities of data, either due to experiments spanning long durations, or due to large volume of space observed during the experiment. As sensors, robotic platforms, and algorithms improve, processing the volume of data becomes more and more a challenge, and employing more and more parallelism (e.g. using Graphic Processing Unit) on robotic platforms becomes a norm. Other researchers, such as Wyeth and Milford, are taking an alternative route to show that a fully metric representation of the environment may not be the only way to reach the autonomous navigation functionality. As often said in the field of robotic research, it all depends on the task.

Error-Driven Refinement of Multi-scale Gaussian Maps

Application to 3-D Multi-scale Map Building, Compression and Merging

Manuel Yguel, Dizan Vasquez, Olivier Aycard, Roland Siegwart,
and Christian Laugier

Abstract. The accuracy of Grid-based maps can be enhanced by putting a Gaussian in every cell of the map. However, this solution works poorly for coarse discretizations in multi-scale maps. This paper proposes a method to overcome the problem by allowing several Gaussians per cell at coarse scales. We introduce a multi-scale approach to compute an error measure for each scale with respect to the finer one. This measure constitutes the basis of an incremental refinement algorithm where the error is used to select the cells in which the number of Gaussians should be increased. As a result, the accuracy of the map can be selectively enhanced by making efficient use of computational resources. Moreover, the error measure can also be applied to compress a map by deleting the finer scale clusters when the error in the coarse ones is low.

The approach is based on a recent clustering algorithm that models input data as Gaussians rather than points, as is the case for conventional algorithms. In addition to mapping, this clustering paradigm makes it possible to perform map merging and to represent feature hierarchies under a sound theoretical framework. Our approach has been validated with both real and simulated 3-D data.

1 Introduction

The idea of producing multi-scale grids has been present since the very first works on grid-based representations, [1]. Coarse maps are used in path planning [2, 3] or

Manuel Yguel · Christian Laugier
INRIA Rhône-Alpes, Grenoble
e-mail: firstname.lastname@inrialpes.fr

Dizan Vasquez · Roland Siegwart
ETHZ, Zürich
e-mail: name@mavt.ethz.ch

Olivier Aycard
UJF, Grenoble
e-mail: firstname.lastname@inrialpes.fr

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 503–518.
springerlink.com

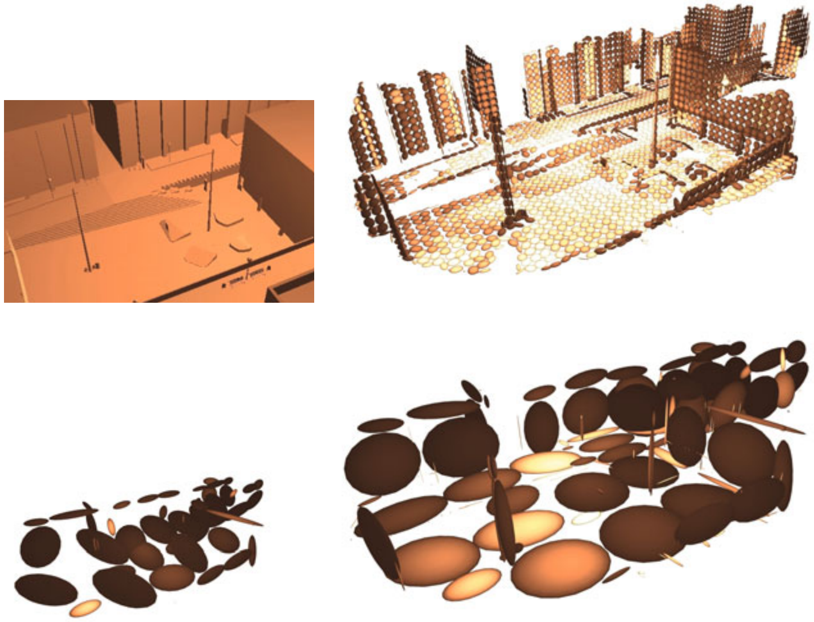


Fig. 1 Top left: simulated scene. Top right: fine scale of the map. Bottom left: coarse scale of the map, one Gaussian per cell. Bottom right: coarse scale of the refined map

localization [4] algorithms in order to obtain a rough trajectory or position estimate at a low computational cost. Then, at a second stage, this estimate is used to initialize the fine scale search algorithms, thus accelerating convergence. For localization, this procedure also enlarges – in most cases – the convergence region. Examples of algorithms that benefit from such an approach include sampling-based, gradient-based and line-to-point or plane-to-point ICP-based algorithms.

However, as the resolution becomes coarser, the aliasing effect of the geometry of the cells becomes more evident and it can no longer be neglected. When considering a cell as a full block, all the information concerning the shape of the cell contents is lost. A way to alleviate this problem is to attach some sort of statistical shape description to every occupied cell. Two seminal works in this direction are tensor maps [5] and the Normal Distribution Approach (NDT) [6], these approaches significantly improve accuracy by approximating the shape of the cell contents with ellipsoids that are encoded as symmetric semi-definite positive (SSDP) matrices. The accuracy of these approaches, together with their relative simplicity has contributed to making them very popular in the map building community [7, 4].

That being said, a single ellipsoid is still a poor representation when there are several objects with different orientations in the cell, which is the case –for instance– of a pole standing on the ground (see fig. 1). In this paper, we present a method to overcome this problem by allowing a coarse resolution cell to contain multiple ellipsoids – more specifically, Gaussian clusters. The idea is to start with a single cluster per cell, and then to *refine* it by inserting additional clusters in order to achieve a balance between representational complexity and accuracy. In particular, from now on, we will assume that there is a given budget of Gaussians per coarse scale that needs to be allocated in an optimal way through a refinement process.

This paper is structured as follows: In the following section, we review related works in robotics and computer graphics. Section 3 provides an overview of our mapping framework. In section 4, we explain how to update a Gaussian map from a range image and how occupancy may be used as a learning rate. Section 5 presents our error-driven refinement algorithm for coarse scales. Section 6 discusses mapping results on simulated and real data sets. Finally, we present our conclusions and outline possible directions for future work.

2 Related Works

2.1 Related Mapping Approaches in Robotics

An interesting approach to grid refinement are multi-level surface maps (MLS) [8], which can be considered as a refinement of an elevation map. An MLS map is a 2-D grid where, for each cell, several planes are stored at different heights, together with the associated thickness. Their structure makes them particularly well suited to represent traversability information, as shown by their impressive results on real data sets. However, they share the aliasing related problems of 2-D grids particularly in the horizontal plane. Moreover, due to their lack of merging mechanisms they often fail to represent tall vertical structures as a single element if those structures were partially occluded during early observations.

A different approach to cope with cell aliasing is to use a multi-scale grid map which is refined where features are denser. Tree-based representations, such as quadtrees and octrees [9, 10] are the most popular data structures of this kind for two and three dimensional input spaces, respectively. Nevertheless, these structures also suffer from the aliasing problem because of their cubic cell shape, which makes them inappropriate to represent curves or surfaces.

Tensor voting and NDT aim at improving geometric accuracy by representing all the points that fall into a given cell by an ellipsoid, whose orientation and shape are such that the representation error is minimized. In both cases, the ellipsoid is encoded as an SSDP matrix (Fig. 2).

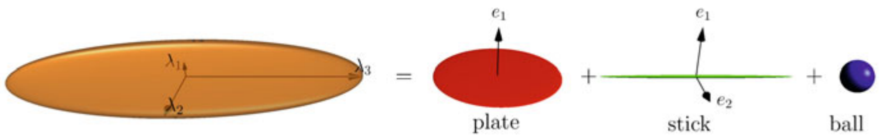


Fig. 2 Decomposition of an SSDP matrix ellipsoid into elementary shapes called tensors in [5].

In both tensor voting and NDT, having one cluster per cell produces large ellipsoids (called junctions in the tensor voting framework) as soon as the resolution is too coarse and the cell encompasses several distinct objects or the object geometry is not linear. This problem has been addressed in the original NDT paper [6] by storing four overlapping maps shifted by half a cell. However this approach is expensive in terms of the number of Gaussians added and the advantages are unclear when compared to a map with twice the resolution.

2.2 Related Approaches in Computer Graphics

In the computer graphics community the problem of 3-D model simplification has received a lot of attention. The objective in this case is to obtain simpler models to streamline manipulation and speed up rendering when high accuracy is not required – *e.g.* when objects are far from the virtual camera or moving rapidly. This is deeply related to refinement as it is, essentially, the inverse problem.

The seminal work in this field is the paper of Garland *et al.* [11] where edge contraction is performed on the mesh edges that introduce the smallest amount of *quadric error* in the model. As indicated by its name, this metric is defined by calculating a special type of quadric on the vertices, described by SSPD matrices. The simplification algorithm uses a priority queue. At every iteration, the edge having the lowest error is contracted, the error of all the affected edges is recomputed and they are reinserted in the queue. The process continues until the target budget of edges is reached.

A second class of effective simplification approaches is based on clustering. They can operate either on meshes or on point clouds. Probably the most relevant example of mesh clustering is [12] where clustering is performed on the triangles of the mesh to be simplified. An essential component of this approach is a shape metric that makes it possible to assign each triangle to its closest cluster and to compute the parameters of the cluster. Cohen and Steiner [12] consider two metrics: Garland's quadric error metric [11] and the Euclidean distance between the normals of the triangles and their cluster normals.

In [13], Pauly *et al.* have studied the simplification of point clouds of the same kind than those obtained with a laser range finder. They describe several agglomerative and partitional approaches, applying techniques proposed in [11].

Our approach is largely inspired by the work of Cohen and Steiner [12] and by the partitional algorithm of Pauly [13]. The main difference lies in the fact that we do not restrict our main representation to surfaces, because at coarse scales many important features such as poles, trees trunks and towers may appear as one-dimensional curves rather than surfaces.

3 Approach Overview

The data are range images, in which each pixel contains a distance value as for stereo camera for instance (see fig. 4). We suppose that a localization is provided and we construct a multi-scale map as follows. We compute 3 scales of Gaussian maps. Each scale is a sparse grid where only occupied cells are stored and in the fine scale there is only one Gaussian per cell. In coarser scales, thanks to refinement, a cell can contain several Gaussians. Our approach processes every scale independently, adjusting the field of view of finer scales in order to have a similar number of cells to update for each scale. As shown in Fig. 3, the approach is composed of three main components:

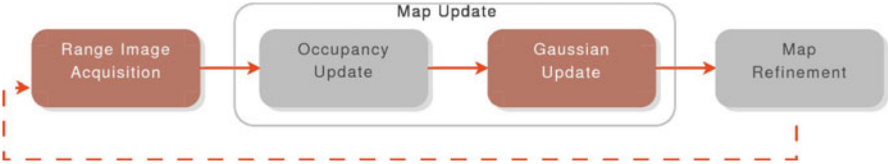


Fig. 3 Framework components. Light gray boxes are processed less than once per range image.

1. Occupancy computation: it is the likelihood that a cluster represents a static part of the map. In our framework it allows to adapt faster the regions that have not been observed often and it is used as a criterion to delete clusters, making it possible to remove dynamic objects from the map.
2. Map updating: it adapts existing clusters in order to minimize the representation error. It is similar to the standard update of conventional Gaussian maps, except that it takes into account the fact that a cell may contain several Gaussians.
3. Map refinement: this step is our main contribution, at every refinement step new Gaussian clusters are added to the cells where the representation error is maximum. It is worth noting that no refinement is performed on the finest scale.

4 Map Updating

This section presents the procedure to update an existing map from sensor data. At this point, we assume that the number k of Gaussian clusters per cell is known. The actual estimation of k is handled by the refinement algorithm that we will discuss in § 5.

Our goal here is to update the Gaussians' mean value μ and covariance Σ in order to minimize the representation error with respect to the input data. Every point in the range image is used to incrementally update the different scales independently. As we will explain in § 4.3, the basic idea is to find the cell where the input point falls and then updating the cluster in that cell that is ”closest” to the input point.

As in most incremental approaches, an important question is how much to adapt the clusters – *i.e.* finding the 'right' learning rate. In the following subsection we describe the use of the cluster's occupancy to control the adaptation. It can be intuitively explained as follows: the more a cluster has been observed, the more is known about it and the less reasonable it is to modify it. As we will see in § 5, occupancy is also used as a criterion to filter out dynamic objects from the map.

4.1 Computing Cluster Occupancy

Occupancy can be seen as a counter associated to every object. Its value gets increased when the object is visible in the range image, and decreased when the object is supposed to be visible but is in fact not. Fig. 4 illustrates the idea: if point C, is visible – *i.e.* the dashed red line is free from obstacles between I and C – then the value of the range image at cell I will correspond to the distance r_C. If, on the other hand, the value of cell I is greater than r_C, this can be considered as evidence that C is not there anymore and its occupancy should be decreased.

For visible cells, we compute occupancy in a per point basis. The occupancy of a point, C, in the map is given by comparing the range measured in the pixel of its projection, I, in the range image with its actual range, δ. For a Gaussian, the occupancy is obtained by averaging the occupancy of n points sampled from the Gaussian distribution (rejecting those that fall outside the cell).

We only need to guarantee that there are enough samples to provide a good estimate. To define n, we compute an upper bound of the number of points in the range image that can be contained in the cell. This is done using the projected bounding sphere of the cell. Let $\delta_{\min}$ and δ be the distance to the image plane in the camera coordinate system and the distance to the center of c (fig. 4), respectively. Then the projection of the bounding sphere of c occupies an area of $\frac{3\pi}{4}\left(\frac{\delta_{\min}}{\delta}a\right)^2$ (orange disc in Fig. 4), where a is the length of the side of c. Knowing the area of one pixel of the range image p, an upper bound for the number of pixels that may be projected back into the original cluster is:

$$B \triangleq \lceil \frac{3\pi}{4p}\left(\frac{\delta_{\min}}{\delta}a\right)^2 \rceil \tag{1}$$

which is the number of samples we are looking for. So, making $n = B$ gives us a good chance to cover every range image cell that effectively contains an observation from the cluster.

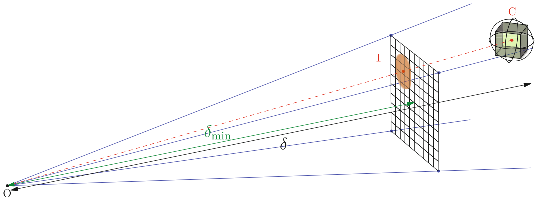


Fig. 4 Computation of occupancy in the range image of the bounding sphere of a cell.

4.2 Hierarchical Culling

In order to minimize computation, we perform hierarchical visibility culling. Consider the cell c centered in C in (fig. 4). If the projection of the bounding sphere of c is outside the range image (camera field of view, blue lines in fig. 4), the finer children cells of c are not further explored. The search is also finished if all the ranges observed in the disc of the projected bounding sphere (orange disc in the image plane) are smaller than the smallest possible range for the cell, meaning that all the cell content is occluded by closer objects.

4.3 Updating Gaussian Clusters from Data

For every input point, a single cluster per scale will be updated. The cluster is selected by finding the cell that contains the input point and then finding the cluster having the minimum distance to that point.

Once the cluster has been selected, its parameters are updated by means of a stochastic gradient descent algorithm. The principle is to update the reference vector by a fraction of the negative gradient with respect to each input datum. As more and more samples are processed, the magnitude of the adaptation should decrease (typically faster than $1/n$) to ensure convergence. A good example is the on-line computation of the sample mean:

$$\mu^n = \mu^{n-1} + \tfrac{1}{n}(\mathsf{z}^n - \mu^{n-1})$$

where n represents the number of samples processed so far, and $\mathsf{z}^n - \mu^{n-1}$ can be understood as the negative gradient, and $\frac{1}{n}$ the fraction of the gradient to be taken into account. This decreasing weight is called the learning rate and is noted ε. In our approach, the value of ε depends on the occupancy, as described in § 4.4.

In the case of points, a distance metric between a point and a Gaussian should be used. We have chosen to use the probability measure given by (2):

$$d(\mathsf{p},\mathsf{w}) \triangleq \frac{1}{2}\left[(\mathsf{p}-\mu_{\mathsf{W}})^T {\Sigma_{\mathsf{W}}}^{-1}(\mathsf{p}-\mu_{\mathsf{W}}) + \log\left(\det(\Sigma_{\mathsf{W}})\right)\right] , \tag{2}$$

This distance is the addition of the Mahalanobis distance and a volume term. Compared to the pure Mahalanobis distance, the volume term aims at compensating the fact that the Mahalanobis distance of a big cluster tends to make every point very close. This measure has the advantage of yielding simple map update rules, since the derivatives are:

$$\frac{\partial\, d(\mathsf{p},\mathsf{w})}{\partial\, \mu_{\mathsf{w}}} = -{\Sigma_{\mathsf{w}}}^{-1}(\mathsf{p}-\mu_{\mathsf{w}}) , \tag{3}$$

$$\frac{\partial\, d(\mathsf{p},\mathsf{w})}{\partial\, \Sigma_{\mathsf{w}}} \propto -\left[(\mathsf{p}-\mu_{\mathsf{w}})(\mathsf{p}-\mu_{\mathsf{w}})^T - \Sigma_{\mathsf{w}}\right] , \tag{4}$$

giving the following gradient descent algorithm for point-based update:

Algorithm 1 Map update with points: pointUpdate

```
$\{\mathsf{w}_1,\ldots,\mathsf{w}_k\} \leftarrow$ the $k$ Gaussian reference vectors of the cell
$\{\varepsilon_j | i = 1,\ldots,k\} \leftarrow$ the associated learning rates
$\mathsf{z} = \mathsf{p} \leftarrow$ the observed point

$n \leftarrow \arg\min_{i=1,\ldots,k} d(\mathsf{z},\mathsf{w}_i)$
$\mu_{\mathsf{w}_n} \leftarrow \mu_{\mathsf{w}_n} + \varepsilon_n(\mathsf{p} - \mu_{\mathsf{w}_n})$
$\Sigma_{\mathsf{w}_n} \leftarrow \Sigma_{\mathsf{w}_n} + \varepsilon_n \left[(\mathsf{p} - \mu_{\mathsf{w}_n})^T(\mathsf{p} - \mu_{\mathsf{w}_n}) - \Sigma_{\mathsf{w}_n}\right]$
```

4.4 *Learning Rate*

Our idea is to define the learning rate from the occupancy: the higher the occupancy of a cluster, the better the accuracy of its position and shape is supposed to be; thus, a small value of ε should be used. If, on the other hand, the occupancy is low, the current estimated state of the reference vector can be assumed to be based on insufficient statistics and the learning rate should be high to permit the reference vector to adapt itself.

In log-ratio the occupancy typically is bounded in $[-o_{\max}, o_{\max}]$ and the learning rate varies within $[\varepsilon_{\min}, \varepsilon_{\max}]$. For our approach we have chosen a linear mapping between both values:

$$\varepsilon(o) = \frac{\varepsilon_{\min} - \varepsilon_{\max}}{2o_{\max}}(o + o_{\max}) + \varepsilon_{\max} \; . \tag{5}$$

In our experiments, we have set $o_{\max} = 10.0$, $\varepsilon_{\max} = 5 \cdot 10^{-2}$ and $\varepsilon_{\min} = 5 \cdot 10^{-4}$.

5 Error-Driven Refinement of Coarse Scales

The refinement process is driven by a measure of the representation error. The map is refined by inserting a new cluster in the cell that has the maximum error. After every insertion, the shape of the other clusters in the same cell should be modified accordingly; this is done by running a clustering algorithm using the cells of the finer scale as input.

We periodically refine the map by adding a fixed number p of Gaussian clusters at a time. In order to choose the p vectors that have the maximum error without sorting the whole set of reference vectors, we use a priority queue of size p as was done in [11]. The following subsections provide the details of the refinement algorithm: the error metric used to build the queue is introduced in § 5.1 and § 5.2 presents the clustering algorithm.

During the mapping process it is often necessary to delete clusters that correspond to moving obstacles; this process is described in § 5.3. Finally, the application of our approach to map merging and simplification is discussed in § 5.4.

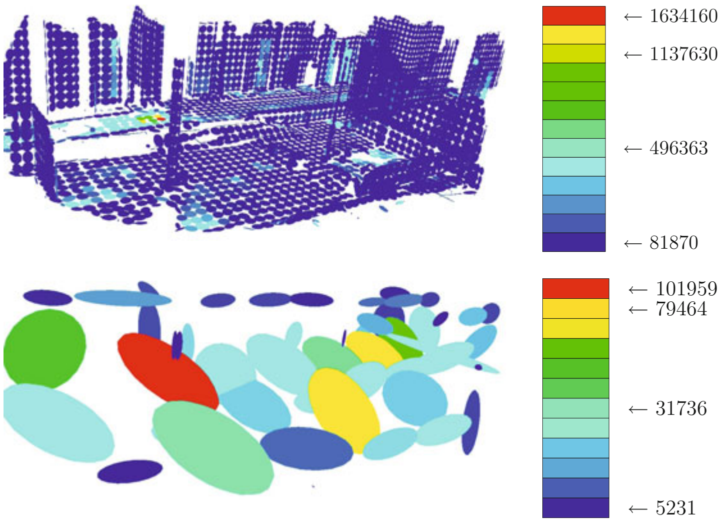


Fig. 5 Up: fine scale is colored with the magnitude of the contribution to the error at the coarser scale. Down: coarse scale, mean error. Error palettes are on the right.

5.1 Error Computation

We aim at adding clusters only in those regions where the Gaussian shapes have already converged to their final shapes, which can be deduced from its occupancy. Accordingly, we choose to refine a cell c only if the average occupancy probability of the finer cells in $\phi(c)$ is above 0.5. Furthermore, we only refine those parts of the map that are visible for the sensor.

To find the cell to refine, we compute an error value per cell. This value is basically the sum of the Mahalanobis distance between the center of the coarse cluster and the Gaussian cluster of the finer scale.

For the cells c^s of the coarse scale, s, having reference vectors (*i.e.* mean values) $\{\mathsf{w}_1,\ldots,\mathsf{w}_k\}$ and finer data at $s-1$: $\{\mathsf{z}_1,\ldots,\mathsf{z}_N\} \in G(\phi(c^s))$, we compute the average distance of each datum to its closest reference vector:

$$\mathscr{E}(c^s) = \frac{1}{N}\sum_{i=1}^{k}\sum_{j=1}^{N}(1-\varepsilon_{\mathsf{z}_j})\delta(\mathsf{w}_i,\mathsf{z}_j)\ \ (\mu_{\mathsf{w}_i}-\mu_{\mathsf{z}_j})^T\Sigma_{\mathsf{z}_j}^{-1}(\mu_{\mathsf{w}_i}-\mu_{\mathsf{z}_j}) \ , \tag{6}$$

where $\delta(\mathsf{w}_i,\mathsf{z}_j)$ is one if w_i is the closest reference vector to z_j using the Mahalanobis distance defined by z_j and zero otherwise. The occupancy is used through the learning rate to assign higher error weights to occupied clusters, disregarding those whose occupancy is low and, in consequence, whose accuracy may still improve without the need of adding extra clusters.

5.2 Clustering for Map Refinement

Algorithm 2 describes our clustering approach for map refinement. This method solves a hard clustering problem: we are looking for a partition $C^* = \{C_1^*, \ldots, C_k^*\}$ of the $G(\phi(c^s))$ into k classes represented by k reference vectors that minimizes the clustering distortion:

$$\mathscr{E}_{(C^*,\{\mathsf{w}_1^*,\ldots,\mathsf{w}_k^*\})} = \underset{\{C_1,\ldots,C_k\},\{\mathsf{w}_1,\ldots,\mathsf{w}_k\}}{\arg\min} \sum_{i=1}^{k} \mathscr{E}_{C_i}(\mathsf{w}_i) \tag{7}$$

Algorithm 2 Map refinement using hard clustering

```
$Z = \{\mathsf{z}_j | j = 1, \ldots, N\} \leftarrow$ the $N$ Gaussians of the fine scale $s$
$A = \{\alpha_j | j = 1, \ldots, N\} \leftarrow$ the non negative weights of the fine Gaussians
$W = \{\mathsf{w}_1^0, \ldots, \mathsf{w}_k^0\}[k-1] \leftarrow$ the $k-1$ Gaussians of the coarse scale $s+1$
$d_{\mathscr{X}\times\mathbb{F}}(\cdot,\cdot) \leftarrow$ the distance function

$W^0 \leftarrow \{\mathsf{w}_1^0, \ldots, \mathsf{w}_k^0\}[k-1] \bigcup \{\mathsf{z}_{\max}^0\}$ // Init. with the data of maximum distortion
$\{(C_i^0, \mathsf{w}_i^0) | i = 1, \ldots, k\}$, $\mathscr{E}_{W^0} \leftarrow$ kMeans$(Z, A, W^0, d_{\mathscr{X}\times\mathbb{F}})$ // clustering partition and distortion

// Simulate a swap
repeat
  for all $C_i^t$ do
    $\mathsf{z}_{\max(i)} \leftarrow \arg\max_{\mathsf{z}_j \in C_i^t} d_{\mathscr{X}\times\mathbb{F}}(\mathsf{z}_j, \mathsf{w}_i)$
    $d_i \leftarrow d_{\mathscr{X}\times\mathbb{F}}(\mathsf{z}_{\max(i)}, \mathsf{w}_i)$
  end for
  $c_{\max} \leftarrow \arg\max_{i=1,\ldots,k} d_i$
  $d_{\max} \leftarrow d_{c_{\max}}$
  $(u_{\min}, v_{\min}) \leftarrow \arg\min_{(u,v),\ u \neq v} d_{\mathscr{X}\times\mathbb{F}}(\mathsf{w}_u, \mathsf{w}_v)$
  $d_{\min} \leftarrow d_{\mathscr{X}\times\mathbb{F}}(\mathsf{w}_{u_{\min}}, \mathsf{w}_{v_{\min}})$
  if $d_{\max} < d_{\min}$ then
    $c_{\min} \leftarrow$ the cluster, $C_{u_{\min}}^t$ or $C_{v_{\min}}^t$, with the smallest number of elements.
    $W^{t+1} \leftarrow (W^t \setminus \{\mathsf{w}_{c_{\min}}\}) \bigcup \{\mathsf{z}_{\max(c_{\max})}\}$
  else
    $c \leftarrow\sim \mathscr{U}([\![1;k]\!] \setminus \{c_{\max}\})$// Draw a random candidate
    $W^{t+1} \leftarrow (W^t \setminus \{\mathsf{w}_c\}) \bigcup \{\mathsf{z}_{\max(c_{\max})}\}$
  end if

  $\{(C_i^{t+1}, \mathsf{w}_i^{t+1}) | i = 1, \ldots, k\}$, $\mathscr{E}_{W^{t+1}} \leftarrow$ kMeans$(Z, A, W^{t+1}, d_{\mathscr{X}\times\mathbb{F}})$

  $t \leftarrow t+1$
until $\mathscr{E}_{W^t} > \mathscr{E}_{W^{t-1}}$ // Accept the swap if the clustering distortion decreases

return $\{(C_i^{t-1}, \mathsf{w}_i^{t-1}) | i = 1, \ldots, k\}$, $\mathscr{E}_{W^{t-1}}$
```

This is done by using the well known k-means clustering algorithm [14].The optimal clusters are computed iteratively from the set of reference vectors : each datum is associated to its closest reference vector; then, the minimizer of each cluster energy is computed. In the basic Lloyd algorithm, both input data and reference vectors are simply points in feature space (3-D space in our case) $\mathscr{Z} = \mathbb{F} = \mathbb{R}^3$ and the distance function, $d_{\mathscr{Z}\times\mathbb{F}}(\mathsf{z},\mathsf{w}) = \|\mathsf{w}_i - \mathsf{z}\|^2$ is the square Euclidean distance.

An important drawback of k-means is that is highly dependent on initialization. Moreover, even if the algorithm is guaranteed to converge, it often gets stuck in local minima of $\mathscr{E}_{W^*}$. To get out of the local minima a so called "swap" procedure is used (line 7 to 25, alg. 2). One cluster is chosen, either randomly or because of its short distance to a different cluster with more elements. Then, a simulation is done by reallocating that reference vector to the region of space with maximum error. If the resulting partition has a lower clustering distortion, the result is accepted and a new simulation is done. Otherwise, the result is rejected and the procedure stops. A reference vector metric is used to evaluate the similarity between clusters: $d_{\mathbb{F}} : (\mathbb{F}\times\mathbb{F}) \longrightarrow \mathbb{R}^+$. If $\mathscr{Z} = \mathbb{F}$ it is possible to use $d_{\mathbb{F}} = d_{\mathscr{Z}\times\mathbb{F}}$, to compute both the distance between clusters and the distortion between a datum and its corresponding cluster (line 16, alg. 2).

It is worth noting that this clustering framework naturally defines a hierarchy: a cluster is the parent of all the clusters of the finer scale that are closer to it than to any other cluster (see fig. 6).

5.2.1 K-Means Extension for Gaussian Inputs

In order for the covariance matrices of the clusters at the coarse scale to be as accurate as possible, we need to use the information provided by the covariance matrices at the finer scale. Therefore, we need a clustering algorithm that is able to properly handle Gaussian input data $\mathscr{Z} = \mathbb{F} = \mathscr{G}^3 \triangleq \{(\mu,\Sigma) | \mu \in \mathbb{R}^3 \text{ and } \Sigma \text{ is SDP}\}$ [1]. Davis [15] has proposed such an algorithm, proving that it converges. The algorithm uses the Kullback-Leibler divergence (Eq. 8) as a distance function $d_{\mathscr{Z}\times\mathbb{F}}$ for Gaussians:

$$D_{KL}(\mathsf{z}\|\mathsf{w}) = \frac{1}{2}\left[(\mu_{\mathsf{z}} - \mu_{\mathsf{w}})^T \Sigma_{\mathsf{w}}^{-1} (\mu_{\mathsf{z}} - \mu_{\mathsf{w}}) + \log\left(\frac{\det\Sigma_{\mathsf{w}}}{\det\Sigma_{\mathsf{z}}}\right) + \mathrm{Tr}\,\left(\Sigma_{\mathsf{z}}\Sigma_{\mathsf{w}}^{-1}\right) - d\right] \tag{8}$$

where d is the dimension. The metric is composed of three terms corresponding to the Mahalanobis distance, the volume and the orientation, respectively.

The use of this metric in clustering means that the decision of grouping fine scale clusters together does not only depend on their positions, but also on their sizes and orientations. This property is particularly important for mapping, since it will tend to preserve sharp features such as corners and edges because the distance between the

[1] SDP matrices are a subset of SSDP matrices, meaning that analysis tools such as those proposed for NDT [6], tensor voting [5] and quadric error [11] approaches, may be applied to them.

linear components of such features will increase with the angle difference between them.

As explained in [15] the computation of the optimal reference vectors from a set of Gaussians $\{z_j = (\mu_{z_j}, \Sigma_{z_j}) | j = 1, \ldots\}$ weighted by positive reals (α_{z_j}), is done in two steps. First the optimal mean is computed using (9):

$$\mu^* = \frac{1}{\sum_j \alpha_{z_j}} \sum_j \alpha_{z_j} \mu_{z_j} , \tag{9}$$

then the covariance matrix is given by (10):

$$\Sigma^* = \left[\frac{1}{\sum_j \alpha_{z_j}} \sum_j \alpha_{z_j} (\Sigma_{z_j} + {\mu_{z_j}}^T \mu_{z_j}) \right] - (\mu^*)^T \mu^* . \tag{10}$$

It is interesting to remark that, if the weights are defined as the number of samples used to compute the Gaussians at the fine scales, then the optimal Gaussian is given by the sample mean and covariance of the fine samples that compose the cluster.

5.3 Cluster Deletion

In order to account for dynamic objects that have been detected once and that are not there anymore, we delete those clusters whose occupation has fallen below a given threshold. As for cluster insertion, the remaining Gaussians are adjusted by running the clustering algorithm.

5.4 Map Merging and Simplification

Now, we explain how to merge two maps whose cells contain multiple Gaussian clusters. This is a form of map simplification since the goal is to delete redundant cluster centers after the union of maps.

Merging is performed through straightforward application of the clustering algorithm to every cell that appears as occupied in both maps. In order to fix the number of clusters for a given cell, we select the maximum number in the two maps. This, of course, can be later refined as described above. Thus, the main problem is the initialization of the clustering algorithm.

The idea is to take all the clusters of both maps, then to compute the inter-cluster divergences as in line 14 of Alg. 2. From there, the algorithm proceeds by replacing by a single cluster, the pair of clusters that are separated by the smallest distance, and then starting over until the target number of clusters is reached. This is done using equations 9 and 10. The procedure is very efficient because no access to the finer scales is required. After finishing the merging step, a run of the clustering algorithm is executed to smooth out the result.

It is worth noting that the same process can be used to simplify the map when a reduction in the number of clusters is required.

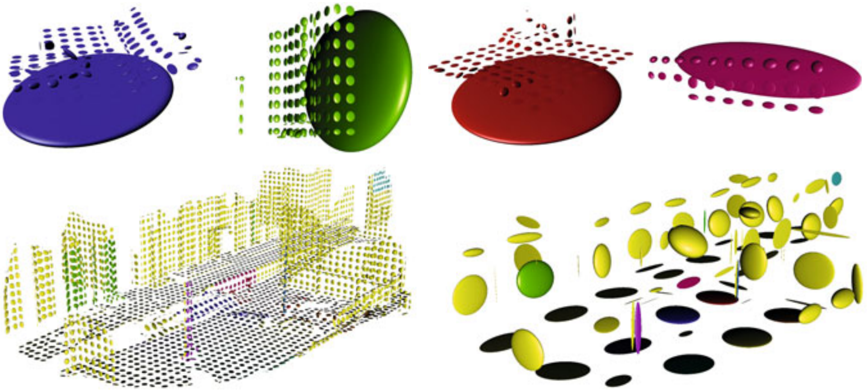


Fig. 6 Cluster hierarchy. Up: coarse clusters are translated for better display.

6 Results

We use the Hausdorff distance to evaluate the results. This metric may be seen as the worst-case distance. For two sets of points, it is obtained by computing, for every point in one set, the distance to closest point in the other set, and then taking the maximum of these distances. In our case, we compute this distance by sampling Gaussians and rejecting points that fall outside the cells. The samples are directly measured against point clouds in the case of real data. In the case of simulated data, the ground truth is available in the form of the original mesh; the corresponding points are obtained by representing each triangle with a number of samples. In all cases, the number of samples is proportional to the size of the object being considered: the volume of the ellipsoids in the case of Gaussians and the area in the case of triangles.

For each scale the cell side is ten times larger than the cell side at the finer scale. For the 2-D data set (fig. 7) the finest side is 5 cm and for the 3-D data set the finest side is 10 cm. Regarding the algorithm approach modules (Fig. 3) we set the occupancy computation to take place at acquisition rate for two dimensional data and every three acquisitions for the three dimensional case. As for refinement, the algorithm has been configured to insert 4 clusters of the remaining Gaussian budget, every 10 acquisitions.

The results we have obtained on real and simulated data sets exhibit similar advantages and drawbacks:

- Advantages:
  - *Accuracy vs map size:* the method is able to significantly improve accuracy with a relatively small increase in the size of the model. In our experiments increasing the number of clusters by four leads to a factor of three reduction of the Hausdorff distance.

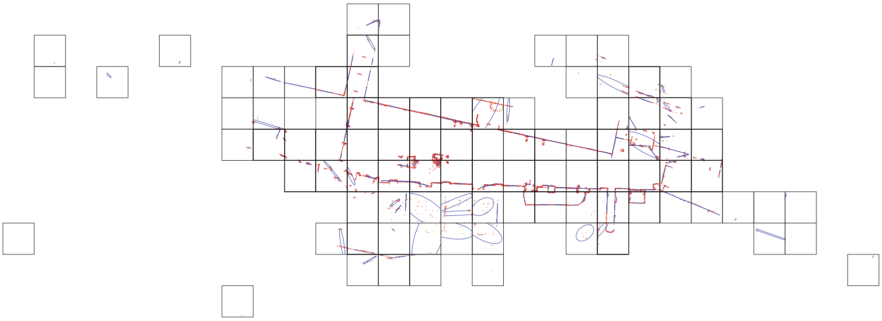


Fig. 7 Real 2-D dataset from [16]. The coarsest (blue ellipsoids) and finest (red ellipsoids) maps are displayed. Black squares represent the coarse cells. All ellipsoids correspond to the covariance matrices plotted with three times the square of the eigenvalues.

  - *Multi-scale error reduction:* the huge size reduction ratio (10^4 to 1 in 2-D and 10^8 to 1 in 3-D) between the finest and the coarsest scales is kept by the refinement process, while considerably improving accuracy. For instance, when refining the coarsest map in the 3-D data set, we increase the number of clusters from 53 to 181 and reduce the error by 3. Note that large flat clusters remain (in particular on the ground and on the walls) while a lot of detail clusters are added at poles and corners (fig. 1). This could not have been done by simply adding an intermediate scale.

- Drawbacks: the main problem we have detected is that, sometimes, the Hausdorff distance does not have a significant decrease when a cluster is added. We believe that there are two reasons for this: first, an aliasing phenomenon that arises from the fact that the underlying cells force the algorithm to artificially cut a big object in pieces, some of which can be very small with respect to other Gaussians in the same cell, leading to big error contributions because of the small size of the covariance. The second reason is that, when the 'right' number of clusters is not yet reached, the resulting Gaussian may represent two separate clusters, yielding a smaller but still important value for the Hausdorff distance.

7 Conclusions and Future Work

In this paper we have proposed a comprehensive framework to build two and three-dimensional maps from range data. The proposed representation enhances the accuracy of previous approaches by enabling the presence of several Gaussians per cell. These Gaussians are added by means of a refinement algorithm which inserts them where the representation error is maximum. The algorithm makes use of a recent Gaussian clustering approach that uses the Kullback-Leibler divergence as a distance function, thanks to this, our algorithm is able to preserve important features of the environment (*e.g.* corners) that are usually smoothed out by other approaches.

The framework provides a theoretically sound foundation for map merging. In order to deal with moving objects and noise, our approach makes use of occupancy to determine when to delete parts of the map that have become empty, as well as to adjust the plasticity of the map. Experiments with real and simulated data show that, for coarse scales, significant accuracy gains may be obtaining by a small augmentation in the number of clusters. Moreover, when compared with existing approaches, the additional computational cost that is required to insert these clusters is marginal.

Further work includes working towards real time mapping of huge streams of 3-D points by exploiting parallelization and hierarchical multi-scale update. Middle term research will be directed to exploring the application of our approach to higher dimensional spaces that include point properties such as color.

References

1. Elfes, A.: Occupancy grids: a probabilistic framework for robot perception and navigation. PhD thesis, Carnegie Mellon University (1989)
2. Yahja, A., Stentz, A.T., Singh, S., Brummit, B.: Framed-quadtree path planning for mobile robots operating in sparse environments. In: IEEE Conference on Robotics and Automation, Leuven, Belgium (May 1998)
3. Pai, D.K., Reissell, L.-M.: Multiresolution rough terrain motion planning. IEEE Transactions on Robotics and Automation 1, 19–33 (1998)
4. Ripperda, N., Brenner, C.: Marker-free registration of terrestrial laser scans using the normal distribution transform. Technical report, Institute of Cartography and Geoinformatics. University of Hannover, Germany (2005)
5. Medioni, G., Lee, M.-S., Tang, C.-K.: A Computational Framework for Segmentation and Grouping. Elsevier Science Inc., New York (2000)
6. Biber, P., Straßer, W.: The normal distributions transform: A new approach to laser scan matching. In: IEEE/RJS International Conference on Intelligent Robots and Systems (2003)
7. Grisetti, G., Stachniss, C., Burgard, W.: Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling. In: Proc. of the IEEE International Conference on Robotics and Automation, pp. 2443–2448 (2005)
8. Triebel, R., Pfaff, P., Burgard, W.: Multi-level surface maps for outdoor terrain mapping and loop closing. In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), Beijing, China (2006)
9. Payeur, P., Hébert, P., Laurendeau, D., Gosselin, C.: Probabilistic octree modeling of a 3-d dynamic environment. In: Proc. IEEE ICRA 1997, Albuquerque, NM, April 20-25, pp. 1289–1296 (1997)
10. Yguel, M., Tay Meng Keat, C., Braillon, C., Laugier, C., Aycard, O.: Dense mapping for range sensors: Efficient algorithms and sparse representations. In: Proceedings of Robotics: Science and Systems, Atlanta, GA, USA (June 2007)
11. Garland, M., Heckbert, P.S.: Surface simplification using quadric error metrics. In: Computer Graphics. Annual Conference Series, vol. 31, pp. 209–216 (1997)
12. Cohen-Steiner, D., Alliez, P., Desbrun, M.: Variational shape approximation. In: ACM SIGGRAPH 2004 Papers, pp. 905–914 (2004)

13. Pauly, M., Gross, M., Kobbelt, L.P.: Efficient simplification of point-sampled surfaces. In: VIS 2002: Proceedings of the conference on Visualization 2002, pp. 163–170. IEEE Computer Society, Washington, DC, USA (2002)
14. Lloyd, S.: Least squares quantization in pcm. IEEE Transactions on Information Theory 28(2), 129–137 (1982)
15. Davis, J.V., Dhillon, I.: Differential entropic clustering of multivariate gaussians. In: Schölkopf, B., Platt, J., Hoffman, T. (eds.) Advances in Neural Information Processing Systems, vol. 19, pp. 337–344. MIT Press, Cambridge (2007)
16. Cyrill Stachniss. Corrected robotic log-files, http://www.informatik.uni-freiburg.de/∼stachnis/datasets.html

Robust 3-D Visual SLAM in a Large-Scale Environment

Jungho Kim, Kuk-Jin Yoon, and In So Kweon

Abstract. Motion estimation approaches enable the robust prediction of successive camera poses when a camera undergoes erratic motion. It is especially difficult to make robust predictions under such conditions when using a constant-velocity model. However, motion estimation itself inevitably involves pose errors that result in the production of an inconsistent map. To solve this problem, we propose a novel 3D visual SLAM approach in which both motion estimation and stochastic filtering are performed; in the proposed method, visual odometry and Rao-blackwellized particle filtering are combined. First, to ensure that the process and the measurement noise are independent (they are actually dependent in the case of a single sensor), we simply divide observations (i.e., image features) into two categories, common features observed in the consecutive key-frame images and new features detected in the current key-frame image. In addition, we propose a key-frame SLAM to reduce error accumulation with a data-driven proposal distribution. We demonstrate the accuracy of the proposed method in terms of the consistency of the global map.

1 Introduction

Simultaneous localization and mapping (SLAM) is the technique of building up a map of the unknown environment while simultaneously keeping track of the current position of the cameras or robots in the environment. This problem has attracted immense attention in the robotics and computer vision communities. To solve the SLAM problem, many methods based on the recursive estimation of the posterior distribution have been introduced. Davison [1] successfully performed monocular

Jungho Kim · In So Kweon
KAIST, 373-1 Guseong-dong, Yuseong-gu, Daejeon, Korea
e-mail: jhkim@rcv.kaist.ac.kr, iskweon@kaist.ac.kr

Kuk-Jin Yoon
GIST, 261 Cheomdan-gwagiro, Buk-gu, Gwangju, Korea
e-mail: kjyoon@gist.ac.kr

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 519–534.
springerlink.com

SLAM by employing an extended Kalman filter (EKF) and adopting an initialization process to determine the depth of the 3D landmarks by using the particle filter-type approach. Eade et al. [2] utilized the FastSLAM-type particle filter in single-camera SLAM to manage a greater number of landmarks because the computational requirements of EKF-based SLAM approaches rapidly grow with the number of landmarks. In [3], the authors described a visual SLAM algorithm that is robust to erratic camera motion and visual occlusion by using efficient scale prediction and examplar-based feature representations in conjunction with the use of an unscented Kalman filter (UFK). Recently, Eade et al. [4] proposed a monocular SLAM system in which map inconsistencies can be prevented by coalescing observations into independent local coordinate frames, building a graph of the local frames, and optimizing the resulting graph. Paz et al. [5] presented a 6-degree-of-freedom (DOF) visual SLAM system based on conditionally independent local maps by using a stereo camera as the only sensor. Here, it is worth noting that in most 3D visual SLAM approaches, a constant-velocity model is employed to achieve the independence between the process and the measurement noise. However, in the constant velocity model, when cameras undergo sudden motion, these SLAM approaches are highly prone to failure, resulting in inconsistencies in the global map. In [6], the authors combined the particle filter-based localization with the UKF-based SLAM problem to cope with erratic camera motion while maintaining a small number of landmarks.

On the other hand, in the vision community, structure-from-motion (SFM) approaches have been studied independent of SLAM to estimate camera trajectories by using only a sequence of images. For example, Nister et al. introduced 'visual odometry' that estimates the relative movements of the stereo head in the Euclidean space [7]. Recently, Zhu et al. [8] developed a helmet-based visual odometry system that consists of two pairs of stereo-cameras mounted on a helmet; one pair faces forward while the other faces backward. By utilizing the multi-stereo fusion algorithm, they improved the overall accuracy in pose estimation. Here, we should note that in many previous studies, optimization techniques such as bundle adjustment [9, 10] have been adopted to avoid inconsistencies in the global map. However, it is not feasible to perform conventional bundle adjustment in on-line approaches because the computational cost rapidly grows with the number of 3D landmarks and their observations (the image coordinates over the sequence).

2 Motivation

SLAM approaches are described by two probabilistic models — the process and measurement models [11]. The current state, x_t, is governed by the probabilistic function of the previous state x_{t-1}, the control input u_t, and the additive process noise w_t as follows :

$$x_t = f\left(x_{t-1}, u_t, w_t\right). \tag{1}$$

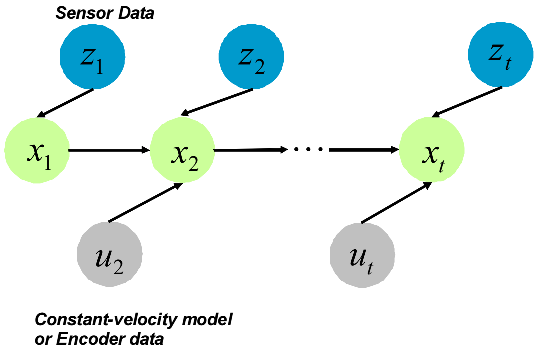


Fig. 1 Bayesian network that describes the relations between process and measurement models

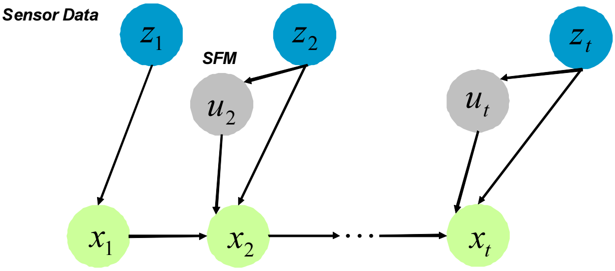


Fig. 2 Bayesian network that describes dependency between process and measurement noise

The measurements z_t are also governed by the probabilistic function of the current state and the additive measurement noise v_t. We re-estimate the posterior distribution of the state produced from the process model by using the measurement model as follows :

$$z_t = h(x_t, v_t) . \tag{2}$$

where w_t and v_t represent the process and measurement noise, respectively; they are assumed to be independent, as shown in Fig. 1.

In many visual SLAM approaches, a constant-velocity model that is independent of sensor data is employed [1, 2, 3] or another sensor such as a wheel encoder [12], or IMU [13] is used for the process model. However, if a camera undergoes sudden motion, a constant-velocity model is not valid, and wheel encoders cannot be used to obtain the 6-DOF poses of the camera. In contrast, motion estimation approaches can be adopted to obtain good estimates for the 6-DOF pose under erratic camera motion.

However, on the other hand, if we use motion estimation methods for the process model, the control input, u_t, directly depends on measurements, and the independence assumption is no longer valid, as shown in Fig. 2. Moreover, SLAM approaches involve some problems related to dimensionality when estimating the 6-DOF camera pose and 3D landmarks, and scalability caused by the large number of landmarks obtained during long-distance navigation.

The contributions of the proposed approach compared to the previous methods can be summarized as follows:

(i) We achieve the independence between the process and measurement noise when using conventional motion estimation approaches for the process model in SLAM. (Section 4.2).

(ii) We use the key-frame SLAM approach to reduce the number of camera poses to be estimated in the path by generating the poses only at key-frame locations. In addition, we propose a method that effectively updates the posterior distribution of the camera path by using many observations obtained at non key-frame locations. (Section 4.3).

(iii) We use a data-driven proposal distribution computed using the RANSAC to efficiently represent the posterior distribution of the camera pose by a limited number of particles. (Section 4.5)

(iv) We develop a novel SLAM approach by integrating the above contributions to reduce the error accumulation involved in conventional motion estimation approaches.

3 The Visual Odometry System

Our visual odometry system consists of a few sub-components. Fig. 3 shows the overall procedure followed in our visual odometry system.

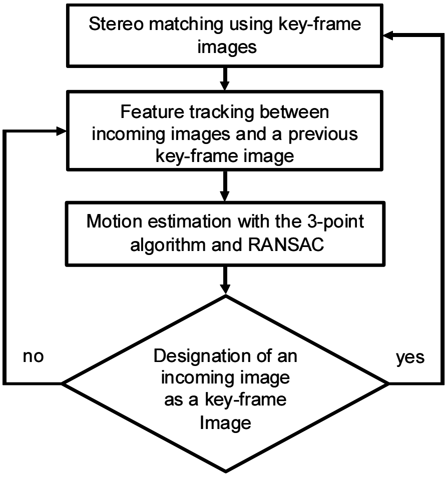

Stereo matching using key-frame images
Feature tracking between incoming images and a previous key-frame image
Motion estimation with the 3-point algorithm and RANSAC
Designation of an incoming image as a key-frame Image
no
yes

Fig. 3 Overall procedure followed in our visual odometry system

3.1 Feature Extraction and Stereo Matching

For each stereo pair, we first extract corner points in the left image and then apply the 1D KLT feature tracker [14] to stereo images to obtain correspondences. The 3D coordinates of the matched corner points are used for map building and for motion estimation.

3.2 Motion Estimation

As mentioned previously, Nister et al. [7] introduced a method called 'visual odometry' for the real-time estimation of the movement of a stereo head or a single camera. In this approach, the 3-point algorithm [15] is employed: the images of three known world points are used to obtain up to four possible camera poses and more than three points are required to automatically obtain one solution. Here, we also employ the RANSAC [16] where a set of 3 world points and their image coordinates are randomly selected to compute the relative camera pose. The estimated pose is evaluated from other correspondences.

3.3 Key-Frame Designation

The number of tracked corners between an incoming image and a previous key image is a measure that is used to determine the key-frame locations. If the number of points tracked by the KLT tracker [17] is smaller than a pre-defined threshold, we designate an incoming image as a key-frame image and estimate the relative pose w.r.t the previous key-frame. This strategy can partially prevent error accumulation because we determine the relative pose w.r.t the previous key-frame pose.

4 Stochastic Filtering

4.1 Rao-Blackwellized Particle Filter

We initially employ a Rao-Blackwellized particle filtering technique [18, 19], by which a SLAM problem is decomposed into a localization problem and a collection of landmark estimation problems that are conditioned on pose estimates as follows:

$$\begin{aligned} p\left(x_{1:t}, M | z_{1:t}, d_{1:t}\right) &= \frac{p\left(x_{1:t}, M, z_{1:t}, d_{1:t}\right)}{p\left(z_{1:t}, d_{1:t}\right)} \\ &= p\left(x_{1:t} | z_{1:t}, d_{1:t}\right) p\left(M | x_{1:t}, z_{1:t}, d_{1:t}\right). \end{aligned} \tag{3}$$

where $x_{1:t}$ and M represent the camera path and a collection of 3D landmarks, respectively. $z_{1:t}$ and $d_{1:t}$ indicate the observations and data association up to t.

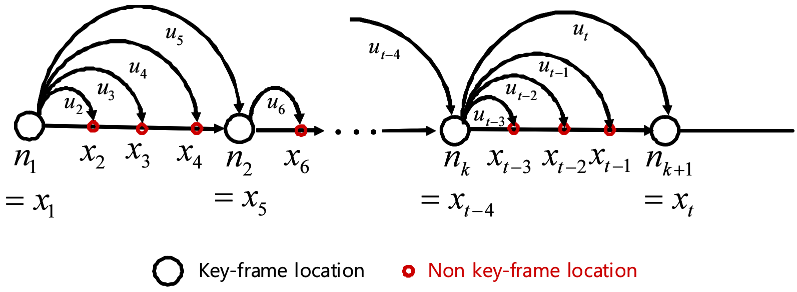


Fig. 4 A sequence of frames divided according to key-frame and non key-frame locations

In FastSLAM [19], the path estimator is implemented using the particle filter. The landmark estimator is implemented using a Kalman filter, with a separate filter for different landmarks, because all the 3D landmarks are independent of each other with a given path as shown in Eq. (4).

$$p\left(M|x_{1:t}, z_{1:t}, d_{1:t}\right) = \prod_{l=1}^{L} p\left(m_l|x_{1:t}, z_{1:t}, d_{1:t}\right) \tag{4}$$

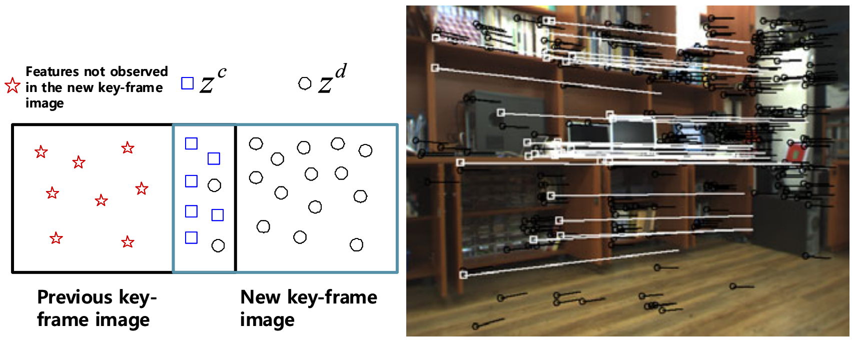
where m_l represents each landmark in M.

4.2 Independence between Process and Measurement Noise

We estimate the camera path that consists of a sequence of camera poses at only the key-frame locations instead of all the frames, as mentioned in Section 3.3. In other words, we estimate the posterior distribution $p\left(n_{1:k}|z_{1:t}, d_{1:t}\right)$ instead of $p\left(x_{1:t}|z_{1:t}, d_{1:t}\right)$. When designating an incoming image as a key-frame image, we elongate the path, $n_{1:k+1}$, by adding the relative pose, u_t, w.r.t the last key-frame pose, n_k, to the previous path, $n_{1:k}$, as shown in Fig. 4. We first divide the observations into two categories: observed features common to two key-frame images z^c and newly detected features in the current key-frame image z^d $(= z - z^c)$, as shown in Fig. 5. z^c are used to evaluate the posterior distribution of the path $n_{1:k}$, and z^d are used to estimate the relative pose, u_t. Thus, we have

$$p\left(n_{1:k}|z_{1:t}, d_{1:t}\right) = p\left(n_{1:k}|z^c_{1:t}, z^d_{1:t}, d_{1:t}\right) \tag{5}$$

We then achieve the independence between the process and measurement noise by simply dividing the observations instead of using another sensor, as shown in Fig. 6.



(a) Consecutive key-frame images

(b) Black circles and white rectangles represent features belonging to z_t^d and z_t^c, respectively

Fig. 5 Achieving independence between process and measurement noise by dividing observations according to purpose

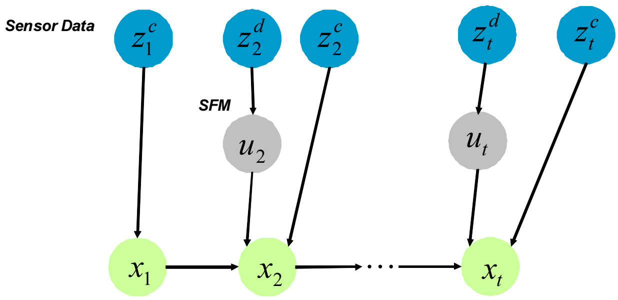


Fig. 6 Bayesian network that describes the independence between process and measurement noise in the case of divided observations

4.3 Path Estimation for Key-Frame SLAM

The posterior distribution of $n_{1:k}$ with given $z_{1:t}$ and $d_{1:t}$ (correspondences between the landmarks, M, and observations, $z_{1:t}^c$) is represented as a weighted set of particles:

$$p\left(n_{1:k}|z_{1:t}^c, z_{1:t}^d, d_{1:t}\right) = \sum_i p\left(n_{1:k}^i|z_{1:t}^c, z_{1:t}^d, d_{1:t}\right)\delta\left(n_{1:k} - n_{1:k}^i\right) \tag{6}$$

where $\delta(x)$ represents the Dirac delta function that returns 1 if x is zero, and 0 otherwise.

For non key-frame locations, we re-estimate the posterior distribution of the camera path $n_{1:k}$ by marginalizing the relative pose u_t using Eq. (7).

$$\begin{aligned} p\left(n^i_{1:k}|z^c_{1:t}, z^d_{1:t}, d_{1:t}\right) &= \int p\left(n^i_{1:k}, u_t|z^c_{1:t}, z^d_{1:t}, d_{1:t}\right) du_t \\ &= \sum_j p\left(n^i_{1:k}, u^j_t|z^c_{1:t}, z^d_{1:t}, d_{1:t}\right) \end{aligned} \tag{7}$$

The posterior distribution of the relative pose is represented by a set of particles coming from the RANSAC, where relative poses are estimated by selecting multiple sets of minimal correspondences. Each pair of the minimal set provides a single hypothesis on the relative pose, and its weight is computed according to the number of inliers among all correspondences. This will be described in Section 4.6. We compute the joint probability of a camera path, $n^i_{1:k}$, and a relative pose, u^j_t, using Eq. (8).

$$\begin{aligned} p\left(n^i_{1:k}, u^j_t|z^c_{1:t}, z^d_{1:t}, d_{1:t}\right) &= \frac{p\left(n^i_{1:k}, u^j_t, z^c_t, z^d_t, d_t, z^c_{1:t-1}, z^d_{1:t-1}, d_{1:t-1}\right)}{p\left(z_{1:t-1}, d_{1:t-1}, z_t, d_t\right)} \\ &= \eta\, p\left(z^c_t|n^i_{1:k}, u^j_t, d_t\right) p\left(u^j_t|z^d_t\right) p\left(n^i_{1:k}|z^c_{1:t-1}, z^d_{1:t-1}, d_{1:t-1}\right) \end{aligned} \tag{8}$$

where $p(z^c_t|n^i_{1:k}, u^j_t, d_t)$ is a likelihood and $p(u^j_t|z^d_t)$ is the posterior distribution of the relative pose, as defined by Eq. (11). $p(n^i_{1:k}|z_{1:t-1}, d_{1:t-1})$ is the previous posterior distribution up to $t-1$, and η is a normalization term that makes the sum of all probabilities 1.

In our key-frame SLAM approach, we can reduce the number of camera poses to be estimated in the path and update the particles of the camera path whose poses are generated at the key-frame locations by using many observations obtained at the non key-frame locations.

4.4 Likelihood Estimation

The likelihood estimation is based on the number of inliers for each particle of the camera pose. It is computed by examining how many scene points m_l are projected close to relevant measurements z^l_t, as defined in Eq. (9).

$$\begin{aligned} p\left(z^c_t|n^i_{1:k}, u^j_t, d_t\right) &= \sum_{l=1}^{L} d\left(z^l_t, m_l, n^i_{1:k}, u^j_t\right)/L \\ d\left(z^l_t, m_l, n^i_{1:k}, u^j_t\right) &= \begin{cases} 1 & \text{if } \left\|z^l_t - z\left(m_l, \left(n^i_{1:k} \oplus u^j_t\right)\right)\right\| < e_l \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{9}$$

where $(n^i_{1:k} \oplus u^j_t)$ indicates the global pose of the camera computed from the path $n^i_{1:k}$ and the relative pose u^j_t. $d(z^l_t, m_l, n^i_{1:k}, u^j_t)$ indicates whether the point m_l is an inlier or outlier with respect to the observation z^l_t and the camera pose $(n^i_{1:k} \oplus u^j_t)$, and $z(m_l, (n^i_{1:k} \oplus u^j_t))$ is the projection of a scene point m_l for a particular camera pose $(n^i_{1:k} \oplus u^j_t)$. L is the number of scene points that are associated with the current measurements, as defined by d_t, and e_l represents the uncertainty in the projection of the 3D scene point (see Section 4.8).

4.5 Outlier Rejection

We eliminate the outliers z^o_t among z^l_t that are not supported by any particles in the computation of the likelihood values as shown in Eq. (10).

$$z^o_t = \left\{ z^l_t | \sum_i \sum_j d\left(z^l_t, m_l, n^i_{1:k}, u^j_t\right) = 0 \right\} \tag{10}$$

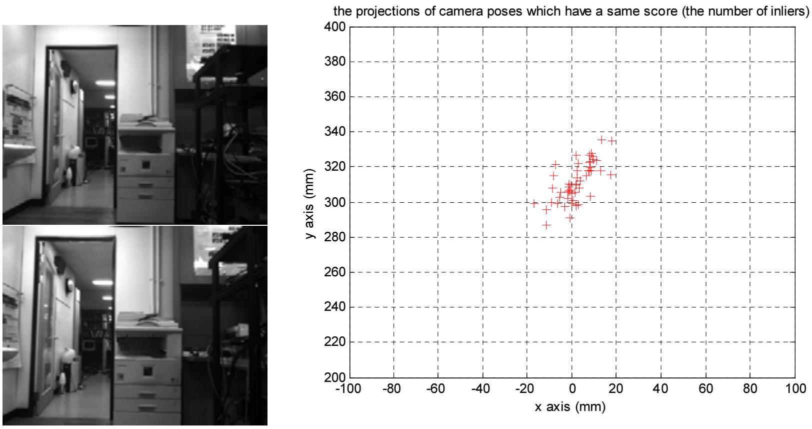
Thus, we eliminate the outliers in z^d_t using the RANSAC when estimating the relative pose, u_t, and the outliers in z^c_t when computing the likelihood values.

4.6 Data-Driven Proposal Distribution

It is especially insufficient to represent the posterior distribution using the limited number of particles in the 6-dimensional space. In our approach, we use multiple hypotheses that are generated in the RANSAC step. The RANSAC is an efficient technique for determining a good hypothesis, but unfortunately the hypothesis selected with the best score (the number of inliers) does not always correspond to the correct estimate. Therefore, in our approach, instead of selecting an unique hypothesis, we propagate multiple reasonable hypotheses to the subsequent frames to re-estimate the posterior distribution by using more observations. Fig. 7 shows the projections of 3D camera poses that have the best score, i.e., the maximum number of inliers computed using the RANSAC. We represent the posterior distribution of the relative pose using these hypotheses and their weights according to the number of inliers, as shown in Eq. (11).

$$p\left(u^j_t | z^d_t\right) \propto \frac{N^j_{inlier}}{N_{total}}, \quad \sum_j p\left(u^j_t | z^d_t\right) = 1 \tag{11}$$

where N^j_{inlier} is the number of inliers for u^j_t, and N_{total} is the total number of correspondences in z^d_t. This means that the multiple hypotheses on the camera pose generated by the RANSAC are probabilistically evaluated by using more incoming observations than just two views.



(a) Two consecutive images (b) The top-down view of the 3D camera poses which have the same score (the number of inliers)

Fig. 7 Hypotheses of the camera pose that have the same score when using the RANSAC

4.7 Path Generation

Whenever we have a new key-frame image, we elongate the path using the previous posterior distribution of the camera trajectory and the relative pose as follows:

$$\begin{gathered} n_{1:k+1}^{N_j\times i+j} \leftarrow \left\{ n_{1:k}^{i}, \left(n_{1:k}^{i} \oplus u_t^{j} \right) \right\}, \\ p\left(n_{1:k+1}^{N_j\times i+j} | z_{1:t}, d_{1:t} \right) \propto p\left(u_t^{j} | z_t^{d} \right) p\left(n_{1:k}^{i} | z_{1:t}, d_{1:t} \right), \end{gathered} \qquad (12)$$

where N_j is the number of particles for the relative pose. Here, before adding the relative pose to the particles of the camera path, we prune some hypotheses on the camera path on the basis of their weights. In our implementation, only the 10 best particles remain.

4.8 Landmark Estimation

When designating an incoming image as a key-frame image, we update the posterior distributions of the landmarks. We model the posterior distribution of each landmark $p(m_l|n_{1:k}, z_{1:k}^{l}, d_{1:k})$ defined in Eq. (4) using a optimized 3D landmark location, $\hat{m}_l$, and its uncertainty ,e_l, in the image space; we re-triangulate 3 observations (first two stereo views and the last view) corresponding to each landmark for each particle of the camera path by using SVD [9] to compute $\hat{m}_l$, as shown in Fig. 8, and e_l is determined by the projection error of $\hat{m}_l$ for the pose of the last key-frame image n_N, as shown in Eq. (13).

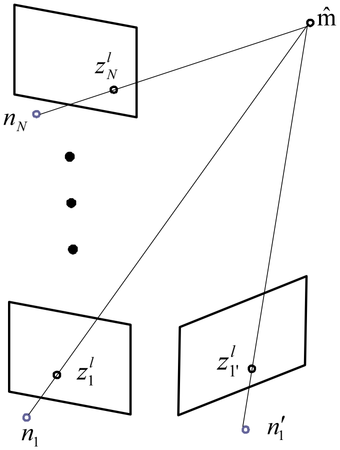


Fig. 8 Landmark update by re-triangulating observations with a given camera path

$$e_l = \left\| z_N^l - z(\hat{m}_l, n_N) \right\| + e_0 \tag{13}$$

where N is the number of observations at the key-frame locations for each landmark, and e_0 is a pre-defined initial projection uncertainty of the landmark.

5 Experimental Results

For experiments, we used a stereo camera with a 12*cm* baseline and a 6*mm* lens, which provide a narrow field of view. The resolution of the images is 320 × 240 pixels. Fig. 9 shows the path obtained by using visual odometry (red line) and the hypotheses on the trajectory computed by using the proposed SLAM method (blue lines). For this experiment, we captured 552 images by driving a robot over a distance of 10*m* in an indoor environment and to evaluate the performance, we added the first image to the end of the image sequence so that the initial and final locations are identical. At the final location, we choose the path $n_{1:k}^m$ and the relative pose u_t^n that maximize the posterior distributions as follows:

$$\begin{aligned} m &= \underset{i}{\operatorname{argmax}}\, p\left(n_{1:k}^i | z_{1:t}, d_{1:t}\right), \text{ where } p\left(n_{1:k}^i | z_{1:t}, d_{1:t}\right) = \sum_j p\left(n_{1:k}^i, u_t^j | z_{1:t}, d_{1:t}\right) \\ n &= \underset{j}{\operatorname{argmax}}\, p\left(u_t^j | z_{1:t}, d_{1:t}\right), \text{ where } p\left(u_t^j | z_{1:t}, d_{1:t}\right) = \sum_i p\left(n_{1:k}^i, u_t^j | z_{1:t}, d_{1:t}\right) \end{aligned} \tag{14}$$

Table 1 lists the errors of the final camera pose for visual odometry and the proposed method. Fig. 11(a) shows the path computed by visual odometry and the corresponding global map obtained by integrating the structures computed by stereo matching over time. In this experiment, many images (102 from the 312 images)

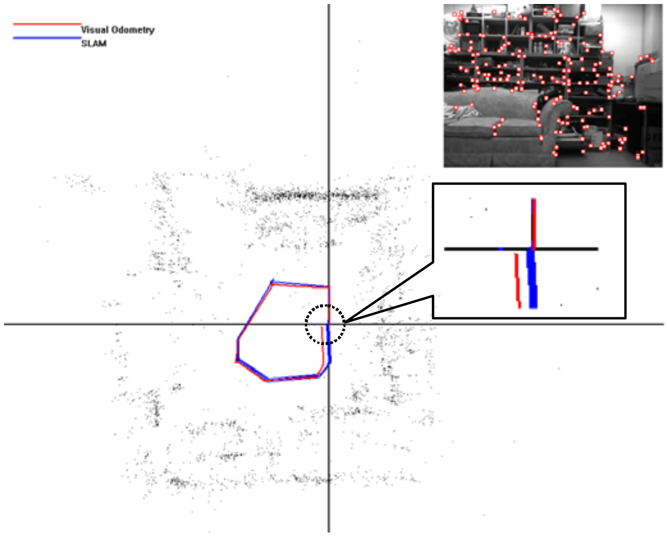


Fig. 9 Camera trajectories estimated by visual odometry only (red line) and by using the proposed method (blue lines)

Table 1 Pose errors for visual odometry and the proposed method (*mm*)

	x	y	z	total
visual odometry only	-182.08	-147.70	-50.50	239.83
proposed	-53.67	5.33	30.57	61.99

were influenced by motion blur because we captured images by using a hand-held stereo camera that underwent erratic motion; the images are shown in Fig. 10. In addition, a camera re-visited the scene where it first observed. We can easily find the inconsistency in the global map caused by error accumulation of the path. However, the map produced by using the proposed method is more consistent than that obtained by visual odometry, as shown in Fig. 11(b). Here, we randomly selected 500 sets of 3 correspondences to obtain 500 hypotheses of which we only chose a maximum of 50 hypotheses on the basis of their weights (the number of inliers). Fig. 12(a) shows the map computed by visual odometry using the same sequence of images. In this experiment, we generated only 300 hypotheses of which we retained a maximum of 50 hypotheses on the basis of their weights. Because the number of hypotheses is small, this map has a larger error than the previous result. However, we can compensate for error accumulation by using the proposed method, as shown in Fig. 12(b). Moreover, we can observe that the results obtained by using the proposed method are more consistent than those obtained by visual odometry because the proposed method is not strongly affected by randomness. Fig. 13 (a) and (b) show the global maps and the camera paths computed by visual odometry and by using the proposed method, respectively. For this experiment, we captured more

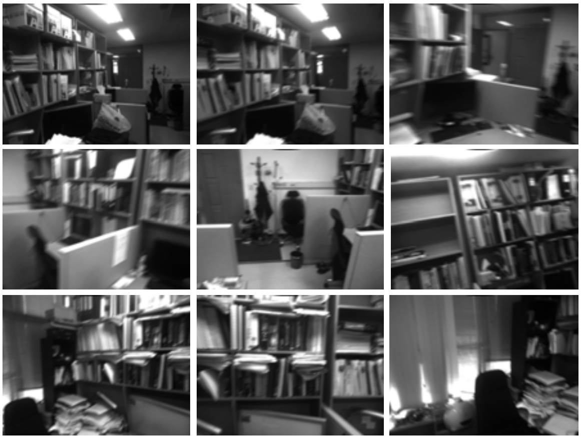
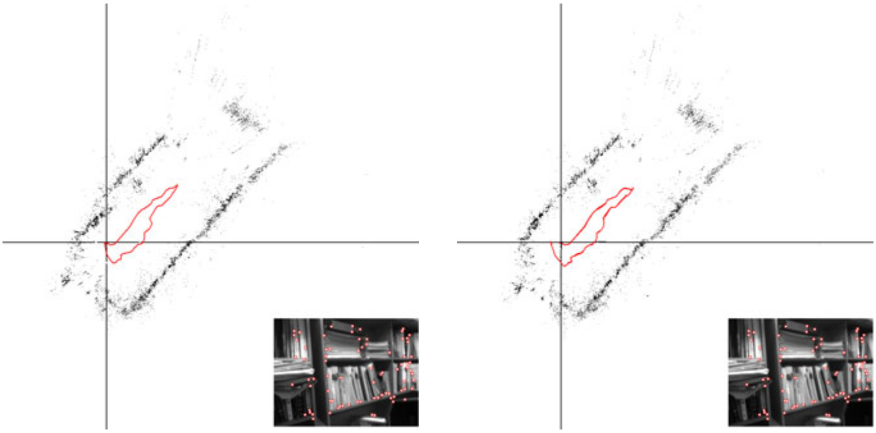


Fig. 10 Some images affected by motion blur

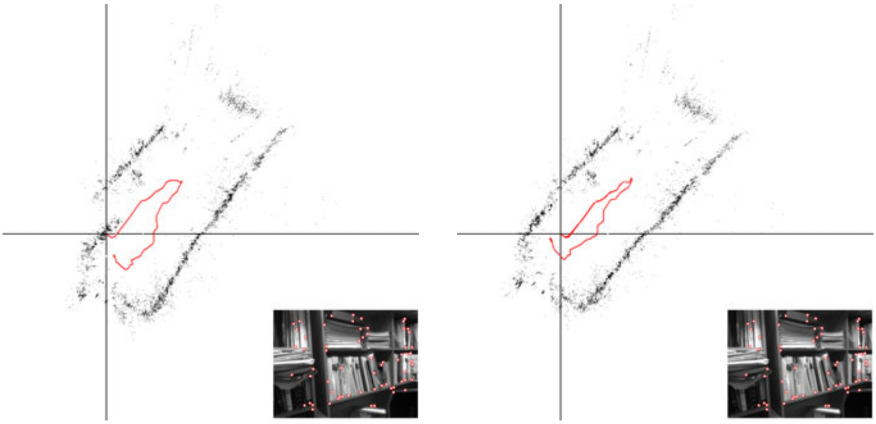


(a) The top-down view of the 3D map and camera trajectory estimated by visual odometry

(b) The top-down view of the 3D map and camera trajectory estimated by the proposed method

Fig. 11 The top-down view of the 3D map and camera trajectories obtained for 500 hypotheses in the RANSAC step

than 3000 images while walking more than 200 *m* in the outdoor environment with the stereo camera in hand. We can see that the results obtained with the proposed visual SLAM approach, in which the visual odometry and stochastic filtering are combined, are much better than those obtained by only visual odometry. To evaluate the consistency, we overlap the maps and paths with the corresponding google map as shown in Fig 13(c) and (d). The proposed SLAM algorithm can process approximately 10 frames per second when using a 2.4 GHz CPU. Table 2 lists the computational complexities for visual odometry and stochastic filtering.



(a) The top-down view of the 3D map and camera trajectory estimated by visual odometry

(b) The top-down view of the 3D map and camera trajectory estimated by the proposed method

Fig. 12 The top-down view of the 3D map and camera trajectories obtained for 300 hypotheses in the RANSAC step

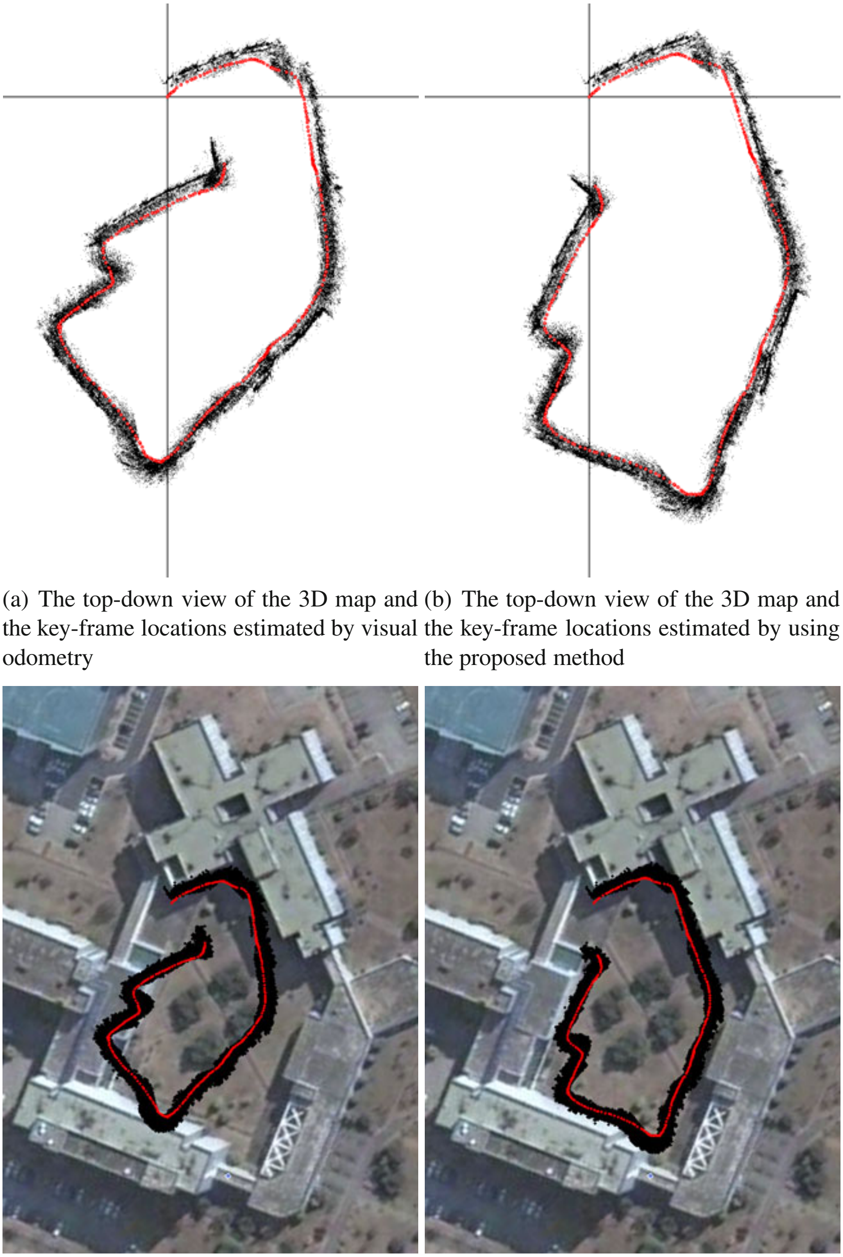
Table 2 Average processing time for visual odometry and proposed stochastic filtering when running 500 frames (ms)

operation	processing time
visual odometry (partially implemented by MMX programming)	27.655
stochastic filtering	75.733
total	103.388

6 Conclusion

We have presented a novel 3D visual SLAM method in which visual odometry and a stochastic filtering approach are combined to cope with sudden camera motion and to obtain consistent maps. To ensure that the process and the measurement noise are independent, we simply divide observations into two categories: common features observed in the consecutive key-frame images and new features detected in the current key-frame image. The proposed stochastic filtering technique can be adopted in existing motion estimation approaches to avoid error accumulation. In addition, our approach is especially efficient in the following sense:

- Dimensionality — we use a data-driven proposal distribution computed by the RANSAC approach with the 3-point algorithm to efficiently represent the posterior distribution of the camera pose.



(a) The top-down view of the 3D map and the key-frame locations estimated by visual odometry

(b) The top-down view of the 3D map and the key-frame locations estimated by using the proposed method

(c) The results obtained by visual odometry overlapped with the corresponding google map

(d) The results obtained by using the proposed method overlapped with the corresponding google map

Fig. 13 The global map and the key-frame locations for the outdoor environment

- Scalability — we reduce the number of possible camera poses in the path by formulating the key-frame SLAM, and our SLAM approach is based on the Rao-Blackwellized particle filter that can manage more landmarks than the EKF and particle filter-based approaches.

Acknowledgements. This work was supported by National Strategic R&D Program for Industrial Technology and a project for Mega City Modeling, Korea.

References

1. Davison, A.J.: Real-Time Simultaneous Localisation and Mapping with a Single Camera. In: ICCV 2003 (2003)
2. Eade, E., Drummond, T.: Scalable Monocular SLAM. In: CVPR 2006 (2006)
3. Chekhlov, D., Pupilli, M., Mayol, W., Calway, A.: Robust Real-Time Visual SLAM Using Scale Prediction and Exemplar Based Feature Description. In: CVPR 2007 (2007)
4. Eade, E., Drummond, T.: Monocular SLAM as a Graph of Coalesced Observations. In: ICCV 2007 (2007)
5. Paz, L.M., Pinié, P., Tardós, J.D., Neira, J.: Large-Scale 6-DOF SLAM With Stereo-in-Hand. IEEE Trans. on Robotics 24(5), 946–957 (2008)
6. Pupilli, M., Calway, A.: Real-Time Visual SLAM with Resilience to Erratic Motion. In: CVPR 2006 (2006)
7. Nister, D., Naroditsky, O., Bergen, J.: Visual Odometry. In: CVPR 2004 (2004)
8. Zhu, Z., Oskiper, T., Samarasekera, S., Kumar, R., Sawhney, H.S.: Real-Time Global Localization wtih A Pre-Built Visual Landmarks Database. In: CVPR 2008 (2008)
9. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge (2000)
10. Triggs, B., McLauchlan, P., Hartley, R., Fitzgibbon, A.: Bundle adjustment - a modern synthesis. In: Triggs, B., Zisserman, A., Szeliski, R. (eds.) ICCV-WS 1999. LNCS, vol. 1883, p. 298. Springer, Heidelberg (2000)
11. Dissanayake, M.W.M.G., Newman, P., Clark, S., Durrant-Whyte, H.F., Csorba, M.: A Solution to the Simultaneous Localization and Map Building (SLAM) Problem. IEEE Trans. on Robotics and Automation 17(3), 229–241 (2001)
12. Kaess, M., Dellaert, F.: Visual SLAM with a Multi-Camera Rig, GVU Technical Report, GIT-GVU-06-06 (2006)
13. Marks, T.K., Howard, A., Bajracharya, M., Cottrell, G.W., Matthies, L.: Gamma-SLAM: Using Stereo Vision and Variance Grid Maps for SLAM in Unstructured Environments. In: ICRA 2008 (2008)
14. Kim, J., Bok, Y., Kweon, I.S.: Robust Vision-based Autonomous Navigation against Environment Changes. In: IROS (2008)
15. Haralick, R., Lee, C., Ottenberg, K., Nölle, M.: Review and Analysis of Solutions of the Three Point Perspective Pose Estimation Problem. IJCV 13(3), 331–356 (1994)
16. Fischler, M., Bolles, R.: Random Sample Consensus: a Paradigm for Model Fitting with Application to Image Analysis and Automated Cartography. Communications ACM (1981)
17. Shi, J., Tomasi, C.: Good Features to Track. In: CVPR 1994 (1994)
18. Murphy, K., Russell, S.: Rao-blackwellized particle filtering for dynamic bayesian networks. Sequential Monte Carlo Methods in Practice. Springer, Heidelberg (2001)
19. Montemerlo, M., Whittaker, W., Thrun, S.: FastSLAM: A factored solution to the simultaneous localization and mapping problem. In: AAAI National Conference on Artificial Intelligence (2002)

Towards Large-Scale Visual Mapping and Localization

Marc Pollefeys, Jan-Michael Frahm, Friedrich Fraundorfer, Christopher Zach, Changchang Wu, Brian Clipp, and David Gallup

Abstract. The topic of this paper is large-scale mapping and localization from images. We first describe recent progress in obtaining large-scale 3D visual maps from images only. Our approach consists of a multi-stage processing pipeline, which can process a recorded video stream in real-time on standard PC hardware by leveraging the computational power of the graphics processor. The output of this pipeline is a detailed textured 3D model of the recorded area. The approach is demonstrated on video data recorded in Chapel Hill containing more than a million frames. While for these results GPS and inertial sensor data was used, we further explore the possibility to extract the necessary information for consistent 3D mapping over larger areas from images only. In particular, we discuss our recent work focusing on estimating the absolute scale of motion from images as well as finding intersections where the camera path crosses itself to effectively close loops in the mapping process. For this purpose we introduce viewpoint-invariant patches (VIP) as a new 3D feature that we extract from 3D models locally computed from the video sequence. These 3D features have important advantages with respect to traditional 2D SIFT features such as much stronger viewpoint-invariance, a relative pose hypothesis from a single match and a hierarchical matching scheme robust to repetitive structures. In addition, we also briefly discuss some additional work related to absolute scale estimation and multi-camera calibration.

1 Introduction

In recent years there has been a growing interest in obtaining realistic visual representations of urban environments. This has mainly been driven by the need to provide a visual and spatial context for information on the internet. While currently

Marc Pollefeys · Friedrich Fraundorfer · Christopher Zach
Institute of Visual Computing, ETH Zürich

Jan-Michael Frahm · Changchang Wu · Brian Clipp · David Gallup
Dept. of Computer Science, University of North Carolina at Chapel Hill

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 535–555.
springerlink.com

most existing commercial products are limited to aerial views, such as Google Earth and Virtual Earth/Bing Maps, or only provide 2D visualization, such as Google Street View, the most effective and flexible representation would be a photo-realistic ground-level 3D model. Besides virtual exploration, this would also support many more applications such as for example autonomous navigation, visual localization and mobile augmented reality.

There are two main types of approaches being explored for capturing realistic 3D models of large-scale urban environments. One type uses LIDAR to capture the 3D geometry and images to capture the appearance, while the other type of approach uses solely images to capture both 3D geometry and appearance simultaneously. An early example of a LIDAR-based approach is the work by Früh and Zakhor [13]. The earliest examples for image-based 3D modeling of urban scenes probably dates back about a hundred years to the origin of photogrammetry. However, only in the last decade or two has automation made it feasible to approach large scale ground-based modeling of urban scenes from images. The early approaches for automated 3D modeling from images such as the ones proposed by Tomasi and Kanade [48] and Pollefeys et al. [34] were limited to modeling more or less what could be seen in a single image. Our more recent approaches could model larger scenes that could not fully be seen from a single view-point [35], but was much too slow to use for larger scale reconstructions as processing was in the order of a minute per frame. In this paper, we will focus on our most recent approach [36], which leverages the computational power of the graphics processing unit (GPU) to recover 3D models from urban imagery at video-rate on a standard PC. The GPU is particularly well-suited to achieve high performance for many image processing tasks such as tracking or matching features [43, 59, 60, 53] and stereo matching [55, 57]. Other approaches to perform 3D reconstruction of urban scenes from images have recently been proposed, but they mostly generate simplified models without a lot of detail, e.g. [8, 54], or require human interaction [44]. Another interesting approach for obtaining visual 3D models leverages the emergence of community photo-collections [45, 25, 1]. These approaches, however, are mostly limited to landmark structures for which many photographs are available.

The simplest approach to obtain consistent maps over large scales is to use a Global Positioning System (GPS), but this can be problematic for some applications such as mapping of indoor spaces, dense urban neighborhoods or other areas where the GPS signals are weak or unavailable (GPS signals can be easily jammed). While structure-from-motion allows to obtain consistent local 3D maps, over long distances errors in position, orientation and scale accumulate. Therefore, an important challenge in large-scale reconstruction and mapping consists of obtaining self-consistent maps. One of the most important steps to achieve this is to close loops when the camera revisits the same location. Many approaches have been proposed based on SIFT [26] and other invariant features. Specific approaches have been proposed to efficiently match novel images to large number of previously acquired images [32, 12, 9, 19]. However, these approaches all rely on the ability to generate enough potential correspondences in the first place. This can be a significant problem in scenarios where the viewing angle can be very different when a place is revisited. Our approach introduced in Wu et al. [51] proposes to use the local geometric reconstruction to extract visual features on the 3D surface instead of in the images (i.e. we extract features in ortho-rectified patches). This provides viewpoint invariance and allows for direct estimation of a 3D similarity transformation from a single match, which enables robust matching even with very few correct correspondences.

The remainder of the paper is organized as follows. In Section 2 we introduce our video-rate urban 3D modeling pipeline. In Section 3 we present our approach to loop-closing under widely varying viewpoints. Section 4 discusses additional issues related to calibration of multi-camera systems and solutions to the problem of absolute scale estimation from video. A discussion of open issues and the conclusion are given in Section 5.

2 Real-Time Urban 3D Modeling from Images

This section describes the different steps of our 3D modeling pipeline. The input to our system consists of a video stream combined with a GPS and an Inertial Navigation System (INS), although we are currently exploring how to perform drift free large scale reconstruction without these additional sensors (see Section 3 for more details). The output is a detailed dense textured 3D surface reconstruction of the recorded urban scene. An example is shown in Fig. 1. The example was recorded in Victorville, CA, on the site of the DARPA Urban Challenge. As our goal in this case was to reconstruct 3D models of the facades (as opposed to robot navigation for example), our cameras were oriented to the side of the vehicle. For this example the camera recorded 170,000 video frames at 30Hz with a resolution of 1024 × 768. This means that at 50 km/h an image is recorded approximately every 50 cm along the path of the vehicle. The small baseline simplifies feature tracking for motion

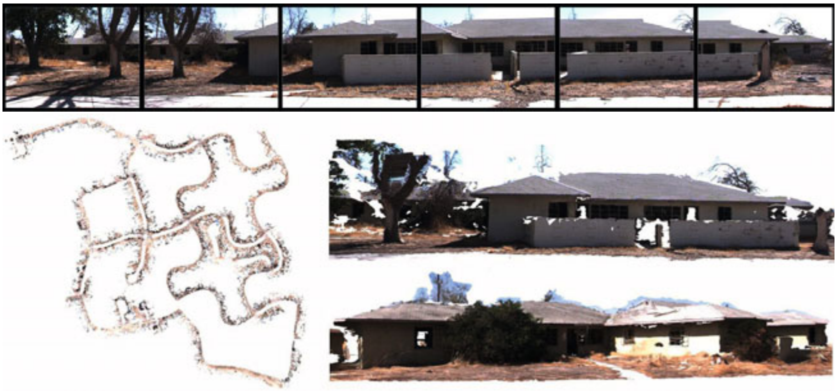


Fig. 1 Urban 3D modeling from video: input video frames (top), top view of 3D model of area of Victorville, CA, (bottom-left) and detailed frontal view of two buildings (bottom-right)

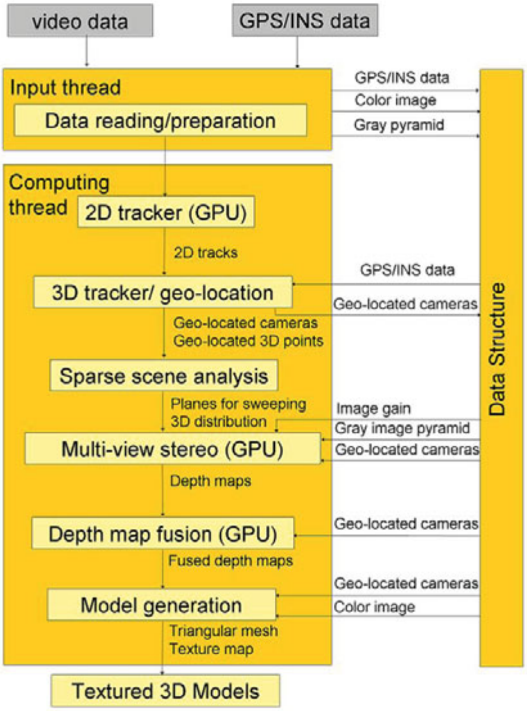


Fig. 2 Processing modules and data flow of our 3D reconstruction pipeline.

estimation. It also ensures very high overlap between the images. The high redundancy facilitates the use of simple multi-view stereo algorithms that can be implemented very efficiently on the GPU. In Fig. 2 an overview of the different stages of our video processing pipeline is given. For efficiency a separate input thread deals with reading the video data from disk. The first computing module extracts features that are then tracked in subsequent frames. To achieve a high performance, this step is performed on the GPU. Next, the feature tracks are used in complement with the INS/GPS system to determine the precise motion of the vehicle and to localize the cameras in world coordinates. At the same time, the 3D location of the tracked features is recovered. This is then used by the next module to obtain a range for the depth of the scene, as well as to extract the dominant orientations of the facades. This information is used to set up the dense multi-view stereo module. This step is followed by a robust depth-map fusion processing, which computes consensus depth-maps by making use of visibility constraints. Both of these steps are efficiently performed on the GPU. Finally, the fused depth-maps are triangulated to obtain a 3D surface mesh. Double representations are removed and a subset of the original images are used as textures for the 3D model. The next sections are discussing the main processing steps in more detail.

2.1 2D Feature Tracking and Motion Estimation

The first step to determine the motion between consecutive video frames is to extract salient image features and track them from frame to frame. For this purpose we use a variant of the well-known Kanade-Lukas-Tomasi (KLT) tracker [41]. To deal with a mix of sunlit and shadow regions, it is important to vary the exposure during recording. In [22] we showed that a consistent global exposure change for the image can efficiently be recovered jointly with the feature displacement and that this performs better than brightness invariant approaches. Our current pipeline uses a very fast KLT implementation, which tracks 1000 features with more than 200Hz in 1024×768 images [59]. This implementation is available in open-source [60].

The next step consists of determining the motion of the camera. As feature tracks can drift and become erroneous, it is important to use robust techniques for motion estimation. For this purpose we use the Random Sampling Consensus (RANSAC) [10]. As this is an essential algorithm for many computer vision systems, many improvements have been proposed. We have for example proposed an approach to deal with quasi-degenerate cases such as scenes where most points lie on a single plane [11] (as these violate some assumptions of the basic algorithm and tend to confuse RANSAC into stopping too early). As hypotheses generated by RANSAC are dependent on a minimal sample, they can often strongly be affected by noise and not allow to directly identify all the inliers. We have recently also proposed a RANSAC algorithm, which properly takes the measurement uncertainties and the transformation uncertainties into account to early identify additional potential inliers and gain up to an order of magnitude in efficiency [38]. In the current system, we use a RANSAC approach, which combines the benefits of several previous approaches to minimize the amount of processing [37]. Knowing the internal calibration of the cameras used we deploy minimal solvers to estimate the relative motion from five points [30] and perform pose estimation from three points [17] as hypothesis generators for RANSAC. The initial triangulation of feature points uses the optimal approach proposed in [18]. Our approach is similar to the visual odometry approach described in [31]. However, this approach only provides satisfying results over relatively short distances. To obtain better results from video only, it is important to perform visual loop-closure as will be discussed in Section 3.

For large scale reconstructions our current system uses a Kalman filter on the 2D feature tracks and the GPS/INS data to estimate geo-located camera poses and 3D feature locations. In Fig. 3 the benefit of fusing the GPS/INS data with the 2D feature tracks in a Kalman filter is illustrated. In addition, as a by-product of the motion estimation, we also obtain the 3D location of the tracked salient scene features. This is very useful as it provides us with a range of interest for the dense stereo matching. In addition, we extract the dominant orthogonal facade orientations from the salient 3D feature points to facilitate the generation of plane hypotheses aligned with building facades as this improves the result of the stereo algorithm [14]. The vertical facade direction is obtained from the INS system or by detecting the corresponding vanishing points. The feature points are projected on a horizontal plane.

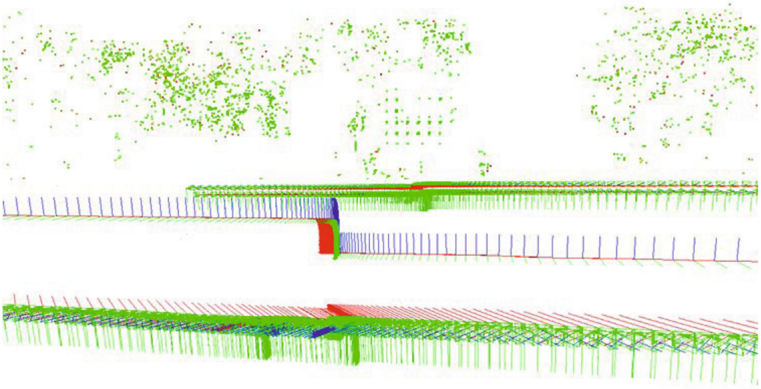


Fig. 3 Illustration of benefit of combining video and GPS/INS for motion estimation. The central track of coordinate frames, which exhibits 10cm vertical drift while the vehicle stopped at a red traffic light, corresponds to the GPS/INS only motion estimation. The coordinate frames in the front and back represent the results from combined GPS/INS and video tracking.

For each possible orientation in the plane the histograms of x and y coordinates are computed and the orientation for which these histograms have the lowest entropy is selected. This process is illustrated in Fig. 4.

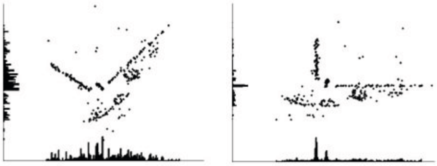


Fig. 4 Top-view of 3D feature locations for two different orientations together with histograms of x and y coordinates. The minimal histogram entropy configuration (shown on the right) is selected.

2.2 *Fast Multi-view Stereo Matching*

To obtain a detailed reconstruction of the surface geometry the sparse set of points reconstructed previously is insufficient. For each video frame and its temporal neighbors we perform multi-view stereo matching to compute the depth for every pixel. Our approach is based on the GPU-friendly multi-view stereo approach proposed in [55, 57]. For a reference video frame its neighbors are identified and a collection of scene planes is hypothesized which samples the depth range sufficiently densely to avoid disparity aliasing. For a particular plane each neighboring image is now projected first on the plane and from there into the reference image where its photo-consistency with respect to the reference image is evaluated. This is done in a single image warping operation during which the exposure change is also compensated. For each pixel the sum of absolute differences is computed. In practice, this is done separately for the reference image and the five previous images and for the reference image and the five succeeding images to provide robustness to occlusions. The minimum of the previous images costs and the succeeding images cost is kept. For each pixel costs are aggregated over a correlation window and the plane with the lowest cost is selected for each pixel independently. From this plane labels it is simple to obtain the depth for each pixel. The same process is repeated for the next frame. A rolling buffer is used to store the eleven frames currently used on the GPU so that each time only one new frame needs to be transferred. More details on this algorithm are provided in [36, 21]. For urban scenes with dominant facade orientations, better results are obtained by using plane hypotheses aligned with the facades. This approach is described in detail in [14]. In this case three sets of planes are hypothesized, two for orthogonal facade directions and one parallel with the ground-plane (not necessarily horizontal in our implementation). For each pixel one can now obtain a depth and a normal. This approach is illustrated in Fig. 5. It can be seen from the figure that the orientations obtained by this approach are largely correct. The depth results are also better as typically the lowest cost is now achieved for the whole aggregation window consistently for the correct depth and orientation. To help resolve the ambiguity in homogenous regions, we add a prior, which is derived from the distribution of the sparse feature points along the different axes as illustrated in Fig. 4. If needed, this stereo algorithm can also be accelerated significantly by only considering the planes with the highest prior likelihoods. One important issue with stereo is that the accuracy degrades quadratically with depth. In many application though the goal is to recover a reconstruction of the scene up to a pre-determined accuracy. In these cases fixed-baseline stereo often has trouble reaching the required depth resolution in the far range of the working volume, this often implies a prohibitive amount of computations are performed in the near range. In [15] we proposed an approach to vary both baseline and resolution used throughout the working volume. The discretization of space for both the standard algorithm and our algorithm are shown in Fig. 6. The amount of computations and accuracy are proportional to the density and shape of the volume elements respectively.

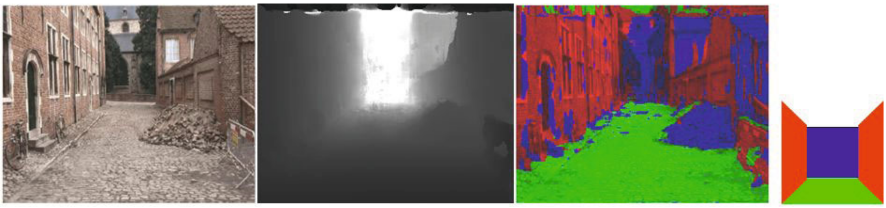


Fig. 5 Multi-view stereo with multiple surface orientation hypotheses: original video frame (left), depth map (middle), orientation map (right).

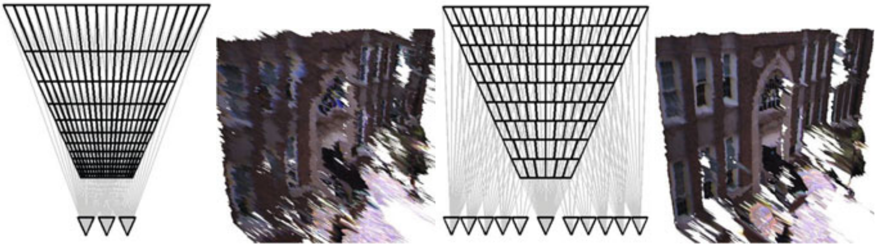


Fig. 6 Discretization of space and results for standard stereo (left) and variable baseline/resolution stereo (right).

Once a depth map has been computed for each video frame, it is important to reconcile overlapping depth maps. This is performed with a robust visibility-based depth map fusion approach. Different types of visibility-based conflicts are shown in Fig. 7. On the left current hypothesis A conflicts with the point A' obtained from view i and we say the free-space of A' is violated. On the right current hypothesis C would be occluded in the reference view by C' obtained from view i. The approach can very efficiently be implemented on the GPU. More details on this approach can be found in [28]. Another very interesting depth-map fusion approach, which minimizes the $TV - L^1$ was recently proposed by Zach et al. [58]. While stereo depth-maps were computed for every frame and had a lot of overlap (every point on the surface is seen in 10-30 frames), fused depth-maps are only computed for a subset of these frames. The goal is to maintain a factor of 2-3 overlap so that regions of the scene that are occluded by foreground objects in one view can be filled in from neighboring fused depth-maps.

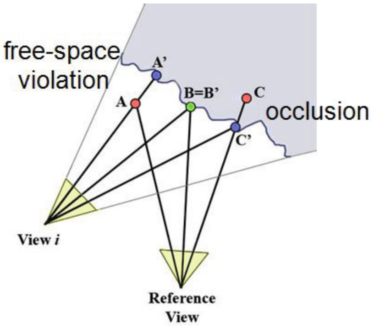


Fig. 7 Visibility-based constraints for depth-map fusion.

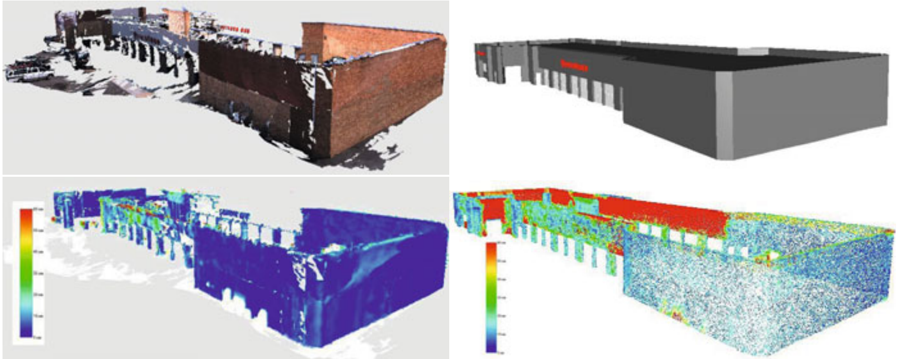


Fig. 8 Firestone evaluation: reconstructed 3D model (top-left), ground-truth 3D model (top-right), color-coded accuracy evaluation (bottom-left) and color-coded completeness evaluation (bottom-right). Blue indicates errors below 10cm while red indicates errors larger than 50cm.

2.3 3D Urban Modeling

Starting from the fused depth maps, a 3D mesh is obtained by overlaying a triangular mesh on the image and projecting it into space according to the depth. An efficient multi-resolution quad-tree algorithm is used to minimize the number of triangles in the mesh [33]. As consecutive fused depth-maps still have a significant amount of overlap, we remove double representations in a post-processing step.

To evaluate the quality of our results, our 3D reconstruction results were compared to a ground-truth model obtained through a professional survey. The results of this comparison are shown in Fig. 8. The accuracy was evaluated by determining the closest point on the ground-truth model for each of the vertices in our model. Completeness is determined similarly by determining if for every vertex of a regular sampled ground-truth model there is a point on our model within some pre-determined distance (30cm in our case). The median and mean error for our approach on this data set are $2-3cm$ and $5-6cm$ respectively, depending on the settings, and the completeness varies from $66\%-73\%$. The relatively low completeness is mostly due to unobserved surfaces and saturated white regions for which no surface was reconstructed.

The complete 3D urban modeling pipeline can process incoming 1024×768 video streams collected at 30 frames per second in real-time on a single PC (with stereo and depth-map fusion being run at half-resolution). This was for example done for a 1.3 million frame data set captured in Chapel Hill. In Fig. 9 the path of the vehicle is shown on the map and a top view of the complete reconstruction is shown. In Fig. 10 a top view of a one reconstruction segment is shown, as well as facade views of two buildings from that segment.

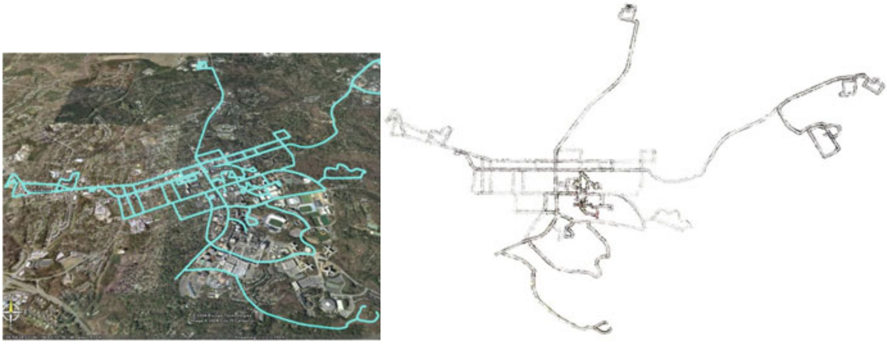


Fig. 9 Map of Chapel Hill, NC, with vehicle path overlaid (left) and top view of recovered reconstruction (right).

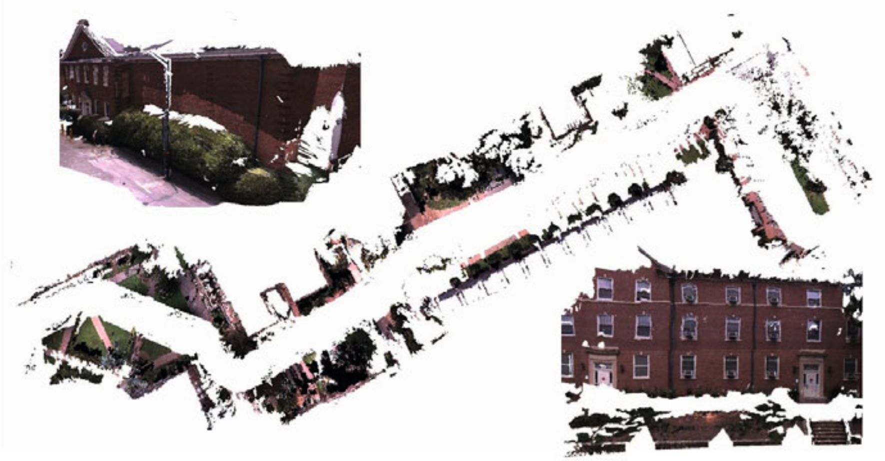


Fig. 10 Top view of segment of the Chapel Hill reconstruction with two views of facades.

3 Visual Loop-Closing

3D models that are created sequentially from input images are always subject to drift. Each small inaccuracy in motion estimation will propagate forward and the absolute positions and motions will be inaccurate. It is therefore necessary to do a global optimization step afterwards to remove the drift. This makes constraints necessary that are capable to remove drift. Such constraints can come from global pose measurements like GPS, but more interesting is to exploit internal consistencies like loops and intersections of the camera path. In this section we will therefore discuss solutions to the challenging task of detecting loops and intersections and using them for global optimization.

3.1 Loop Detection Using Visual Words

Visual loop detection can be phrased as a location recognition problem as described in [39]. The camera path is split up into distinct locations and the visual appearance of each location is described by visual features. A similarity matrix based on the visual features is computed for all image pair combinations. Locations that show a high similarity are considered as loop hypothesis. Each loop hypothesis is then verified by a geometric consistency check. For this the epipolar geometry is computed from feature matches between the two locations. Loop hypotheses that pass the geometric verification threshold are further used for the loop closing process. For an efficient computation of the similarity matrix the visual word based approach originating from [32] is used. The approach quantizes a high-dimensional feature vector (e.g. SIFT) by means of hierarchical k-means clustering, resulting in a so called hierarchical vocabulary tree. The quantization assigns a single integer value, called a visual word (VW), to the originally high-dimensional feature vector. This results in a very compact image representation, where each location is represented by a list of visual words, each only of integer size. The list of visual words from one location forms a document vector, which is a v-dimensional vector where v is the number of possible visual words (a typical choice would be $v = 10^6$). The document vector is a histogram of visual words normalized to 1. To compute the similarity matrix the $L2$ distance between all document vectors is calculated. The document vectors are naturally very sparse and the organization of the database as an inverted file structure makes this very efficient. To describe the visual appearance of a location we propose the use of local features. Techniques for feature matching using scale invariant or affine invariant local features have already achieved a level of maturity and can be considered reliable enough for this application [29]. For each location a set of discriminative and descriptive visual features is extracted. In our approach we use SIFT features [26] or, when a local 3D model can be obtained, VIP features [51] (see next section).

Fig. 11 shows the effect of loop closing on a 400m long trajectory of a vehicle equipped with a camera. The path in blue is from the initial camera poses from structure-from-motion. After filtering the obvious matches from neighboring frames, one loop hypothesis, where the vehicle returns to the start position, can be verified. Loop closing is performed using bundle adjustment and the result is the red trajectory in Fig. 11, which shows that the loop is nicely closed. For this experiment bundle adjustment [49] was used to optimize the camera positions and the 3D features. The detected loops were added as constraints to the bundle adjustment optimization. We provide open source code for sparse bundle adjustment [60].

For the fast computation of SIFT features, we make use of SIFTGPU [53], which can for example extract SIFT features at $\sim 10Hz$ from 1024×768 images. This was one of the key components enabling the real-time localization system described in [19]. It is also very important for efficient large-scale reconstruction from community photo collection [25]. An example of such a reconstruction is shown in Fig.12

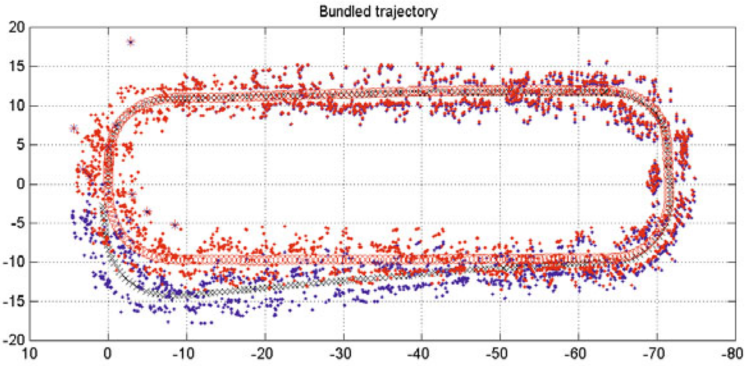


Fig. 11 Camera positions and triangulated features after loop closing (red). Initial estimates are shown in blue and black. The loop is nicely closed.

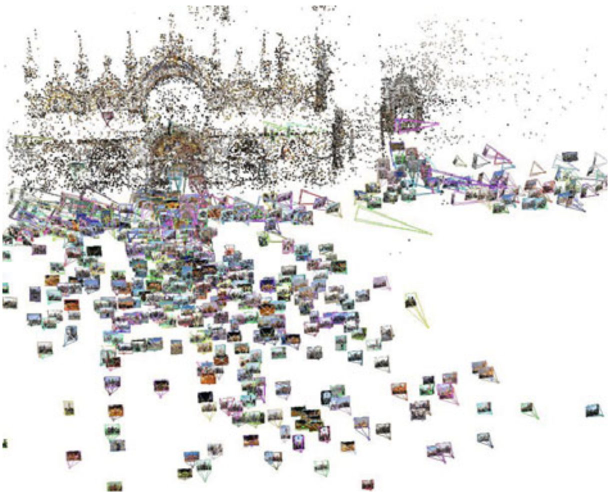


Fig. 12 Reconstruction of part of San Marco square in Venice from a community photo collection.

3.2 *Viewpoint Invariant Patches (VIP) for Loop Closing and 3D Model Registration*

Almost all existing approaches attempt to close loops by matching 2D image features only [9, 39]. However, in most applications scenarios for loop closure and localization images are not recorded in isolation. Indeed, a robot navigating through an environment typically collects videos. This imagery is often sufficient to build a local 3D model at each pass using structure from motion and dense stereo matching techniques as explained earlier in this paper. Therefore, we propose to leverage

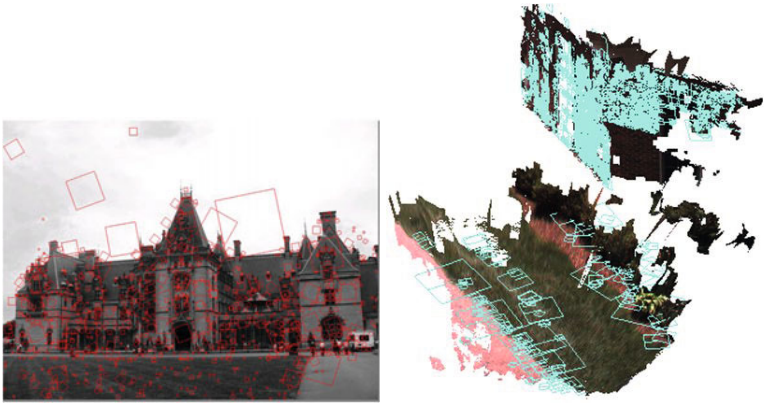


Fig. 13 While SIFT features are extracted from 2D images and are not fully invariant to viewpoint (left), VIP features are extracted on the 3D surface which provide viewpoint invariance and enables single match hypotheses (right).

local 3D scene geometry to achieve viewpoint invariance. For this purpose we have introduced Viewpoint Invariant Patches (VIP) in [51]. VIP's are textured 3D patches extracted from the local textured 3D model in a viewpoint invariant way. Conceptually, our goal is to extract features directly on the surface, in stead of in the images. In urban areas with many planar regions, this can be done efficiently by generating an ortho-texture for each planar surface and then extract features from those texture maps. As the ambiguity for a 3D model extracted from images is a 3D similarity transformation (i.e. scale, orientation and location), for features on a 2D plane embedded in the 3D world, 2D similarity invariance is sufficient to deal with viewpoint changes. Therefore, extracting SIFT features (which are designed to be invariant to 2D similarities) from the ortho-textures provides us full viewpoint invariance (up to view-dependent effects due to non-Lambertian surface reflectance and non-planar details). In addition, a single VIP correspondence is sufficient to uniquely determine a full 3D similarity (scale is obtained from the scale of the feature, orientation from the patch normal and the dominant texture gradient in the patch, location from the keypoint at the center of the patch). The VIP concept is illustrate in Fig. 13

The viewpoint invariance of the VIP features makes them a perfect choice to be used for 3D model registration or loop closing. In the case of 3D model registration we seek a similarity transformation between two overlapping 3D models. For this, VIP features are extracted from each model and subsequently matched. It should be noticed that the relative scale between all matching features extracted from two separate models should be the same. Similarly, the relative rotation between models should also be constant for all patches. Therefore, these can be verified independently. This allows a very effective Hierarchical Efficient Hypothesis Testing (HEHT) scheme. We first verify relative scale by finding the dominant scale and remove all potential feature matches with inconsistent scales. Next, we find the dominant rotation and eliminate outliers and finally we verify inliers for the

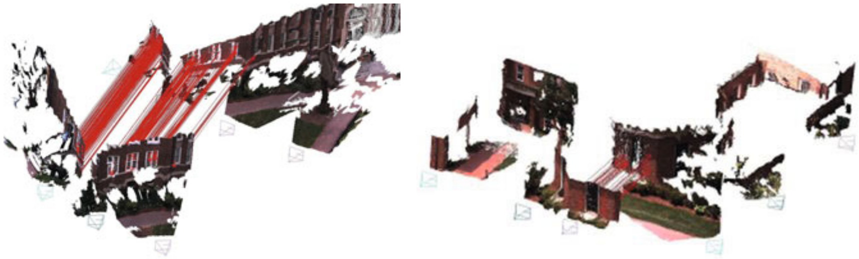


Fig. 14 3D registration of 2 3D models with 45° viewing direction change using VIP features.

dominant translation. It turns out that is approach is particularly effective on urban scenes with many repeating structures and only few good matches supporting the correct hypothesis. The reason is that repeated structures generally support the right scale and –for structures repeating on the same or parallel planes– also the right orientation. For the example shown in Fig. 14 (left), there were 2085 potential VIP matches, 1224 scale inliers, 654 rotation and scale inliers and 214 true inliers. For the example shown on the right of Fig. 14 there were 494 potential VIP matches, 141 scale inliers, 42 rotation and scale inliers and 38 true inliers. For this last example alternative 2D registration approaches failed to return any valid solution. The viewpoint invariance allows the detection of loops with much more significant viewpoint changes. The hierarchical matching enables robustness to repetitive structures and very high levels of outliers ($> 90\%$).

4 Additional Calibration and Motion Estimation Issues

To be able to fuse the models of the different cameras of our capture system into one coherent 3D model we need to determine a common coordinate system for the reconstructions of each individual camera. In the case of known GPS tracking this can be solved by calibrating all cameras internally and registering them into a common coordinate system for which relative scale to the world coordinate system is known, as well as translation and orientation difference to the world coordinate system. In Section 4.1 we provide more detail about the method for internal calibration and external calibration of all cameras into a single common coordinate system. Even with a calibrated (multi-)camera system it is often not straight-forward to determine the scale of the vehicle motion. In Section 4.2 we discuss several approaches to obtain the absolute scale of motion from cameras mounted on a vehicles.

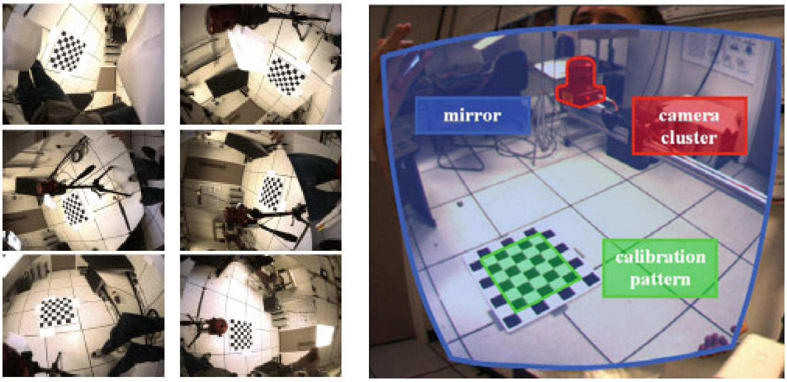


Fig. 15 The calibration setup up for a six camera head is shown on the left. On the right a set of example calibration frames as observed by the cameras are provided.

4.1 Mirror-Based Calibration of Non-overlapping Cameras

For many mapping and robotics applications a wide field of view (FOV) is required of the cameras system. This can be achieved using omnidirectional sensors such as cata-dioptric or fish-eye lenses, or by using camera clusters. In both cases calibration poses specific challenges. In [46, 47] we have proposed a simple self-calibration approach for rotating omnidirectional cameras based on multi-view geometry. When high resolution and high frame rates are desired, camera clusters can be a better choice. To obtain the external calibration for the different cameras of the capture system traditional calibration pattern based methods [50, 61] can not be used due to the fact that the fields of view of the cameras have no overlap. To establish an external calibration of all camera into a single coordinate system we deploy our recently proposed technique for the calibration of non-overlapping cameras using a mirror and a standard calibration pattern [24]. Our technique places one calibration pattern in a fixed position to define the reference coordinate frame for the external calibration of the system. This pattern is typically not seen by any of the cameras or only a very few cameras. Then we use a planar front surface mirror to enable each camera to observe the calibration pattern under multiple mirror positions. Since for the internal calibration of the cameras the mirroring of the cameras does not have any influence we can use any standard technique [50, 61] for pattern based internal calibration directly.

A byproduct of these standard calibration methods is the camera position for each frame captured that shows the pattern reflected by the mirror. Given that the camera frame shows the reflected pattern the reconstructed camera pose for each frame is also the reflected camera pose. This set of reflected camera poses describes a three-dimensional family of camera poses corresponding to the three degrees of freedom for the mirror position, which are the two off-mirror plane rotations and the translation along the mirror normal. Since the mirror positions are unknown from the frames captured for the calibration the camera position in the pattern coordinate system can not be computed through inverse reflection. We showed that in fact the observed three dimensional family of mirrored camera poses determines the real cameras position and orientation uniquely without requiring known mirror positions [24] from as few as two images (five when using only linear equations). The calibration accuracy obtained through this method are comparable to the precision obtained from standard calibration methods like [50, 61]. One could also imagine a self-calibration approach based on the shared motion (up to the relative pose) of the different cameras in the cluster. This idea was explored for rotated cameras [3] and for orthographic cameras [2], but more work is needed for an approach that works well in the general case. In the next section we discuss how to obtain the true world scale even in the absence of GPS measurements.

4.2 Absolute Scale Estimation of Motion

The standard way to get the absolute scale in motion estimation is the use of a stereo setup with a known baseline, e.g. [31, 7]. The fields of views of the two cameras need sufficient overlap and motion estimation is done by triangulating feature points, tracking them, and estimating new poses from them. In [4, 20] we developed algorithms that could compute the absolute scale of motion even without overlap between the two cameras. From independently tracked features in both cameras and with known baseline, full 6DOF motion can be estimated. Another approach [6] makes use of a minimally overlapping stereo pair, which maximizes the field of view of the combined system but leaves some minimal overlap to help compute the absolute scale.

For the case of interest in this paper, where the camera is mounted on a wheeled vehicle, we demonstrated recently that it is even possible to compute the absolute scale of the motion from a single camera only [40] (for the planar motion case). This is possible due to the non-holonomicity of a wheeled vehicle. A wheeled vehicle (e.g. car, bike, etc.) that is constructed to follow the Ackermann steering principle will undergo locally circular motion [16]. In particular, any point on the vertical plane containing the fixed rear axle performs a circular motion, the others will not. A camera, that is not located at the rear axle, will undergo a planar motion, different than that of the circular motion of the car coordinate center. From this camera motion and a measurement of the offset from the rear axle (in meters) the absolute scale of the camera motion can be computed. This makes it possible to upgrade an up-to-scale motion estimate for the camera to an absolutely scaled motion. The absolute scale can be estimated at multiple location throughout the vehicle's path. From these points, the absolute scale can be propagated through structure from motion. Fig. 16 shows results of the absolute scale estimation on a 3 km long camera path. To achieve accurate measurements the absolute scale is only estimated at specific spots where circular vehicle motion is ensured and the turning angle is sufficiently high.

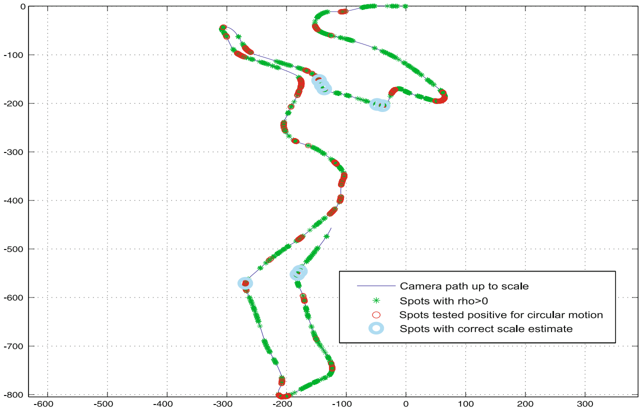


Fig. 16 Absolute scale results on a 3km camera path. The blue circles show spots where the absolute scale was computed. To achieve accurate measurements the absolute scale is only estimated at specific spots where circular vehicle motion is ensured and the turning angle is high.

5 Discussion and Conclusion

In this paper we have discussed a video-rate processing pipeline for large-scale scene reconstruction from ground reconnaissance video. To obtain the high performance reconstructions the system deploys the graphics processing unit throughout the different computation steps of the reconstruction pipeline. The current system relies on GPS for consist modeling of large areas, but we discussed progress on loop-closing and other techniques to enable consistent large scale reconstruction. In particular, the VIP features presented in this paper were shown to be very effective for closing loops in challenging circumstances. Also, depending on the camera configuration, different methods exist to recover and maintain the correct absolute scale of the motion. However, while many of the subproblems now have solutions, many challenges remain to develop an automatic system for wide area reconstruction that does not rely on GPS. Solving this will be important for example to allow the deployment of robots that can operate fully autonomously in large buildings with vision as their main sensor. The techniques discussed here are also important for other applications such as image-based localization. The possibility to determine the location from an image for example is very important to enable advanced location-based services for mobile phones. Although the hardware for our current real-time system is only a single PC, future applications would greatly benefit from the possibility to perform localization and mapping functions on smaller and more energy efficient embedded platforms. We are currently investigating the possibility of performing visual SLAM on very small embedded platforms to support autonomous navigation of micro-aerial vehicles. Other related projects we are currently pursuing are image-based localization for mobile phones and visual SLAM for humanoid robots. Notice that for autonomous robot navigation, besides performing traditional visual SLAM with sparse features, we would also aim to recover a dense surface model and free space from images.

Acknowledgements. We would like to acknowledge the DARPA UrbanScape project and other funding from the US government, the EU sFly project, as well as our collaborators David Nister, Amir Akbarzadeh, Philippos Mordohai, Sudipta Sinha, Paul Merrell, Svetlana Lazebnik, Davide Scaramuzza, Rahul Raguram, Roland Angst, Kevin Köser, Ram Krishan Kumar, Chris Engels, Henrik Stewenius, Brad Talton, Liang Wang, Qingxiong Yang, Ruigang Yang, Greg Welch, Herman Towles, Xiaowei Li.

References

1. Agarwal, S., Snavely, N., Simon, I., Seitz, S., Szeliski, R.: Building Rome in a Day. In: Proc. Int. Conf. on Computer Vision (2009)
2. Angst, R., Pollefeys, M.: Static Multi-Camera Factorization Using Rigid Motion. In: Int. Conf. on Computer Vision (2009)
3. Caspi, Y., Irani, M.: Aligning Non-Overlapping Sequences. Int. J. of Computer Vision 48(1), 39–51 (2002)
4. Clipp, B., Frahm, J.-M., Pollefeys, M., Kim, J.-H., Hartley, R.: Robust 6DOF Motion Estimation for Non-Overlapping Multi-Camera Systems. In: Proc. IEEE Workshop on Applications of Computer Vision (WACV 2008), 8 p. (2008)
5. Clipp, B., Raguram, R., Frahm, J.-M., Welch, G., Pollefeys, M.: A Mobile 3D City Reconstruction System. In: IEEE Virtual Reality Workshop on Virtual Cityscapes (2008)
6. Clipp, B., Zach, C., Frahm, J.-M., Pollefeys, M.: A New Minimal Solution to the Relative Pose of a Calibrated Stereo Camera with Small Field of View Overlap. In: Proc. Int. Conf. on Computer Vision, 8 p. (2009)
7. Clipp, B., Zach, C., Lim, J., Frahm, J.-M., Pollefeys, M.: Adaptive, Real-Time Visual Simultaneous Localization and Mapping. In: Proc. IEEE Workshop on Applications of Computer Vision, WACV 2009 (2009)
8. Cornelis, N., Cornelis, K., Van Gool, L.: Fast Compact City Modeling for Navigation Pre-Visualization. In: Proc. CVPR 2006 IEEE Conf. on Computer Vision and Pattern Recognition, vol. 2, pp. 1339–1344 (2006)
9. Cummins, M., Newman, P.: FAB-MAP: Probabilistic Localisation and Mapping in the Space of Appearance. Int. J. of Robotics Research (June 2008)
10. Fischler, M., Bolles, R.: Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Comm. of the ACM 24, 381–395 (1981)
11. Frahm, J.-M., Pollefeys, M.: RANSAC for (Quasi-)Degenereate data (QDEGSAC). In: Proc. CVPR 2006 IEEE Conf. on Computer Vision and Pattern Recognition (2006)
12. Fraundorfer, F., Frahm, J.-M., Pollefeys, M.: Visual Word based Location Recognition in 3D models using Distance Augmented Weighting. In: Proc. 3DPVT 2008 Int. Symp. on 3D Data Processing, Visualization and Transmission (2008)
13. Früh, C., Zakhor, A.: An Automated Method for Large-Scale, Ground-Based City Model Acquisition. Int. J. of Computer Vision 60(1), 5–24 (2004)
14. Gallup, D., Frahm, J.-M., Mordohai, P., Yang, Q., Pollefeys, M.: Real-Time Plane-sweeping Stereo with Multiple Sweeping Directions. In: Proc. CVPR 2007, IEEE Conf. on Computer Vision and Pattern Recognition (2007)

15. Gallup, D., Frahm, J.-M., Mordohai, P., Pollefeys, M.: Variable Baseline/Resolution Stereo. In: Proc. CVPR 2008, IEEE Conf. on Computer Vision and Pattern Recognition (2008)
16. Gillespie, T.: Fundamentals of Vehicle Dynamics. SAE, Inc., Warrendale (1992)
17. Haralick, R., Lee, C.-N., Ottenberg, K., Nolle, M.: Review and Analysis of Solutions of the Three Point Perspective Pose Estimation Problem. Int. J. of Computer Vision 13(3), 331–356 (1994)
18. Hartley, R., Sturm, P.: Triangulation. Computer Vision and Image Understanding (CVIU) 68(2), 146–157 (1997)
19. Irschara, A., Zach, C., Frahm, J.-M., Bischof, H.: 3D Scene Summarization for Efficient View Registration. In: Proc. CVPR 2009, IEEE Conf. on Computer Vision and Pattern Recognition (2009)
20. Kim, J.-H., Hartley, R., Frahm, J.-M., Pollefeys, M.: Visual Odometry for Non-Overlapping Views Using Second-Order Cone Programming. In: Yagi, Y., Kang, S.B., Kweon, I.S., Zha, H. (eds.) ACCV 2007, Part II. LNCS, vol. 4844, pp. 353–362. Springer, Heidelberg (2007)
21. Kim, S.J., Gallup, D., Frahm, J.-M., Akbarzadeh, A., Yang, Q., Yang, R., Nister, D., Pollefeys, M.: Gain Adaptive Real-Time Stereo Streaming. In: Proc. Int. Conf. on Computer Vision Systems (2007)
22. Kim, S.-J., Frahm, J.-M., Pollefeys, M.: Joint Feature Tracking and Radiometric Calibration from Auto-Exposure Video. In: Proc. ICCV 2007, Int. Conf. on Computer Vision (2007)
23. Kim, S.J., Pollefeys, M.: Robust Radiometric Calibration and Vignetting Correction. IEEE Trans. on Pattern Analysis and Machine Intelligence 30(4), 562–576 (2008)
24. Kumar, R.K., Ilie, A., Frahm, J.-M., Pollefeys, M.: Simple calibration of non-overlapping cameras with a mirror. In: Proc. CVPR 2008, IEEE Int. Conf. on Computer Vision and Pattern Recognition (2008)
25. Li, X., Wu, C., Zach, C., Lazebnik, S., Frahm, J.-M.: Modeling and Recognition of Landmark Image Collections Using Iconic Scene Graphs. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 427–440. Springer, Heidelberg (2008)
26. Lowe, D.: Distinctive image features from scale-invariant keypoints. Int. J. of Computer Vision 60(2), 91–110 (2004)
27. Merrell, P., Mordohai, P., Frahm, J.M., Pollefeys, M.: Evaluation of Large Scale Scene Reconstruction. In: Proc. Workshop on Virtual Representations and Modeling of Large-scale environments, VRML 2007 (2007)
28. Merrell, P., Akbarzadeh, A., Wang, L., Mordohai, P., Frahm, J.-M., Yang, R., Nister, D., Pollefeys, M.: Fast Visibility-Based Fusion of Depth Maps. In: Proc. ICCV 2007, Int. Conf. on Computer Vision (2007)
29. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., Van Gool, L.: A comparison of affine region detectors. Int. J. of Computer Vision 65(1/2), 43–72 (2005)
30. Nister, D.: An efficient solution to the five-point relative pose problem. IEEE Trans. on Pattern Analysis and Machine Intelligence 26(6), 756–777 (2004)
31. Nister, D., Naroditsky, O., Bergen, J.: Visual odometry for ground vehicle applications. J. of Field Robotics 23(1) (2006)
32. Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. In: Proc. CVPR 2006, IEEE Conf. on Computer Vision and Pattern Recognition, vol. 2, pp. 2161–2168 (2006)
33. Pajarola, R.: Overview of quadtree-based terrain triangulation and visualization. Tech. Report UCI-ICS-02-01, Information & Computer Science, University of California Irvine (2002)

34. Pollefeys, M., Koch, R., Van Gool, L.: Self-Calibration and Metric Reconstruction in spite of Varying and Unknown Internal Camera Parameters. Int. J. of Computer Vision 32(1), 7–25 (1999)
35. Pollefeys, M., Van Gool, L., Vergauwen, M., Verbiest, F., Cornelis, K., Tops, J., Koch, R.: Visual modeling with a hand-held camera. Int. J. of Computer Vision 59(3), 207–232 (2004)
36. Pollefeys, M., Nister, D., Frahm, J.-M., Akbarzadeh, A., Mordohai, P., Clipp, B., Engels, C., Gallup, D., Kim, S.-J., Merrell, P., Salmi, C., Sinha, S., Talton, B., Wang, L., Yang, Q., Stewenius, H., Yang, R., Welch, G., Towles, H.: Detailed Real-Time Urban 3D Reconstruction From Video. Int. J. of Computer Vision 78(2), 143–167 (2008)
37. Raguram, R., Frahm, J.-M., Pollefeys, M.: A Comparative Analysis of RANSAC Techniques Leading to Adaptive Real-Time Random Sample Consensus. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 500–513. Springer, Heidelberg (2008)
38. Raguram, R., Frahm, J.-M., Pollefeys, M.: Exploiting Uncertainty in Random Sample Consensus. In: Proc. ICCV 2009, Int. Conf. on Computer Vision (2009)
39. Scaramuzza, D., Fraundorfer, F., Pollefeys, M., Siegwart, R.: Closing the Loop in Appearance-Guided Structure-from-Motion for Omnidirectional Cameras. In: Proc. Eight Workshop on Omnidirectional Vision, ECCV (2008)
40. Scaramuzza, D., Fraundorfer, F., Pollefeys, M., Siegwart, R.: Absolute Scale in Structure from Motion from a Single Vehicle Mounted Camera by Exploiting Nonholonomic Constraints. In: Proc. ICCV 2009, Int. Conf. on Computer Vision (2009)
41. Shi, J., Tomasi, C.: Good Features to Track. In: CVPR 1994, IEEE Conf. on Computer Vision and Pattern Recognition, pp. 593–600 (1994)
42. Sinha, S., Mordohai, P., Pollefeys, M.: Multi-View Stereo via Graph Cuts on the Dual of an Adaptive Tetrahedral Mesh. In: Proc. ICCV 2007, Int. Conf. on Computer Vision (2007)
43. Sinha, S., Frahm, J.-M., Pollefeys, M., Genc, Y.: Feature Tracking and Matching in Video Using Programmable Graphics Hardware. Machine Vision and Application (November 2009) (online)
44. Sinha, S., Steedly, D., Szeliski, R., Agrawala, M., Pollefeys, M.: Interactive 3D Architectural Modeling from Unordered Photo Collections. ACM Trans. on Graphics (SIGGRAPH ASIA 2008) 27(5), 159, 1–10 (2008)
45. Snavely, N., Seitz, S., Szeliski, R.: Photo Tourism: Exploring image collections in 3D. In: Proc. of SIGGRAPH 2006 (2006)
46. Thirthala, S., Pollefeys, M.: The Radial Trifocal Tensor: A Tool for Calibrating Radial Disortion of Wide-Angle Cameras. In: Proc. CVPR 2005, IEEE Conf. on Computer Vision and Pattern Recognition, vol. 1, pp. 321–328 (2005)
47. Thirthala, S., Pollefeys, M.: Multi-view geometry of 1D radial cameras and its application to omnidirectional camera calibration. In: Proc. ICCV 2005, Int. Conf. on Computer Vision, vol. 2, pp. 1539–1546 (2005)
48. Tomasi, C., Kanade, T.: Shape and motion from image streams under orthography—a factorization method. Int. J. of Computer Vision 9(2), 137–154 (1992)
49. Triggs, B., McLauchlan, P., Hartley, R., Fitzgibbon, A.: Bundle Adjustment – A Modern Synthesis. In: Triggs, B., Zisserman, A., Szeliski, R. (eds.) ICCV-WS 1999. LNCS, vol. 1883, pp. 298–372. Springer, Heidelberg (2000)
50. Tsai, R.Y.: A versatile camera calibration technique for high accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses. IEEE J. for Robotics and Automation 3, 323–344 (1987)

51. Wu, C., Clipp, B., Li, X., Frahm, J.-M., Pollefeys, M.: 3D Model Matching with Viewpoint Invariant Patches (VIPs). In: Proc. CVPR 2008, IEEE Conf. on Computer Vision and Pattern Recognition (2008)
52. Wu, C., Fraundorfer, F., Frahm, J.-M., Pollefeys, M.: 3D Model Search and Pose Estimation from Single Images using VIP Features. In: Proc. S3D Workshop, CVPR 2008 (2008)
53. Wu, C.: Open Source SIFTGPU, http://www.cs.unc.edu/~ccwu/siftgpu/
54. Xiao, J., Fang, T., Zhao, P., Lhuillier, M., Quan, L.: Image-Based Street-Side City Modeling. ACM Trans. on Graphics, SIGGRAPH ASIA (2009)
55. Yang, R., Pollefeys, M.: Multi-Resolution Real-Time Stereo on Commodity Graphics Hardware. In: Proc. CVPR 2003, IEEE Conf. on Computer Vision and Pattern Recognition, pp. 211–218 (2003)
56. Yang, R., Pollefeys, M., Welch, G.: Dealing with Textureless Regions and Specular Highlight: A Progressive Space Carving Scheme Using a Novel Photo-consistency Measure. In: Proc. ICCV 2003, Int. Conf. on Computer Vision, pp. 576–584 (2003)
57. Yang, R., Pollefeys, M.: A Versatile Stereo Implementation on Commodity Graphics Hardware. J. of Real-Time Imaging 11(1), 7–18 (2005)
58. Zach, C., Pock, T., Bischof, H.: A globally optimal algorithm for robust TV-L1 range image integration. In: Proc. ICCV 2007, IEEE Int. Conf. on Computer Vision (2007)
59. Zach, C., Gallup, D., Frahm, J.-M.: Fast Gain-Adaptive KLT Tracking on the GPU. In: CVPR Workshop on Visual Computer Vision on GPU's, CVGPU (2008)
60. Zach, C.: Open Source Code for GPU-based KLT Feature Tracker and Sparse Bundle Adjustment, http://www.cs.unc.edu/~cmzach/opensource.html
61. Zhang, Z.: A flexible new technique for camera calibration. IEEE Trans. on Pattern Analysis and Machine Intelligence, 1330–1334 (2000)

Place-Dependent People Tracking

Matthias Luber, Gian Diego Tipaldi, and Kai O. Arras

Abstract. People typically move and act under the constraints of an environment, making human behavior strongly place-dependent. Motion patterns, the places and the rates at which people appear, disappear, walk or stand are not random but engendered by the environment. In this paper, we learn a non-homogeneous spatial Poisson process to spatially ground human activity events for the purpose of people tracking. We show how this representation can be used to compute refined probability distributions over hypotheses in a multi-hypothesis tracker and to make better, place-dependent predictions of human motion. In experiments with data from a laser range finder, we demonstrate how both extensions lead to more accurate tracking behavior in terms of data association errors and number of track losses. The system runs in real-time on a typical desktop computer.

1 Introduction

As robots enter more domains in which they interact and cooperate closely with humans, people tracking is becoming a key technology for areas such as human-robot interaction, human activity understanding or intelligent cars. In contrast to air- and waterborne targets, people typically move and act under environmental constraints. These constraints vary over space and enable and limit possible motions and action patterns (people cannot walk through walls, cooking is constraint to places with a stove), making human behavior strongly place-dependent.

In this paper we learn and represent human spatial behavior for the purpose of people tracking. By learning a spatial model that represents activity

Matthias Luber · Gian Diego Tipaldi · Kai O. Arras
Social Robotics Lab, University of Freiburg, Department of Computer Science, Georges-Koehler-Allee 106, 79110 Freiburg, Germany
e-mail: {luber,tipaldi,arras}@informatik.uni-freiburg.de

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 557–572.
springerlink.com

events in a global reference frame and on large time scales, the robot acquires place-dependent priors on human behavior. As we will demonstrate, such priors can be used to better hypothesize about the state of the world (that is, the state of people in the world), and to make place-dependent predictions of human motion that better reflect how people are using space. Concretely, we propose a non-homogeneous spatial Poisson process to represent the spatially varying distribution over relevant human activity events. This representation, called *spatial affordance map*, holds space-dependent Poisson rates for the occurrence of track events such as creation, confirmation or false alarm. The map is then incorporated into a multi-hypothesis tracking framework using data from a laser range finder.

In most related work on laser-based people tracking [1, 2, 3, 4, 5, 6, 7], a person is represented as a single state that encodes torso position and velocities. People are extracted from range data as single blobs or found by merging nearby point clusters that correspond to legs. People tracking has also been addressed as a leg tracking problem [8, 9, 10] where people are represented by the states of two legs, either in a single augmented state [9] or as a high-level track to which two low-level leg tracks are associated [8, 10].

Different tracking and data association approaches have been used for laser-based people tracking. The nearest neighbor filter and variations thereof are typically employed in earlier works [1, 2, 3]. A sample-based joint probabilistic data association filter (JPDAF) has been presented in Schulz *et al.* [4] and adopted by Topp *et al.* [5]. The Multi-hypothesis tracking (MHT) approach according to Reid [11] and Cox *et al.* [12] has been used in [8, 7, 10]. What makes the MHT an attractive choice is that it belongs to the most general data association techniques. The method generates joint compatible assignments, integrates them over time, and is able to deal with track creation, confirmation, occlusion, and deletion events in a probabilistically consistent way. Other multi-target data association techniques such as the global nearest neighbor filter, the track splitting filter or the JPDAF are suboptimal in nature as they simplify the problem in one or the other way [13, 14]. For these reasons, the MHT has become a widely accepted tool in the target tracking community [14].

For people tracking, however, the MHT approach relies on statistical assumptions that are overly simplified and do not account for place-dependent target behavior. It assumes new tracks and false alarms being uniformly distributed in the sensor field of view with fixed Poisson rates. While this might be acceptable in settings for which the approach has been originally developed (using, e.g., radar or underwater sonar), it does not account for the non-random usage of an environment by people. Human subjects appear, disappear, walk or stand at specific locations. False alarms are also more likely to arise in areas with cluttered backgrounds rather than in open spaces. In this paper, we extend prior work by incorporating learned distributions over track interpretation events in order to support data association and show how a non-homogeneous spatial Poisson process can be used to seamlessly extend the MHT approach for this purpose.

For motion prediction of people, most researchers employ the Brownian motion model and the constant velocity motion model. The former makes no assumptions about the target dynamics, the latter assumes linear target motion. Better motion models for people tracking have been proposed by Bruce and Gordon [15] and Liao *et al.* [16].

In [15], the robot learns goal locations in an environment from people trajectories obtained by a laser-based tracker. Goals are found as end points of clustered trajectories. Human motion is then predicted along paths that a planner generates from the location of people being tracked to the goal locations. The performance of the tracker was improved in comparison to a Brownian motion model. Liao *et al.* [16] extract a Voronoi graph from a map of the environment and represent the state of people being on edges of that graph. This allows them to predict motion of people along the edges that follow the topological shape of the environment.

With maneuvering targets, a single model can be insufficient to represent the target's motion. Multiple model based approaches in which different models run in parallel and describe different aspects of the target behavior are a widely accepted technique to deal with maneuvering targets, in particular the Interacting Multiple Model (IMM) algorithm [17]. Different target motion models are also studied by Kwok and Fox [18]. The approach is based on a Rao-Blackwellized particle filter to model the potential interactions between a target and its environment. The authors define a discrete set of different target motion models from which the filter draws samples. Then, conditioned on the model, the target is tracked using Kalman filters.

Regarding motion models, our approach extends prior work in two aspects: learning and place-dependency. Opposed to [16, 18] and IMM related methods, we do not rely on predefined motion models but apply learning for this task in order to acquire place-dependent models. In [16], the positions of people is projected onto a Voronoi graph which is a topologically correct but metrically poor model for human motion. While sufficient for the purpose of their work, there is no insight why people should move on a Voronoi set, particularly in open spaces whose topology is not well defined. Our approach, by contrast, tracks the actual position of people and predicts their motion according to metric, place-dependent models. Opposed to [15] where motion prediction is done along paths that a planner plans to a set of goal locations, our learning approach predicts motion along the trajectories that people are actually following.

The paper is structured as follows: the next section gives a brief overview of the people tracker that will later be extended. Section 3 introduces the theory of the spatial affordance map and expressions for learning its parameters. Section 4 describes how the spatial affordance map can be used to compute refined probability distributions over hypotheses, while section 5 contains the theory for the place-dependent motion model. Section 6 presents the experimental results followed by the conclusions in section 7.

2 Multi-Hypothesis Tracking of People

For people tracking, we pursue a Multi-Hypothesis Tracking (MHT) approach described in Arras *et al.* [10] based on the original MHT by Reid [11] and Cox and Hingorani [12]. As we will use the tracker to learn the spatial affordance map described hereafter, we give a short outline. Sections 4 and 5, where the approach will be extended, contains the technical details.

The MHT algorithm, in summary, hypothesizes about the state of the world by considering all statistically feasible assignments between measurements and tracks and all possible interpretations of measurements as false alarms or new track and tracks as matched, occluded or obsolete. A hypothesis Ω_i^t is one possible set of assignments and interpretations at time t.

For learning the spatial affordance map, the hypothesis with maximal probability Ω_{best}^t at time t is chosen to produce the track events that subsequently serve as observations for the map. In case of a sensor mounted on a mobile platform, we assume the existence of a metric map of the environment and the ability of the robot to self-localize. Observations are then transformed from local robot-centric coordinates into the world reference frame of the map.

3 Spatial Affordance Map

We pose the problem of learning a spatial model of human behavior as a parameter estimation problem of a non-homogeneous spatial Poisson process. The resulting model, called spatial affordance map, is a global long-term representation of human activity events in the environment. The name lends itself to the concept of affordances as we consider the possible sets of human actions and motions as a result from environmental constraints. An affordance is a resource or support that an object (the environment) offers an (human) agent for action. This section describes the theory and how learning in the spatial affordance map is implemented.

A Poisson distribution is a discrete distribution to compute the probability of a certain number of events given an expected average number of events over time or space. The parameter of the distribution is the positive real number λ, the rate at which events occur per time or volume units. As we are interesting in modeling events that occur randomly in time, the Poisson distribution is a natural choice.

Based on the assumption that events in time occur independently of one another, a *Poisson process* can deal with distributions of time intervals between events. Concretely, let $N(t)$ be a discrete random variable to represent the number of events occurring up to time t with rate λ. Then we have that $N(t)$ follows a Poisson distribution with parameter λt

$$P(N(t)=k)=\frac{e^{-\lambda t}(\lambda t)^k}{k!} \qquad k=0,1,\dots \tag{1}$$

In general, the rate parameter may change over time. In this case, the generalized rate function is given as $\lambda(t)$ and the expected number of events between time a and b is

$$\lambda_{a,b}=\int_a^b \lambda(t)\,dt. \tag{2}$$

A homogeneous Poisson process is a special case of a non-homogeneous process with constant rate $\lambda(t)=\lambda$.

The *spatial* Poisson process introduces a spatial dependency on the rate function given as $\lambda(\mathbf{x},t)$ with $\mathbf{x}\in X$ where X is a vector space such as $\mathbb{R}^2$ or $\mathbb{R}^3$. For any subset $S\subset X$ of finite extent (e.g. a spatial region), the number of events occurring inside this region can be modeled as a Poisson process with associated rate function $\lambda_S(t)$ such that

$$\lambda_S(t)=\int_S \lambda(\mathbf{x},t)\,d\mathbf{x}. \tag{3}$$

In the case that this generalized rate function is a separable function of time and space, we have:

$$\lambda(\mathbf{x},t)=f(\mathbf{x})\lambda(t) \tag{4}$$

for some function $f(\mathbf{x})$ for which we can demand

$$\int_X f(\mathbf{x})\,d\mathbf{x}=1 \tag{5}$$

without loss of generality. This particular decomposition allows us to decouple the occurrence of events between time and space. Given Eq. 5, $\lambda(t)$ defines the occurrence rate of events, while $f(\mathbf{x})$ can be interpreted as a probability distribution on where the event occurs in space.

Learning the spatio-temporal distribution of events in an environment is equivalent to learn the generalized rate function $\lambda(\mathbf{x},t)$. However, learning the full continuous function is a highly expensive process. For this reason, we approximate the non-homogeneous spatial Poisson process with a piecewise homogeneous one. The approximation is performed by discretizing the environment into a bidimensional grid, where each cell represents a local homogeneous Poisson process with a fixed rate over time,

$$P_{ij}(k)=\frac{e^{-\lambda_{ij}}(\lambda_{ij})^k}{k!} \qquad k=0,1,\dots \tag{6}$$

where λ_{ij} is assumed to be constant over time. Finally, the spatial affordance map is the generalized rate function $\lambda(\mathbf{x},t)$ using a grid approximation,

$$\lambda(\mathbf{x},t) \simeq \sum_{(i,j)\in X} \lambda_{ij}\mathbf{1}_{ij}(\mathbf{x}) \tag{7}$$

with $\mathbf{1}_{ij}(\mathbf{x})$ being the indicator function defined as $\mathbf{1}_{ij}(\mathbf{x}) = 1$ if $\mathbf{x} \in \text{cell}_{ij}$ and $\mathbf{1}_{ij}(\mathbf{x}) = 0$ if $\mathbf{x} \notin \text{cell}_{ij}$. The type of approximation is not imperative and goes without loss of generality. Other space tessellation techniques such as graphs, quadtrees or arbitrary regions of homogeneous Poisson rates can equally be used. Subdivision of space into regions of fixed Poisson rates has the property that the preferable decomposition in Eq. 4 holds.

Each type of human activity event can be used to learn its own probability distribution in the map. We can therefore think of the map as a representation with multiple layers, one for every type of event. For the purpose of this paper, the map has three layers, one for new tracks, for matched tracks and for false alarms. The first layer represents the distribution and rates of people appearing in the environment. The second layer can be considered a space usage probability and contains a walkable area map of the environment. The false alarm layer represents the place-dependent reliability of the detector.

3.1 Learning

In this section we show how to learn the parameter of a single cell in our grid from a sequence $K_{1..n}$ of n observations $k_i \in \{0,1\}$. We use Bayesian inference for parameter learning, since the Bayesian approach can provide information on cells via a prior distribution. We model the parameter λ using a Gamma distribution, as it is the conjugate prior of the Poisson distribution. Let λ be distributed according to the Gamma density, $\lambda \sim \text{Gamma}(\alpha,\beta)$, parametrized by the two parameters α and β,

$$\text{Gamma}(\lambda;\alpha,\beta) = \frac{\beta^{\alpha}}{\Gamma(\alpha)}\lambda^{\alpha-1}e^{-\beta\lambda} \qquad \text{for } \lambda > 0. \tag{8}$$

Then, learning the rate parameter λ consists in estimating the parameters of a Gamma distribution. At discrete time index i, the posterior probability of λ_i according to Bayes' rule is computed as $P(\lambda_i|K_{1..i}) \sim P(k_i|\lambda_{i-1})P(\lambda_{i-1})$ with $P(\lambda_{i-1}) = \text{Gamma}(\alpha_{i-1},\beta_{i-1})$ being the prior and $P(k_i|\lambda_{i-1}) = P(k_i)$ from Eq. 6 the likelihood. Then by substitution, it can be shown that the update rules for the parameters are $\alpha_i = \alpha_{i-1} + k_i$ and $\beta_i = \beta_{i-1} + 1$. The posterior mean of the rate parameter in a single cell is finally obtained as the expected value of the Gamma,

$$\widehat{\lambda}_{\text{Bayesian}} = \mathbb{E}[\lambda] = \frac{\alpha}{\beta} = \frac{\#\text{positive events} + 1}{\#\text{observations} + 1}\ . \tag{9}$$

For $i = 0$ the quasi uniform Gamma prior for $\alpha = 1$, $\beta = 1$ is taken. The advantages of the Bayesian estimator are that it provides a variance estimate which is a measure of confidence of the mean and that it allows to properly initialize never observed cells.

Given the learned rates we can estimate the space distribution of the various events. This distribution is obtained from the rate function of our spatial affordance map $\lambda(\mathbf{x},t)$. While this estimation is hard in the general setting of a non-homogeneous spatial Poisson process, it becomes easy to compute if the separability property of Eq. 4 holds[1]. In this case, the pdf, $f(\mathbf{x})$, is obtained by

$$f(\mathbf{x}) = \frac{\lambda(\mathbf{x},t)}{\lambda(t)} \tag{10}$$

where $\lambda(\mathbf{x},t)$ is the spatial affordance map. The nominator, $\lambda(t)$, can be obtained from the map by substituting the expression for $f(\mathbf{x})$ into the constraint defined in Eq. 5. Hence,

$$\lambda(t) = \int_X \lambda(\mathbf{x},t)\, d\mathbf{x}. \tag{11}$$

In our grid, those quantities are computed as

$$f(\mathbf{x}) = \frac{\sum_{(i,j)\in X} \lambda_{ij} \mathbf{1}_{ij}(\mathbf{x})}{\sum_{(i,j)\in X} \lambda_{ij}}. \tag{12}$$

In case of several layers in the map, each layer contains the distribution $f(\mathbf{x})$ of the respective type of events. Note that learning in the spatial affordance map is simply realized by counting in a grid. This makes life-long learning particularly straightforward as new information can be added at any time by one or multiple robots.

Figure 1 shows two layers of the spatial affordance map of our laboratory, learned during a first experiment. The picture on the left shows the space usage distribution of the environment. The modes in this distribution correspond to often used places and have the meaning of goal locations in that room (three desks and a sofa). On the right, the distribution of new tracks is depicted whose peaks denote locations where people appear (doors). The reason for the small peaks at other locations than the doors is that when subjects use an object (sit on a chair, lie on the sofa), they cause a track loss. When they reenter space, they are detected again as new tracks.

[1] Note that for a non-separable rate function, the Poisson process can model places whose importance changes over time.

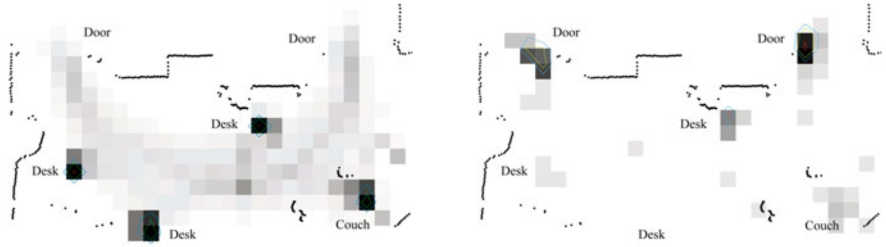


Fig. 1 Spatial affordance map of the laboratory in experiment 1. The probability distribution of matched track events is shown on the left, the distribution of new track events is shown on the right. The marked locations in each distribution (extracted with a peak finder and visualized by contours of equal probability) have different meanings. While on the left they correspond to places that are often used by people (three desks and a sofa), the maxima of the new track distribution (right) denote locations where people appear (two doors, the couch and a desk).

4 MHT Data Association with Spatial Target Priors

The MHT assumes a Poisson distribution for the occurrences of new tracks and false alarms over time and an uniform probability of these events over space within the sensor field of view V. While this is a valid assumption for a radar aimed upwards into the sky, it does not account for the place-dependent character of human spatial behavior: people typically appear, disappear, walk and stand at specific locations that correspond, for instance, to doors, elevators, entrances, or convex corners.

It is exactly this information that the spatial affordance map holds. We can therefore seamlessly extend the MHT approach with the learned Poisson rates for the arrival events of people and learned location statistics for new tracks and false alarms.

At time t, each possible set of assignments and interpretations forms a hypothesis Ω_i^t. Let $Z(t) = \{z_i(t)\}_{i=1}^{m_t}$ be the set of m_t measurements which in our case is the set of detected people in the laser data. For detection, we use a learned classifier based on a collection of boosted features [19]. Let further $\psi_i(t)$ denote a set of assignments which associates predicted tracks to measurements in $Z(t)$ and let Z^t be the set of all measurements up to time t. Starting from a hypothesis of the previous time step, called a parent hypothesis $\Omega_{p(i)}^{t-1}$, and a new set $Z(t)$, there are many possible assignment sets $\psi(t)$, each giving birth to a child hypothesis that branches off the parent. This makes up an exponentially growing hypothesis tree. For a real-time implementation, the growing tree needs to be pruned. To guide the pruning, each hypothesis receives a probability, recursively calculated as the product of a normalizer η, a measurement likelihood, an assignment set probability and the parent hypothesis probability [11],

$$p(\Omega_l^t \mid Z^t) = \eta \cdot p(Z(t) \mid \psi_i(t), \Omega_{p(i)}^{t-1}) \\ p(\psi_i(t) \mid \Omega_{p(i)}^{t-1}, Z^{t-1}) \cdot p(\Omega_{p(i)}^{t-1} \mid Z^{t-1}). \tag{13}$$

While the last term is known from the previous iteration, the two expressions that will be affected by our extension are the measurement likelihood and the assignment set probability.

For the measurement likelihood, we assume that a measurement $z_i(t)$ associated to a track $\mathbf{x}_j$ has a Gaussian pdf centered on the measurement prediction $\hat{z}_j(t)$ with innovation covariance matrix $S_{ij}(t)$, $\mathcal{N}(z_i(t)) := \mathcal{N}(z_i(t); \hat{z}_j(t), S_{ij}(t))$. The regular MHT now assumes that the pdf of a measurement belonging to a new track or false alarm is uniform in V, the sensor field of view, with probability V^{-1}. Thus

$$p(Z(t) \mid \psi_i(t), \Omega_{p(i)}^{t-1}) = V^{-(N_F+N_N)} \cdot \prod_{i=1}^{m_t} \mathcal{N}(z_i(t))^{\delta_i} \tag{14}$$

with N_F and N_N being the number of measurements labeled as false alarms and new tracks respectively. δ_i is an indicator variable being 1 if measurement i has been associated to a track, and 0 otherwise.

Given the spatial affordance map, the term changes as follows. The probability of new tracks V^{-1} can now be replaced by

$$p_N(\mathbf{x}) = \frac{\lambda_N(\mathbf{x},t)}{\lambda_N(t)} = \frac{\lambda_N(\mathbf{x},t)}{\int_V \lambda_N(\mathbf{x},t)\, d\mathbf{x}} \tag{15}$$

where $\lambda_N(\mathbf{x},t)$ is the learned Poisson rate of new tracks in the map and $\mathbf{x}$ the position of measurement $z_i'(t)$ transformed into global coordinates. The same derivation applies for false alarms. Given our grid, Eq. 15 becomes

$$p_N(\mathbf{x}) = \frac{\lambda_N(z_i'(t),t)}{\sum_{(i,j)\in V} \lambda_{ij,N}}. \tag{16}$$

The probability of false alarms $p_F(\mathbf{x})$ is calculated in the same way using the learned Poisson rate of false alarms $\lambda_F(\mathbf{x},t)$ in the map.

The original expression for the assignment set probability can be shown to be [10]

$$p(\psi_i(t) \mid \Omega_{p(i)}^{t-1}, Z^{t-1}) = \eta' \cdot p_M^{N_M} \cdot p_O^{N_O} \cdot p_D^{N_D} \\ \lambda_N^{N_N} \cdot \lambda_F^{N_F} \cdot V^{(N_F+N_N)} \tag{17}$$

where N_M, N_O, and N_D are the number of matched, occluded and deleted tracks, respectively. The parameters p_M, p_O, and p_D denote the probability of matching, occlusion and deletion that are subject to $p_M + p_O + p_D = 1$. The regular MHT now assumes that the number of new tracks N_N and false alarms N_F both follow a fixed rate Poisson distribution with expected number of occurrences $\lambda_N V$ and $\lambda_F V$ in the observation volume V.

Given the spatial affordance map, they can be replaced by rates from the learned spatial Poisson process with rate functions $\lambda_N(t)$ and $\lambda_F(t)$ respectively.

Substituting the modified terms back into Eq. 13 makes, like in the original approach, that many terms cancel out leading to an easy-to-implement expression for a hypothesis probability

$$p(\Omega_l^t \mid Z^t) = \eta'' \cdot p_M^{N_M} \cdot p_O^{N_O} \cdot p_D^{N_D} \cdot \prod_{i=1}^{m_t} [\mathcal{N}(z_i(t))^{\delta_i} \qquad (18)$$
$$\lambda_N(z_i'(t),t)^{\kappa_i} \cdot \lambda_F(z_i'(t),t)^{\phi_i}] \cdot p(\Omega_{p(i)}^{t-1} \mid Z^{t-1})$$

with δ_i and κ_i being indicator variables whether a track is matched to a measurement or new, respectively, and ϕ_i indicating if a measurement is declared to be a false alarm.

The insight of this extension of the MHT is that we replace fixed parameters by learned distributions. This kind of domain knowledge helps the tracker to better interpret measurements and tracks, leading to refined probability distributions over hypotheses at the same run-time costs.

5 Place-Dependent Motion Models

Tracking algorithms rely on the predict-update cycle, where a motion model predicts the future target position which is then validated by an observation in the update phase. Without validation, caused, for instance, by the target being hidden during an occlusion event, the state evolves blindly following only the prediction model. Good motion models are especially important for people tracking as people typically undergo lengthy occlusion events during interaction with each other or with the environment.

As motion of people is hard to predict, having a precise model is difficult. People can abruptly stop, turn back, left or right, make a step sideways or accelerate suddenly. However, motion of people is not random but follows place-dependent patterns. They, for instance, turn around convex corners, avoid static obstacles, stop in front of doors and do not go through walls. Clearly, the Brownian, the constant velocity and even higher-order motion models are unable to capture the complexity of these movements.

For this reason, we extend the constant velocity motion assumption with a place-dependent model derived from the learned space usage distribution in the spatial affordance map. Let $\mathbf{x}_t = (x_t \ y_t \ \dot{x}_t \ \dot{y}_t)^T$ be the state of a track at time t and Σ_t its covariance estimate. The motion model $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ is then defined as

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; F\,\mathbf{x}_{t-1}, F\,\Sigma_{t-1}\,F^T + Q) \tag{19}$$

with F being the state transition matrix. The entries in Q represent the acceleration capability of a human. We extend this model by considering how the distribution of the state at a generic time t is influenced by the previous state and the map. This distribution is approximated by the following factorization

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}, m) \simeq p(\mathbf{x}_t|\mathbf{x}_{t-1}) \cdot p(\mathbf{x}_t|m) \tag{20}$$

where m is the spatial affordance map and $p(\mathbf{x}_t|m) = f(\mathbf{x}_t)$ denotes the space usage probability of the portion of the environment occupied by $\mathbf{x}_t$, as defined by Eq. 12.

A closed form estimation of this distribution does not exist since the map contains a general density, poorly described by a parametric distribution. We therefore follow a sampling approach and use a particle filter to address this estimation problem. The particle filter is a sequential Monte Carlo technique based on the importance sampling principle. In practice, it represents a target distribution in form of a set of weighted samples

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}, m) \simeq \sum_i w^{(i)} \delta_{\mathbf{x}_t^{(i)}}(\mathbf{x}_t). \tag{21}$$

where $\delta_{\mathbf{x}_t^{(i)}}(\mathbf{x}_t)$ is the impulse function centered in $\mathbf{x}_t^{(i)}$. Sampling directly from that distribution is not possible so the algorithm first computes samples from a so called proposal distribution, π. The algorithm, then, computes the importance weight related to the i-th sample that takes into account the mismatch among the target distribution τ and the proposal distribution $w = \frac{\tau}{\pi}$. The weights are then normalized such that $\sum w = 1$.

In our case, we take the constant velocity model to derive the proposal π. The importance weights are then represented by the space usage probability

$$w^{(i)} = \frac{p(\mathbf{x}_t|\mathbf{x}_{t-1}, m)}{p(\mathbf{x}_t^{(i)}|\mathbf{x}_{t-1})} = p(\mathbf{x}_t^{(i)}|m). \tag{22}$$

The new motion model has now the form of a weighted sample set. Since we are using Kalman filters for tracking, the first two moments of this distribution is estimated by

$$\hat{\mu} = \sum_i w^{(i)} \mathbf{x}_t^{(i)} \tag{23}$$

$$\hat{\Sigma} = \sum_i w^{(i)} (\hat{\mu} - \mathbf{x}_t^{(i)})(\hat{\mu} - \mathbf{x}_t^{(i)})^T. \tag{24}$$

The target is then predicted using $\hat{\mu}$ as the state prediction with associated covariance $\hat{\Sigma}$. Obviously, the last step is not needed when using particle filters for tracking.

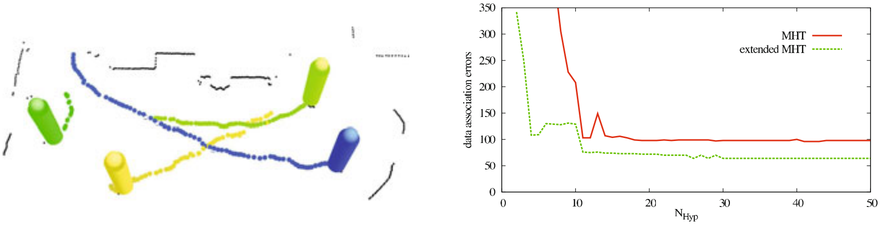


Fig. 2 Four of 135 example tracks from experiment 1 (left). The total number of data association errors as a function of N_{Hyp}, the maximum number of hypotheses for the MHT algorithm to maintain (right). The solid red line shows the regular MHT tracker, the dotted green line the extended approach. The graph shows that by replacing the fixed Poisson rates by the learned, place-dependent ones, the tracker requires fewer hypothesis to reach the same level of accuracy and, given more hypotheses, makes around 30% fewer data association errors.

An example situation that exemplifies how this motion model works is shown in Figure 3. A person that takes a left turn in a hallway is tracked over a lengthy occlusion event. The constant velocity motion model (dashed ellipse) predicts the target into a wall and outside the walkable area of the environment. The place-dependent model (solid ellipse) is able to follow the left turn with a state covariance shaped like the hallway. In other words, the model predicts the target "around the corner". The tracker with the constant velocity motion loses track as the reappearing person is outside the validation gate (shown as 95% ellipses).

6 Experiments

For the experiments we collected two data sets, one in a laboratory (experiment 1, Figure 2) and one in an office building (experiment 2, Figure 3). As sensors we used a fixed Sick laser scanner with an angular resolution of 0.5 degree.

The spatial affordance maps were trained based on the tracker described in [10], the grid cells were chosen to be 30 cm in size. The parameters of the tracker have been learned from a training data set with 95 tracks over 28242 frames. All data associations including occlusions have been hand-labeled. This led to a matching probability $p_M = 0.59$, an occlusion probability $p_O = 0.40$, a deletion probability $p_D = 0.01$, a fixed Poisson rate for new tracks $\lambda_N = 0.0102$ and a fixed Poisson rate for false alarms as $\lambda_F = 0.00008$. The rates have been estimated using the Bayesian approach in Eq. 9.

The implementation of our system runs in real-time on a 2.8 GHz quad-core CPU. The cycle time of a typical setting with $N_{Hyp} = 50$, 500 samples for the particle filter, and up to eight parallel tracks is around 12 Hz when sensor data are immediately available.

6.1 MHT Data Association with Spatial Target Priors

The original MHT is compared to the approach using the spatial affordance map on the data set from the laboratory over 38994 frames and with a total number of 135 people entering and leaving the sensor field of view. The ground truth has been determined by manual inspection. For the comparison we count the total number of data association errors that are created by the best hypotheses of the two tracking methods. The data association error is defined to be the number of identifier switches over the life cycle of a ground truth track. We use a pruning strategy which limits the maximum number of hypotheses at every step to N_{Hyp} (the multi-parent variant of the pruning algorithm proposed by Murty [20]). In order to show the evolution of the error as a function of N_{Hyp}, the computational effort, N_{Hyp} is varied from 1 to 50.

The result shows a significant improvement of the extended MHT over the regular approach (see Figure 2). The explanation is given by an example. As can be seen in Figure 1 right, few new track events have been observed in the center of the room. If at such a place a track occlusion occurs (e.g. from another person), hypotheses that interpret this as an obsolete track followed by a new track receive a much smaller probability through the spatial affordance map than hypotheses that assume this to be an occlusion.

A data set with 15 people was collected to investigate whether the model is overfitted and generalizes poorly for unusual behavior of people. Subjects entered the sensor field of view through entry points that have never been used (in between the couch and the desk at the bottom in Fig 1) or appeared in the center of the room (by jumping off from tables). Manual inspection of the resulting trees (using the graphviz-lib for visualization) revealed that all 15 people are tracked correctly. The difference to the approach with fixed Poisson rates is that, after track creation, the best hypothesis is not the true one during the first few (less than five) iterations. However, the incorrect hypotheses that successively postulate the subjects being a false alarm become very unlikely, causing the algorithm to backtrack to the true hypothesis.

6.2 Place-Dependent Motion Model

In the second experiment, the constant velocity motion model is compared to our place-dependent motion model. A training set over 7443 frames with 50 person tracks in a office-like environment was recorded to learn the spatial affordance map (see Figure 3). A test set with 6971 frames and 28 people tracks was used to compare the two models. The data set was labeled by hand to determine both, the ground truth positions of people and the true data associations. In order to make the task more difficult, we defined areas in which target observations are ignored as if the person had been occluded by an object or another person. These areas were placed at hallway corners and U-turns where people typically maneuver. As the occlusion is simulated,

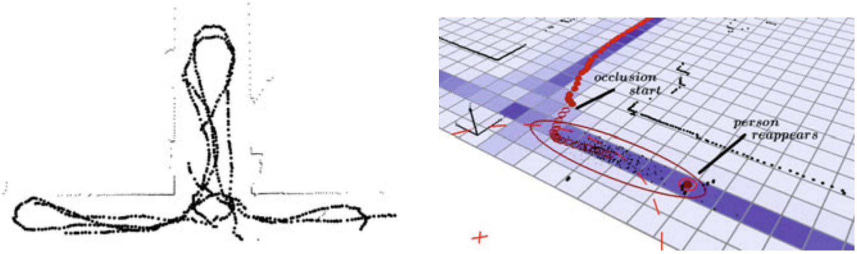


Fig. 3 Six (of 50) example tracks from experiment 2. Trajectory of a person in experiment 2 taking a left turn during an occlusion event. Predictions from a constant velocity motion model (dashed ellipse) and the new model (solid ellipse) are shown. The background grid (in blue) shows the learned space usage distribution of the spatial affordance map. The small black dots are the weighted samples of the place-dependent motion model. The model is able to predict the target "around the corner" yielding much better motion predictions in this type of situations.

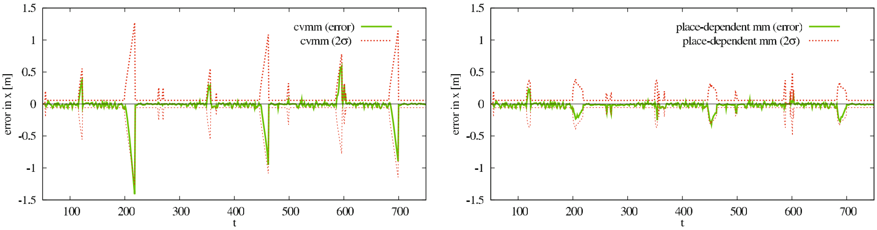


Fig. 4 Comparison between constant velocity motion model (cvmm, left) and place-dependent motion model (right). Peaks correspond to occluded target maneuvers (turns around corners and U-turns). Fig. 3 shows the left turn of a person at step 217 of this experiment. While both approaches are largely consistent from a estimation point of view, the place-dependent model results in an overall smaller estimation error and smaller uncertainties. For 28 manually inspected tracks, the constant velocity motion model lost a track 15 times while the new model had only one track loss.

the ground truth position of the targets is still available. As a measure of accuracy, the posterior position estimates of both approaches to the ground truth is calculated. The resulting estimation error in x is shown in Figure 4 (the error in y is similar).

The diagram shows much smaller estimation errors and 2σ bounds for the place-dependent motion model during target maneuvers. An important result is that the predicted covariances do not grow boundless during the occlusion events (peaks in the error plots). As illustrated in Figure 3, the shape of the covariance predictions follows the walkable area map at the very place of the target. Smaller covariances lead to lower levels of data association ambiguity, and thus, to decreased computational costs and more accurate probability distribution over pruned hypothesis trees.

For 28 manually inspected tracks, the constant velocity motion model lost a track 15 times while the new model had only a single track loss. As a naive countermeasure, one could enlarge the entries of the process noise covariance Q to make the constant velocity motion model avoid such losses, but this is clearly the wrong way to go as it brings along an even higher level of data association ambiguity.

7 Conclusions

In this paper we presented a people tracking approach that accounts for the place-dependency of human behavior. We posed the problem of learning a spatial model of human behavior as a parameter estimation problem of a non-homogeneous spatial Poisson process. The model, called spatial affordance map, is learned using Bayesian inference from observations of track creation, matching and false alarm events, gained by introspection of a laser-based multi-hypothesis people tracker.

The map enabled us to relax and overcome the simplistic fixed Poisson rate assumption for new tracks and false alarms in the MHT approach. Using a learned spatio-temporal Poisson rate function, the system was able to compute refined probability distributions over hypotheses, resulting in a clearly more accurate tracking behavior in terms of data association errors at the same costs.

The map further allowed us to derive a new, place-dependent model to predict target motion. The model showed superior performance in predicting maneuvering targets especially during lengthy occlusion events when compared to a constant velocity motion model.

Future work will consider model-driven hypothesis generation at places where the spatial affordance map is highly multi-modal for target predictions during very long occlusion events.

Acknowledgment

This work has partly been supported by the German Research Foundation (DFG) under contract number SFB/TR-8.

References

1. Kluge, B., Köhler, C., Prassler, E.: Fast and robust tracking of multiple moving objects with a laser range finder. In: Proc. of the Int. Conf. on Robotics & Automation, ICRA (2001)
2. Fod, A., Howard, A., Mataríc, M.: Laser-based people tracking. In: Proc. of the Int. Conf. on Robotics & Automation, ICRA (2002)
3. Kleinhagenbrock, M., Lang, S., Fritsch, J., Lömker, F., Fink, G., Sagerer, G.: Person tracking with a mobile robot based on multi-modal anchoring. In: IEEE International Workshop on Robot and Human Interactive Communication (RO-MAN), Berlin, Germany (2002)

4. Schulz, D., Burgard, W., Fox, D., Cremers, A.: People tracking with a mobile robot using sample-based joint probabilistic data association filters. International Journal of Robotics Research (IJRR) 22(2), 99–116 (2003)
5. Topp, E., Christensen, H.: Tracking for following and passing persons. In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), Alberta, Canada (2005)
6. Cui, J., Zha, H., Zhao, H., Shibasaki, R.: Tracking multiple people using laser and vision. In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), Alberta, Canada (2005)
7. Mucientes, M., Burgard, W.: Multiple hypothesis tracking of clusters of people. In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), Beijing, China (2006)
8. Taylor, G., Kleeman, L.: A multiple hypothesis walking person tracker with switched dynamic model. In: Proc. of the Australasian Conf. on Robotics and Automation, Canberra, Australia (2004)
9. Cui, J., Zha, H., Zhao, H., Shibasaki, R.: Laser-based interacting people tracking using multi-level observations. In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), Beijing, China (2006)
10. Arras, K.O., Grzonka, S., Luber, M., Burgard, W.: Efficient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities. In: Proc. of the Int. Conf. on Robotics & Automation, ICRA (2008)
11. Reid, D.B.: An algorithm for tracking multiple targets. IEEE Transactions on Automatic Control 24(6) (1979)
12. Cox, I.J., Hingorani, S.L.: An efficient implementation of reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. IEEE Trans. Pattern Anal. Mach. Intell (PAMI) 18(2), 138–150 (1996)
13. Bar-Shalom, Y., Li, X.-R.: Multitarget-Multisensor Tracking: Principles and Techniques. YBS Publishing, Storrs (1995)
14. Blackman, S.S.: Multiple hypothesis tracking for multiple target tracking. IEEE Aerospace and Electronic Systems Magazine 19(1), 5–18 (2004)
15. Bruce, A., Gordon, G.: Better motion prediction for people-tracking. In: Proc. of the Int. Conf. on Robotics & Automation (ICRA), Barcelona, Spain (2004)
16. Liao, L., Fox, D., Hightower, J., Kautz, H., Schulz, D.: Voronoi tracking: Location estimation using sparse and noisy sensor data. In: Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS (2003)
17. Mazor, E., Averbuch, A., Bar-Shalom, Y., Dayan, J.: Interacting multiple model methods in target tracking: a survey. IEEE Transactions on Aerospace and Electronic Systems 34(1), 103–123 (1998)
18. Kwok, C., Fox, D.: Map-based multiple model tracking of a moving object. In: Nardi, D., Riedmiller, M., Sammut, C., Santos-Victor, J. (eds.) RoboCup 2004. LNCS (LNAI), vol. 3276, pp. 18–33. Springer, Heidelberg (2005)
19. Arras, K.O., Mozos, Ó.M., Burgard, W.: Using boosted features for the detection of people in 2d range data. In: Proc. of the Int. Conf. on Robotics & Automation (ICRA), Rome, Italy (2007)
20. Murty, K.: An algorithm for ranking all the assignments in order of increasing cost. Operations Research 16 (1968)

Combining Laser-Scanning Data and Images for Target Tracking and Scene Modeling

Hongbin Zha, Huijing Zhao, Jinshi Cui, Xuan Song, and Xianghua Ying

Abstract. Working environments of modern robots have changed to unstructured, dynamic and outdoor scenes. There emerged several new challenges along with these changes, mainly in perception of both static and dynamic objects of the scenes. To tackle these new challenges, this research focused on study of advanced perception systems that can simultaneously model static scenes and track dynamic objects. Our research has three features. Multi-view and multi-type sensors, together with machine learning based algorithms, are utilized to obtain robust and reliable mapping/tracking results. In addition, a car-based mobile perception system is developed for exploring large sites. Finally, to improve robustness of the multi-view and mobile perception system, some new camera calibration methods are proposed. This paper presents an overview of our recent study on above mentioned ideas and technologies. Specifically we will focus on multi-sensor based multiple target tracking, simultaneous 3D mapping and target tracking in a mobile platform, and camera calibration.

1 Introduction

During last decade, with rapid developments in robotics technologies and their applications, working environments of robots have changed much. They change from indoor to outdoor, from structured to unstructured, from static to dynamically changing environments.

To adapt to these changes, and also to achieve more complicated tasks in new working environments, it is necessary to develop advanced robotics perception systems that are more reliable, robust and capable in scene recognition and understanding. Specifically, the challenges emerging with changing environments include three aspects. Firstly, most of the environment is unknown, and contains both static and

Hongbin Zha · Huijing Zhao · Jinshi Cui · Xuan Song · Xianghua Ying
Key Laboratory of Machine Perception (MoE), Peking University
e-mail: {zha,zhaohj,cjs,songxuan,xhying}@cis.pku.edu.cn

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 573–587.
springerlink.com

dynamic objects, with complex relationships and structures among them. Secondly, there are various targets needing to be recognized, and their categories are more complicated. The targets may include static objects for instance architectures, trees and roads, as well as dynamic objects, like pedestrians, bicycles and cars. Finally, in a dynamic and multi-object environment, when multiple dynamic objects interact with each other, the perception task becomes more challenging in object recognition, localization and tracking due to inter-occlusion.

To address above problems, we need to find some new research directions. First of all, it would be necessary to make full use of 3D information of various objects. At the same time, we also have to utilize prior information of specific outdoor environments, gather data from multiple sensors and then fuse them efficiently. Moreover, to cover a wide and complicated area, it would be indispensable to use a distributed sensor network or an actively controlled mobile platform.

According to above mentioned directions, we explored research in multi-target tracking and 3D environment mapping with laser scanners and video cameras. It has three main features: 1) Multi-sensors and machine learning based algorithms are utilized to obtain robust and reliable mapping and tracking. 2) A car-based mobile perception system is developed for 3D mapping of static environment scenes, as well as for recognizing and tracking of dynamic targets. 3) To improve robustness of the multi-view and mobile perception systems, some new camera calibration methods are proposed.

This paper presents an overview of our recent study on above technologies and systems. Specifically we will focus on multi-sensor based multiple target tracking, simultaneous 3D mapping and target tracking in a mobile platform, and camera calibration.

2 Multi-target Tracking in Dynamic Scene

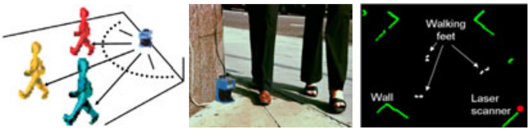
Multi-target tracking plays a vital role in various applications, such as surveillance, sports video analysis, human motion analysis and many others. Multi-target tracking is much easy when the targets are distinctive and do not interact with each other. It can be solved by employing multiple independent trackers. However, for those targets that are similar in appearance, obtaining their correct trajectories becomes significantly challenging when they are in close proximity or partial occlusions. Previous approaches using joint trackers searching in joint state space requires high time consumptions. Moreover, maintaining the correct tracking seems almost impossible when the well-known ”merge/split” condition occurs (some targets occlude others completely, but they split after several frames). Hence, the goals of our research are: 1) to design a multi-sensor system that will help obtain a better tracking performance with lower time consumption than those obtained from independent or joint trackers when the interactions occur; 2) to make a new attempt to solve the ”merge/split” problem in multi-target tracking.

Our recent research on these goals covers three aspects: 1) To solve partial occlusion and interaction in a joint space, a detection-driven MCMC based particle filter framework is proposed. In this approach, MCMC sampling is utilized to increase search speed with detection maps providing searching directions. 2) To address the problem of severe occlusion and interaction, we use learning and classification loops for data association. 3) To reduce time consumption and to improve tracking performance, we fuse laser and vision data. Compared to traditional vision-based tracking systems, the laser range scanner can provide directly 3D depth information of targets, which is absent in visual images. In a laser-based tracking system (as shown in Fig.1), the targets are represented by several points, and hence the tracking become much easy with good performance in both accuracy and time-cost.

2.1 Detection-Driven MCMC Based Particle Filter

In the visual tracking area, to keep track of multiple moving objects, one generally has to estimate the joint probability distribution of the state of all objects. This, however, is intractable in practice even for a small number of objects since the size of the state space grows exponentially in the number of objects. An optional solution of the problem is to use the detection based data association framework, which is originated from radar tracking techniques. Most of existing laser based tracking systems used this framework. In these systems, a clustering/segmentation based detection algorithm provides locations of potential targets. Then, the measurements are associated with previously estimated target trajectories in a data association step. Above detection driven tracking schemes greatly rely on the performance of the detection algorithms. Only observation at locations with high detection responses are considered as potential measurements. This will incur that false alarms and non-detections significantly influence performance of the tracker.

In our recent research, we construct our novel observation with two types of measurements, including a foreground image frame given by background subtraction and a detection map on the single frame. For inference, we proposed a detection incorporated joint particle filter, considering the both types of measurements. First, data association for the targets to detected measurements is incorporated to the state



(a) Measurement (b) Experimental site (c) Sample data

Fig. 1 A typical laser based tracking system

proposal, to form a mixture proposal that combines information from the dynamic model and the detected measurements. Then, we utilize a MCMC sampling step to obtain an efficient multi-target filter.

We applied this idea to laser scan frames [1]. Four laser scanners are used for scanning on the height of 16cm from horizontal ground. Fig.2 is a screen copy of trajectories of tracked persons using our MCMC particle filter, where green points represent laser points of background (doors, tables, walls, etc.); white points represent the laser points of moving feet. Colour lines are trajectories. Red circles are position of people at current time. The numbers denotes the trajectory indices. The standard particle filter can track an individual person very well using 100 particle samples, if he/she is quite far away from other persons. However, if two persons walk closely, it is very common that one person's trajectory "hijacks" another person's, since there is not a joint likelihood to handle the interaction situation. Our MCMC particle filter benefits from the feature detection and the mixture transition proposal. It tracks 28 persons simultaneously and nearly in real-time, and gives a robust tracking result, even using only 100 particle samples.

We also applied this idea to vision based tracking systems. In [2],we proposed a Probabilistic Detection-based Particle Filter (PD-PF) for multi-target tracking. In our method, we incorporate possible probabilistic detections and information from dynamic models to construct a mixed proposal for the particle filter, which models interactions and occlusions among targets quite effectively.

2.2 On-Line Learning for Data Association

To address long-time occlusion and severe interactions, we proposed an online learning based approach for data association. The core idea of our research is illustrated in Fig.3 and Fig.4. When two targets do not interact with each other (see Fig.3), tracking becomes very easy and multiple independent trackers are employed. Due to the reliability of these tracking results, they are used as positive or negative samples to train a classifier for each target. When the two targets are in close

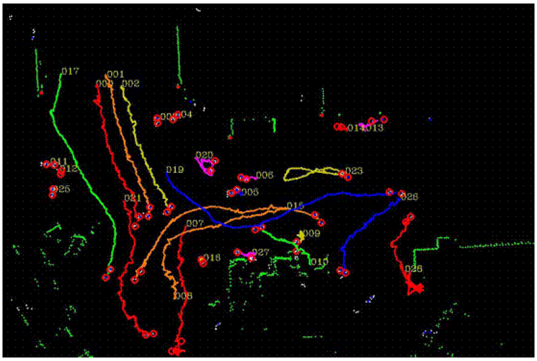


Fig. 2 Laser based tracking results using detection-driven MCMC based PF.

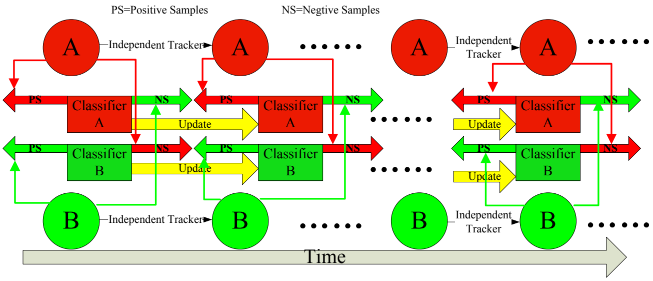


Fig. 3 Tracking for learning.

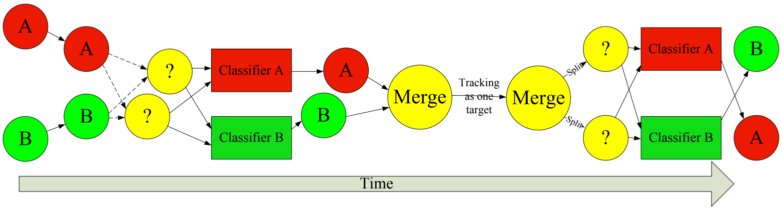
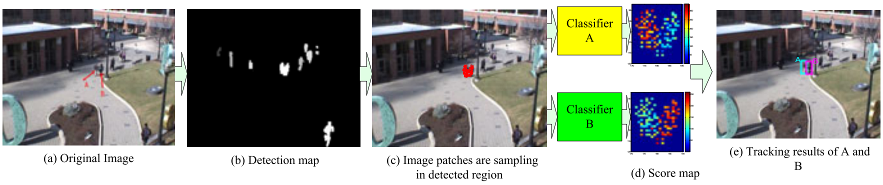


Fig. 4 Learning for tracking.

proximity (see Fig.4), the learned classifiers are used to assist in tracking. Specifically, when the two targets merge, we assign a new state space and track this ”merging target” as one target. When they split, their classifiers are used again to specify a correct identification.

We applied this idea respectively to vision based systems [3] and laser based systems [4]. In a vision based tracking system, when the targets do not interact with each other, we utilize independent trackers to perform the tracking. Once we obtaine the tracking results of each target, a set of random image patches are spatially sampled within the image region of each target. We utilize these random image patches as samples for the online supervised learning. In this case, each target is represented by a ”bag of patches” model.When the targets are in close proximity or present partial occlusions, a set of random image patches are sampled within the interacting region of the detected map, and the feature vectors of these image patches are input to the classifiers of interacting targets respectively. The outputs of these classifiers are scores, which are used to weight the observation model in the particle filter. The overview of the process is shown in Fig.5.

Sometimes, several targets occlude another target completely. Maintaining the correct tracking of targets seems quite impossible. Once it occurs, we initialize the state of the ”merging targets” and track it as one target. If we detect that this ”merging target” splits and becomes an interacting condition or a non-correlated condition, we utilized the classifiers of these targets to identify them (as shown in Fig.6). Hence, we can link the trajectories of these targets without difficulty. Some experimental results were shown in Fig.7.



(a) Original Image

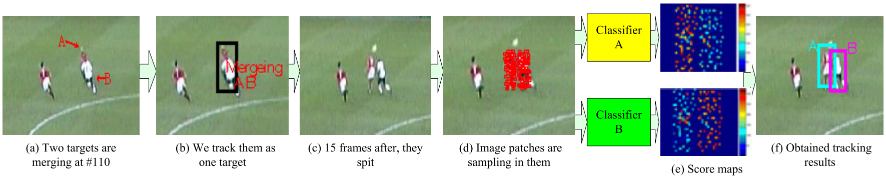
(b) Detection map

(c) Image patches are sampling in detected region

(d) Score map

(e) Tracking results of A and B

Fig. 5 Correlated targets tracking.



(a) Two targets are merging at #110

(b) We track them as one target

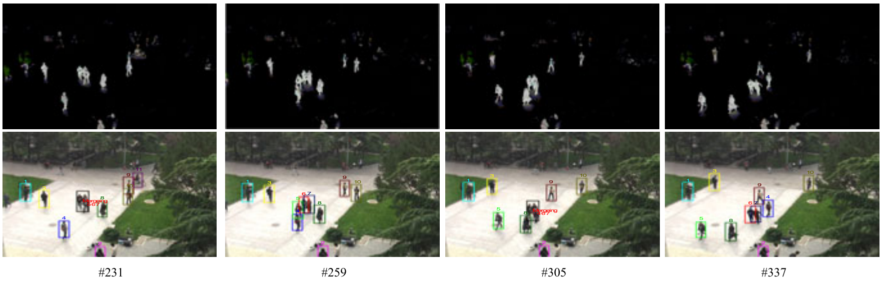
(c) 15 frames after, they spit

(d) Image patches are sampling in them

(e) Score maps

(f) Obtained tracking results

Fig. 6 Merge/Split condition.


#231

#259

#305

#337

Fig. 7 Tracking results of surveillance video.

We have applied the same idea to laser based tracking systems [4]. Although it performs better than previous approaches in the situations of interactions and occlusions, due to the missing appearance information, it is very hard to obtain a set of features that uniquely distinguish one object from another.

2.3 Fusion of Laser and Vision

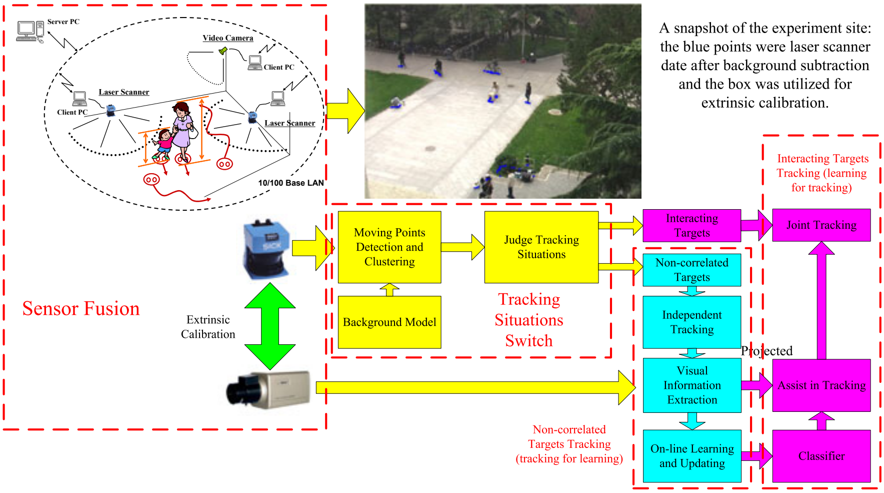
The work on fusion of laser and visual data is motivated from pursuing a reliable and real-time multi-target tracking system, which is difficult to achieve via only laser or only visual data. We have tried several strategies for the fusion of two modes. In [5], a Kalman Filter based approach is utilized for decision-level fusion of two independent tracking results respectively from a laser-based sub-system and a vision-based subsystem. Although the time-consuming of this approach is rather high, it can provide reasonable tracking results. In [6], the fusion is processed in the detection stage at first to improve detection rates and decrease false-alarm rates. Then, an observation model that combines visual and laser features is utilized for filtering. The fusion in detection-level saves a lot time since no more window scanning is necessary in image frames. However, it still request much time for gathering measurements from both modes in filtering process. Moreover, it is difficult to decide the confidence coefficients of laser data and vision data when complex interaction situations occur.

In our most recent research, we proposed a fusion strategy that tries to make these two modes to fully display their respective advantages in one framework. The key idea of this work is illustrated in Fig.8. When the targets do not interact with each other, the laser scanner can perform the efficient tracking and it is easy for us to extract visual information from the camera data. Due to the reliability of these tracking results, they are used as positive or negative samples to train some classifiers for the "possible interacting targets". When the targets are in close proximity, the learned classifiers and visual information will in turn assist in tracking. This mode of cooperation between laser and vision, and between tracking and learning, offers several advantages: (1) Laser and vision can fully display their respective advantages (fast measurements of laser scanners and rich information of cameras) in this system. (2) Because the "possible interacting targets" are represented by a discriminative model with a supervised learning process, the method can employ information from the "confusing targets" and can sufficiently exploit the targets' history. Through these discriminative models, we can easily deal with some challenging situations in the tracking. (3) This "tracking-learning adaptive loop" ensures that the entire processes can be completely on-line and automatic.

3 Omni-Directional Sensing of a Dynamic Environment Using an Intelligent Vehicle

This research focuses on the sensing technologies of intelligent vehicles. We intend to develop a car of omni-directional eyes looking at the environment of both static and dynamic objects, where the car detects the moving objects in surrounds, and tracks their states, such as speed, direction, and size, so that dangerous situations can be predicted. Moreover, the car can be also used to generate a 3D copy of the dynamic urban scenery that contains both stationary objects, e.g. buildings, trees, road, and mobile objects, e.g. people, bicycles and cars. Here, we need to consider the following issues: finding the vehicle's pose as it moves around, detecting and tracking the moving objects in surroundings, and generate a 3D representation of the whole environment.

An intelligent vehicle system has been developed as shown in Fig. 9, where five single layer laser scanners (briefly noted by ”L”) are mounted on the car to profile object geometry along the streets from different viewpoints and with different directions; a video camera is also integrated to monitor the front of the vehicle and obtain textures; a GPS (Global Positioning System)/IMU (Inertial Measurement Unit) based navigation unit is applied to give outputs of the vehicle pose. Sensor layouts might be varied according to applications. However, a common and important issue


Server PC
Video Camera
Client PC
Laser Scanner
Client PC
Laser Scanner
10/100 Base LAN
Sensor Fusion
Extrinsic Calibration
Moving Points Detection and Clustering
Background Model
Judge Tracking Situations
Tracking Situations Switch
Interacting Targets
Non-correlated Targets
Independent Tracking
Visual Information Extraction
On-line Learning and Updating
Projected
Interacting Targets Tracking (learning for tracking)
Joint Tracking
Assist in Tracking
Classifier
Non-correlated Targets Tracking (tracking for learning)

A snapshot of the experiment site: the blue points were laser scanner date after background subtraction and the box was utilized for extrinsic calibration.

Fig. 8 Fusion of laser and vision

here is how to fuse such a large number of sensors, so that the multi-modal sensing data can conduct the above missions, while a comprehensive perception that overcomes the shortages of each singular sensor is achieved.

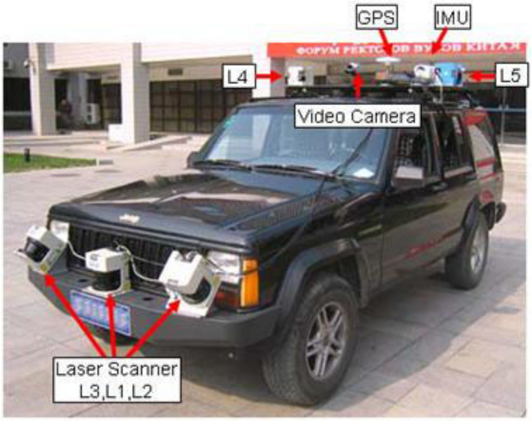


Fig. 9 A picture of the intelligent vehicle.

3.1 System Architecture

Normally, localization of the host vehicle is conducted by using the positioning sensors such as GPS, IMU, and VMS (Vehicle Motion Sensor). Thus the vehicle pose that contains the position (x,y,z) and orientation (ω, ϕ, κ) of the host vehicle is estimated in a high frequency. With it as input, the environmental sensors such as laser scanners, radars, video cameras are exploited, so that the sensing of local environments as the intelligent vehicle running along streets can be integrated to generate a global knowledge of the whole environment.

Such an approach is widely accepted in existing intelligent vehicles and mobile mapping systems. Localization and environmental perception are conducted individually using different sensing technologies. The system architecture is straightforward. However, disadvantage of such an approach is that the environmental perception is heavily dependent on outputs of the localization module. For example, erroneous localization outputs yield displacements between the environmental sensing data to the same static objects, and the slow motion objects such as pedestrians might not be reliably detected due to localization errors.

In this research, the system architecture is designed as shown in Fig. 10, where localization of the host vehicle is formulated as a SLAM with MODT (Simultaneous Localization And Mapping with Moving Object Detection and Tracking) by fusing both the positioning sensors (GPS/IMU) and the environmental sensors (laser scanners/cameras). We proposed a method of integrating both positioning (GPS/IMU) and environmental (laser scanners) sensors to improve localization accuracy, meanwhile, to conduct a 2D mapping and moving object detection/tracking. With these as inputs, other environmental sensors can perform an advanced (e.g. 3D) environmental perception with high accuracy.

In order to achieve the purpose, a sensor fusion strategy and software implementation is proposed as shown in Fig. 11. There are four processors inside the vehicle, for sensor data logging, SLAM with MODT, 3D mapping, moving object (briefly ”MO”) recognition, respectively. All the processors are connected through Ethernet, so that the data processing and transferring are conducted in an online mode. The processor 1 controls all the sensors, records GPS/IMU/laser scan data, and distribute them to other processors through Ethernet. The processor 2 will receive the

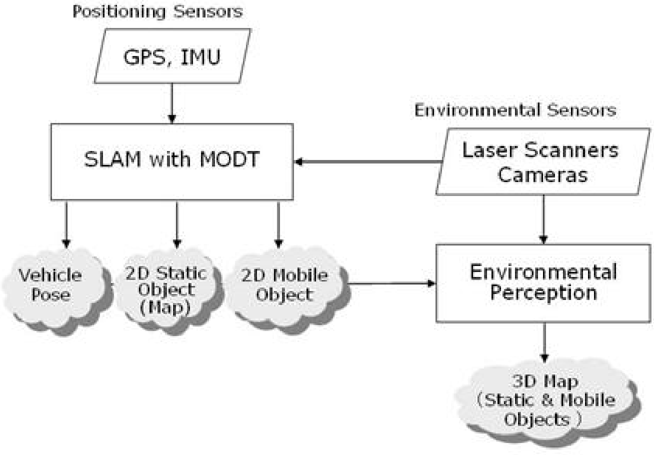


Fig. 10 System Architecture.

GPS/IMU data and the horizontal scanning laser data (L1) to conduct SLAM with MODT [7],[8]. The laser data of moving objects are forwarded to the processer 4, where each object is recognized as a person, a group of people, a bicycle or a car by fusion with video images [9] . On the other hand, the vehicle pose estimated by processor 2 is forwarded to processor 3, as well as the other processing results on static and moving objects. A 3D mapping is conducted by processor 4, which with the input of vehicle poses and sensor geometry parameters, integrates all laser scan data in a global coordinate system [10] .

3.2 SLAM with MODT

A laser-based SLAM is proposed in our previous research [7] , where the problem is formulated as SLAM with object tracking and classification, and the focus is on managing a mixture of data from both dynamic and static objects in a highly cluttered environment. Here people and cars might get very close to each other, and their motion patterns have much variability and are always unpredictable. Thus, it is risky to discriminate moving or static objects just by buffering an area using the data from a previous measurement. Also, it is risky to judge based only on an instance measurement, as many objects might have similar data appearance due to limited spatial resolution, range error, partial observation, and occlusions. The general idea behind our system is that the detected objects should be discriminated in a spatial-temporal domain. In this way, after an object is detected, it is tracked until the system can classify the object into either a static or moving object with certainty. On the other hand, in order to achieve a localization of global accuracy and robustness, especially when the vehicle makes a non-cyclical measurement in

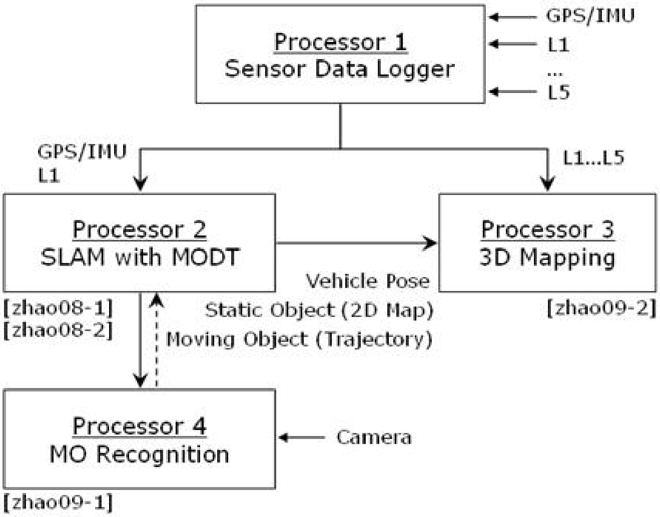


Fig. 11 Sensor fusion and software modules.

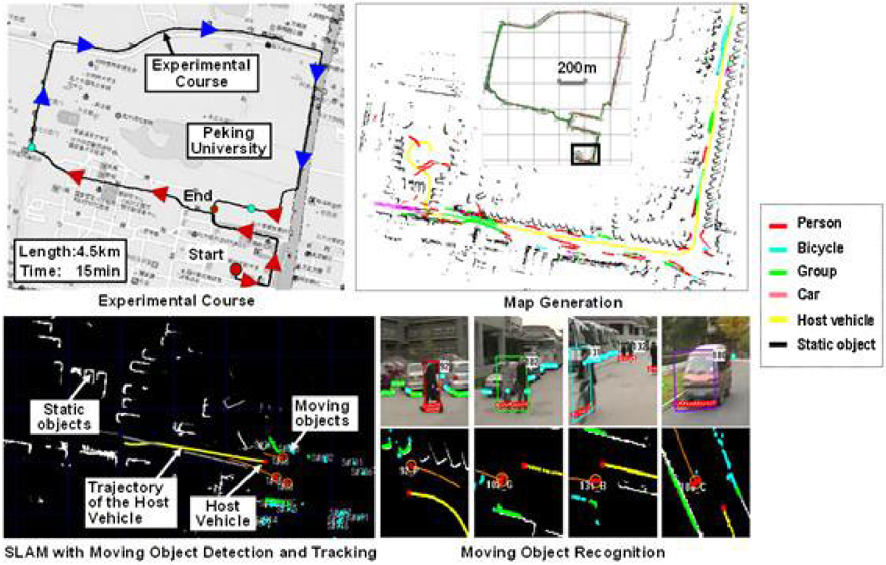


Fig. 12 Experimental results on SLAM with MODT and MO recognition.

a large outdoor environment, a GPS/IMU assisted strategy is also developed. The sporadically-available GPS measurements in urban areas are used to diagnose errors in the vehicle pose estimation, and the vehicle trajectory is then adjusted to close the gap between the estimated vehicle pose and the GPS measurement. An extensive experiment is presented in [8] ,where the host vehicle ran a course about 4.5km in a highly dynamic environment, and a map is generated containing the data of both the static and moving objects observed along the road. Some of the results are demonstrated in Fig. 12.

3.3 Sensor Alignment and 3D Mapping

A method is developed in [10] for calibrating the sensor system of multiple single-layer laser scanners. The problem is formulated into registration of laser point clouds of different laser scanners in a short run where the relative vehicle poses are considered of necessary accuracy. The registration is conducted in two steps: horizontal and vertical matching, using the laser points of vertical objects and ground surfaces respectively. An experiment is shown in Fig. 13, demonstrating the results before and after the sensor alignment. After the sensor alignment, the laser points from different laser scanners can be integrated into a global coordinate system with more consistency. In Fig. 14, a result is shown for integrating the laser points from laser scanners L1 (colored trajectories), L2 (on-the-road in gray, off-the-road in green), L4 (blue) and L5 (red). In the center of the view, a person is captured by the laser scanner L5 (red). It is easy to know that the person is walking on the road, with the laser points

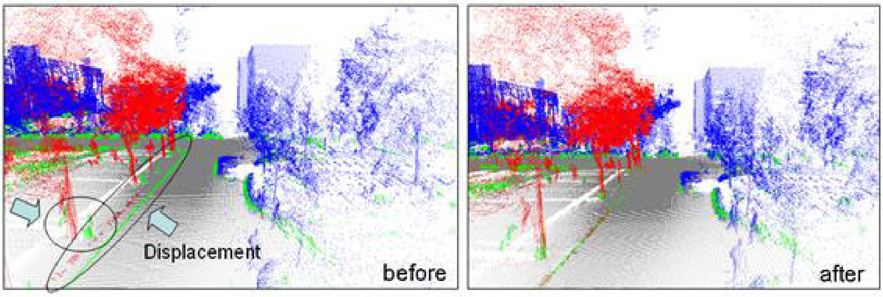


Fig. 13 Experimental results of sensor alignment.

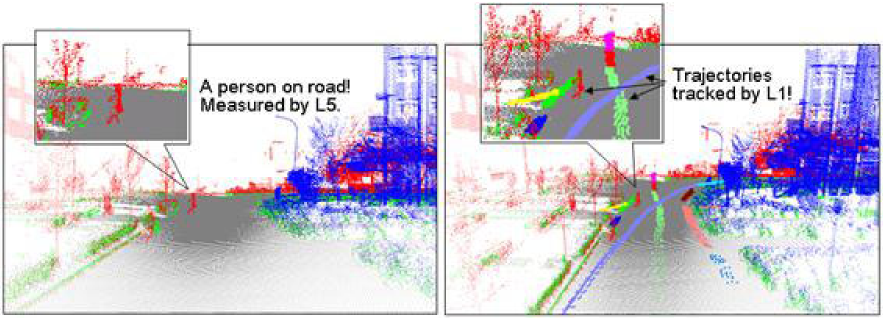


Fig. 14 An omni-directional view of the dynamic environment.

measured by the laser scanner L2 nearby its feet being colored with gray. A trajectory of the person is also known, which is captured by the laser scanner L1 (colored trajectories), so that we can predict where the person is walking ahead, etc.

4 Camera Calibration

Camera calibration is a process of modeling the mapping between 3D objects and their 2D images, and this process is often required when recovering 3D information from 2D images, such as in 3D reconstruction and motion estimation. The parameters of a camera to be calibrated are divided into two classes: intrinsic and extrinsic. The intrinsic parameters describe the camera's imaging geometric characteristics, and the extrinsic parameters represent the camera's orientation and position with respect to the world coordinate system. Many approaches to camera calibration have been proposed and they can be classified into two categories: using calibration objects and self-calibration. We have investigated those using spheres, conics and circles [11], [12],[13]. Here we describe some results given in [12] and [13].

4.1 Calibration from Unknown Principal-Axes Aligned Central Conics

Conics are one of the most important image features like points and lines in computer vision. The motivation to study the geometry of conics arises from the facts that conics have more geometric information, and can be more robustly and exactly extracted from images than points and lines. In addition, conics are very easy to be produced and identified than general algebraic curves, though general algebraic curves may have more geometric information. In our research [13], we discovered a novel useful pattern, principal-axes aligned. Moreover, the properties of two arbitrary principal-axes aligned conics with unknown or known eccentricities are deeply investigated and discussed.

These properties are obtained by utilizing the generalized eigenvalue decomposition of two principal-axes aligned conics. We define the absolute points of a conic in standard form, which is analogy of the circular points of a circle. Furthermore, we define the dual conic formed by two absolute points, which is analogy of the dual conic consisted of circular points. By using the dual conic formed by two absolute points, we proposed a linear algorithm to obtain the extrinsic parameters of the camera. We also discovered a novel example of the principal-axes aligned conics, which is consisted of a circle and a conic concentric with each other while the parameters of the circle and the conic are both unknown, and two constraints on the IAC can be obtained from a single image of this pattern.

4.2 Calibration from a Circle and a Coplanar Point at Infinity

In a large scene, like football or basketball courts, very large calibration patterns may be required. In most of such cases, however, setting a large planar pattern on the site is infeasible. One may suggest that the lines in the courts may be used for calibration. However, it requires that the locations of these lines must be given in advance. Another problem is that, the field of view (FOV) of a camera usually involves a portion of the whole court, and this often makes the number of lines in FOV not sufficient to estimate any intrinsic or extrinsic parameters of the camera. In the case of calibrating multiple cameras distributed in the football or basketball scenes, cameras with common FOVs are often required. It is not difficult to achieve this by changing the orientations of all cameras in the scenes to view the center circle and the midfield line.

The main contributions of our another research [13] is: It is the first work to deal with the problem of camera calibration just using the information in the midfield, i.e., the center circle and the midfield line. It shows that the camera calibration using one circle and one line passing through the circle's center, and that using one circle and one point at infinity in the support plane of the circle, are equivalent. From the latter one, the derivations may become very clear. It may be used for multiple camera calibration or camera network calibration in football or basketball scenes.

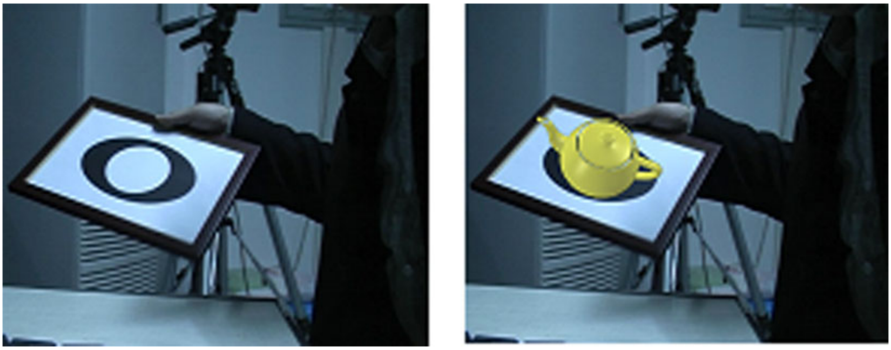


Fig. 15 Some calibration results

We performed a number of experiments, both simulated and real, to test our algorithms with respect to noise sensitivity. In order to demonstrate the performance of our algorithm, we capture an image sequence of 209 real images, with resolution 800∗ 600, to perform an augmented reality task. Edges were extracted using Canny's edge detector and the ellipses were obtained using a least squares ellipse fitting algorithm. One of the examples is shown in Fig. 15 to illustrate the calibration results.

5 Discussion

When dealing with unknown, unstructured and dynamic working environments, robots need to recognize both static and dynamic objects of the scene. This paper gives an overview of our recent research on moving target tracking and 3D scene modeling via multi-sensor perception systems in a multi-view or mobile platform. 3D information from laser scanning data complements the missing depth in video images. Therefore 3D scene modeling and target detection/tracking become much easy and robust. On the other hand, assistance from visual data improves the recognition accuracy by allowing many machine learning algorithms to be put into use easily. The results showed that the proposed methods can find many potential applications in robotics and other fields such as intelligent transportation and surveillance.

Acknowledgements. This work was supported in part by NKBRPC (No.2006CB303100), NSFC Grant (No.60333010), NSFC Grant (No.60605001), the NHTRDP 863 Grant (No.2006AA01Z302) and NHTRDP 863 Grant (No.2007AA11Z225).

References

1. Cui, J., Zha, H., Zhao, H., Shibasaki, R.: Fusion of detection and matching based approaches for laser based multiple people tracking. In: Proc. IEEE Int. Conf. on Computer Vision and Pattern Recognition, pp. 642–649 (2006)
2. Song, X., Cui, J., Zha, H., Zhao, H.: Probabilistic detection-based particle filter for multi-target tracking. In: Proc. of British Machine Vision Conference, pp. 223–232 (2008)
3. Song, X., Cui, J., Zha, H., Zhao, H.: Vision based mutilple interacting targets via on-line supervised learning. In: Proc. European Conference on Computer Vision, pp. 642–655 (2008)
4. Song, X., Cui, J., Zha, H., Zhao, H.: Tracking interacting targets with laser scanner via on-line supervised learning. In: Proc. IEEE International Conference on Robotics and Automation, pp. 2271–2276 (2008)
5. Cui, J., Zha, H., Zhao, H., Shibasaki, R.: Multimodal tracking of people using laser scanners and video camera. Image and Vision Computing (IVC), 240–252 (2008)
6. Song, X., Cui, J., Zhao, H., Zha, H.: A Bayesian approach: fusion of laser and vision for multiple pedestrains detection and tracking. International Journal of Advanced Computer Engineering, IJACE (2009)
7. Zhao, H., Chiba, M., Shibasaki, R., Shao, X., Cui, J., Zha, H.: SLAM in a dynamic large outdoor environment using a laser scanner. In: Proc. IEEE Int. Conf. on Robotics and Automation, pp. 1455–1462 (2008)
8. Zhao, H., Chiba, M., Shibasaki, R., Katabira, K., Cui, J., Zha, H.: Driving safety and traffic data collection - a Laser scanner based approach. In: Proc. IEEE Intelligent Vehicles Symposium, pp. 329–336 (2008)
9. Zhao, H., Zhang, Q., Chiba, M., Shibasaki, R., Cui, J., Zha, H.: Moving object classification using horizontal laser scan data. In: Proc. IEEE Int. Conf. on Robotics and Automation (2009)
10. Zhao, H., Xiong, L., Jiao, Z., Cui, J., Zha, H.: Sensor alignment towards an omni-directional measurement using an intelligent vehicle. In: Proc. IEEE Intelligent Vehicle Symposium, pp. 292–298 (2009)
11. Ying, X., Zha, H.: Geometric interpretations of the relation between the image of the absolute conic and sphere Images. IEEE Trans. on Pattern Analysis and Machine Intelligence (PAMI) 28(12), 2031–2036 (2006)
12. Ying, X, Zha, H.: Camera calibration using principal-axes aligned conics. In: Proc. 8th Asian Conf. on Computer Vision, 138–148 (2007)
13. Ying, X, Zha, H.: Camera calibration from a circle and a coplanar point at infinity with applications to sports scenes analyses. In: Proc. 2007 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, 220–225 (2007)

Towards Lifelong Navigation and Mapping in an Office Environment

Gordon Wyeth and Michael Milford

Abstract. This paper addresses the challenge of developing robots that map and navigate autonomously in real world, dynamic environments throughout the robot's entire lifetime — the problem of lifelong navigation. Static mapping algorithms can produce highly accurate maps, but have found few applications in real environments that are in constant flux. Environments change in many ways: both rapidly and gradually, transiently and permanently, geometrically and in appearance. This paper demonstrates a biologically inspired navigation algorithm, RatSLAM, that uses principles found in rodent neural circuits. The algorithm is demonstrated in an office delivery challenge where the robot was required to perform mock deliveries to goal locations in two different buildings. The robot successfully completed 1177 out of 1178 navigation trials over 37 hours of around the clock operation spread over 11 days.

1 Introduction

A mobile robot can achieve its goals more efficiently if it can remember the layout of its surroundings, maintain a representation of its own position, and use its memory of the spatial layout to execute paths to goal locations. The problem of getting a robot to build a spatial representation of its world and to localize within that spatial representation has been extensively studied as the problem of Simultaneous Localization and Mapping (SLAM). There are several solutions to the SLAM problem,

Gordon Wyeth
Information Technology and Electrical Engineering, University of Queensland, Australia
e-mail: wyeth@itee.uq.edu.au

Michael Milford
Information Technology and Electrical Engineering, University of Queensland,
Australia and Queensland Brain Institute, University of Queensland, Australia
e-mail: milford@itee.uq.edu.au

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 589–603.
springerlink.com

and many demonstrations of highly accurate mapping ability (such as [6, 13]), yet it is widely recognized that these solutions are not finding their way into practical applications with autonomous robots [14]. The existing solutions are geared towards building maps based on static features, and assume that the mapped features of the surroundings will not change. SLAM maps are not easily updated, and rapidly become out of date as alterations to the robot's surroundings accrue. In visual SLAM, where the robots use cameras to detect features, the problem is exacerbated by constant visual changes caused by variations in lighting. For a map to be useful to a robot, the map must adapt to all kinds of change: rapid and gradual, transient and permanent, visual and functional.

Animals, on the other hand, seem to adapt to changes in spatial layout with ease. The navigation ability of the rodent has been widely studied in biology at both a behavioral and neural level [4]. Rats can forage over ranges of kilometers, remembering sources of food, avoiding dangerous areas, and returning to the nest [5]. The rat navigates equally well through urban back alleys, under houses, through pipes, and across grasslands. All of these environments are constantly changing from the activities of other animals or humans, and from the weather and the season. Biologists inspired by the remarkable navigation performance of the rodent family have made major advances in unraveling the neural circuitry that stores the map, localizes the rat and plans the paths. In this paper, we use that neural circuitry as inspiration for a solution to the problem of lifelong mapping and navigation for a mobile robot.

There have been a number of robot systems built based on the neural circuitry of the rat: some have been designed to test biological ideas (for example, [1]), others have explored improvements for robotic applications (for example, [2]). Our previous work has been focused on the application of the rodent inspired algorithm, RatSLAM, to challenging robot problems. RatSLAM has been demonstrated mapping indoor environments [10], a university campus [11], and an entire suburb [8]. In this paper, we show that the maps built by RatSLAM are well suited to planning for navigation, and readily maintainable in changing environments.

The challenge we set for RatSLAM in this paper was to build a map suitable to plan "deliveries" in two working office environments operating at all times in the 24 hour day/night cycle over a two week period. The robot was given an initial period to autonomously explore the first environment and generate a user readable map. The delivery locations were then marked on that map, and delivery requests randomly generated for over 1000 trials. The robot was then moved (without notice) to a second building where it built a new map, and performed deliveries to new locations. The robot was finally returned to the original building where it autonomously relocalized and performed deliveries to locations in the original map. While performing deliveries in both buildings, the robot autonomously found and remembered the location of its charger, and autonomously recharged its batteries as required.

The paper describes the RatSLAM system in the next section, detailing how the system can be used to build maps that remain stable in size and computation requirements over time. After describing the delivery challenge in detail, the paper

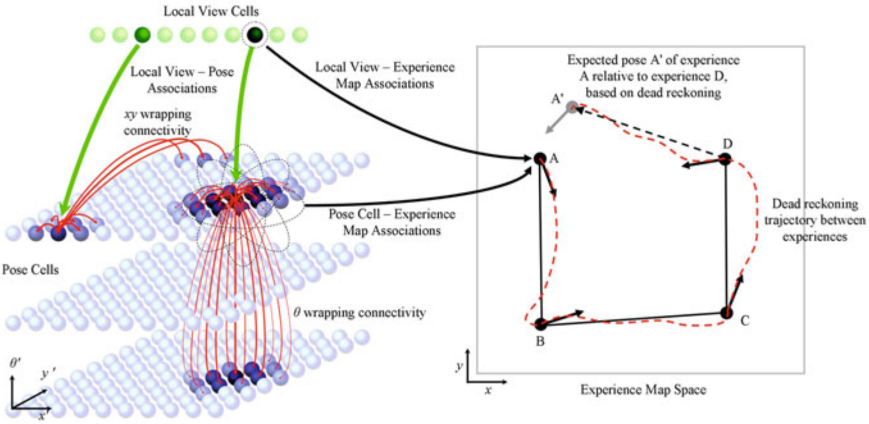

Local View Cells
Local View – Pose Associations
Local View – Experience Map Associations
xy wrapping connectivity
Pose Cells
Pose Cell – Experience Map Associations
θ wrapping connectivity
θ'
y'
x'
Expected pose A' of experience A relative to experience D, based on dead reckoning
A'
A
B
C
D
Dead reckoning trajectory between experiences
y
x
Experience Map Space

Fig. 1 The RatSLAM system. Each local view cell is associated with a distinct visual scene in the environment, and becomes active when the robot sees that scene. A three-dimensional continuous attractor network forms the pose cells, where active pose cells encode the estimate of the robot's pose. Each pose cell is connected to proximal cells by excitatory and inhibitory connections, with wrapping across all six faces of network. Intermediate layers in the (x', y') plane are not shown. The network connectivity leads to clusters of active cells known as activity packets. Active local view and pose cells drive the creation of experience nodes in the experience map, a semi-metric graphical representation of places in the environment and their interconnectivity.

then shows results that illustrate both reliability and stability of the mapping and navigation system. A more detailed and authoritative version of this study can be found in [9].

2 RatSLAM

RatSLAM has three principal components: the *pose cells* which use odometry to provide a locally consistent spatial reference, the *local view cells* which provide the interface to the robot's external sensors, and the *experience map* which fuses the information in the pose cells and the local view cells to provide a representation suitable for autonomous navigation. The components of RatSLAM and the interactions of the components are illustrated in Figure 1, and described briefly in the following sections. Further details of the operation of RatSLAM can be found in [8, 9].

2.1 Pose Cells

The pose cells are a three-dimensional Continuous Attractor Network (CAN), a type of neural network that consists of an array of neural units [12]. Unlike other neural networks that operate by changing the value of connections between neural units, the

CAN predominantly operates by varying the activity of the neural units while keeping the connection strengths fixed. During operation, the pose cell network will generally have clusters of highly active units: *activity packets*. The active cells of an activity packet provide a representation of the robot's pose that is consistent with the pose cell network's rectangular prism structure, as shown in Figure 1. Each of the three dimensions of the prism corresponds to one of the three spatial dimensions x', y', and θ'. Primed co-ordinates are used as the pose cells' representation of space is heavily distorted and aliased. To interpret the pose of the robot from the pose cells, the activity packet(s) must be transformed into the more useful experience map representation. The purpose of the pose cells and the activity packet is to provide a representation that is readily associated with external perception through the local view.

Attractor Dynamics

An activity packet is self-maintained by fixed local excitatory connections that increase the activity of units that are close in (x', y', θ') space to an active unit. Fixed inhibitory connections suppress the activity of smaller clusters of activity elsewhere in the network. Connections wrap across all six faces of the pose cell network, as shown by the longer red arrows in Figure 1. The change in the cells' activity level ΔP is given by:

$$\Delta P = P * \varepsilon - \phi \tag{1}$$

where P is the activity matrix of the network, ε is the connection matrix, and $*$ is the convolution operator. As well as the inhibition in the connection matrix, the constant ϕ creates further global inhibition. At each time step, activation levels in P are restricted to non-negative values and the total activation is normalized.

Path Integration

Path integration involves shifting the activity packet in the pose cell network based on odometry information. At each time step, RatSLAM interprets the odometry information to displace a copy of the current activity state in the pose cell network. The path integration process can cause a cluster of activity in the pose cells to shift off one face of the pose cell structure and wrap around to the other, as is shown in both packets in Figure 1, one of which is wrapping across the θ' boundary, the other across the y' boundary. Recording from a single cell under path integration will create firing fields with rectangular tessellations, similar to the triangular tessellations seen in a rodent's grid cells found in entorhinal cortex [7].

2.2 *Local View Cells*

The local view cells produce a sparse vector based on a classification of the robot's external perception. A local view cell is created every time the robot sees a scene that is distinct from any other scene that the robot has seen before. Each local view cell is paired with a template of its associated distinct scene. If the scene is viewed again then the local view cell will become active.

Scene Recognition

In this implementation of RatSLAM, the external perception for the local view system was driven entirely from panoramic images obtained from a camera facing vertically upwards at a parabolic mirror, mounted at the central rotation axis of the robot. Some typical images are shown in Figure 2. The image is unwrapped to 128 pixels representing 360° in the horizontal dimension and 20 pixels representing 90° in the vertical dimension. Gain and exposure control are applied to the entire image, with patch normalization on the lower half of the image. In order to provide rotationally invariant matching, each row in the image is transformed to a set of Fourier coefficients. Image similarities between the current image and template images are calculated using the multiplication of the Fourier coefficients, which is equivalent to convolution in image space:

$$C = F^{-1}\left[\sum_{y=1}^{h} F\left(i_y\right)\cdot F\left(r_y\right)\right] \tag{2}$$

where $F()$ is the Fourier Transform operator and i_y and r_y are the pixels rows at y in the current and template images, respectively. The value of the maximum real correlation coefficient gives the quality of the match m:

$$m = \max\left(\mathrm{Re}\left(C\right)\right) \tag{3}$$

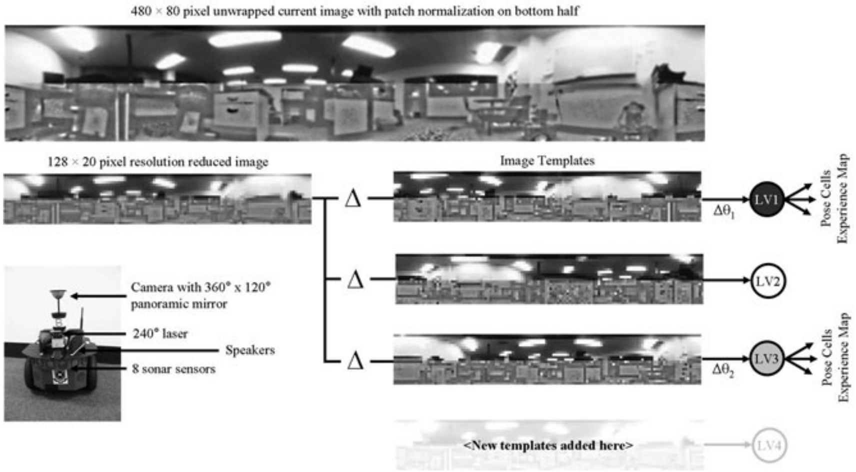

480 × 80 pixel unwrapped current image with patch normalization on bottom half
128 × 20 pixel resolution reduced image
Image Templates
Δ
Δ
Δ
Δθ$_1$
LV1
Pose Cells
Experience Map
LV2
Camera with 360° x 120° panoramic mirror
240° laser
Speakers
8 sonar sensors
Δθ$_2$
LV3
Pose Cells
Experience Map
<New templates added here>
LV4

Fig. 2 Vision hardware and vision processing system. A camera and mirror produces panoramas which are unwrapped into 360 degree by 90 degree images. The image is then reduced in resolution, patch normalized to enhance local image contrast and correlated with all template images. Template images that are close matches to the current image activate local view cells, which link to the pose cells and experience map.

A new image template and local view cell is created if the best match for all current image-template image pairings is below a threshold m_{min}. The match quality scores for each image pair are used to set the activation levels for the corresponding local view cells:

$$V_i = \max(m_i, 0)\, \forall i \tag{4}$$

Multiple local view cells can be simultaneously active to varying degrees in regions with perceptual aliasing, and there is no competition imposed between the cells. The connections from the local view cells to the pose cells, and the spatio-temporal filtering properties of the pose cells, prevent perceptual aliasing from adversely affecting the representation built in the experience map.

Connecting Local View to Pose

RatSLAM increases the strength of connections between local view cells and pose cells that are active simultaneously. In other words, RatSLAM learns an association between a visual scene and the robot pose. During a loop closure event, the familiar visual scene activates local view cells with learnt connections to the pose cells representing the pose where the visual scene was first encountered. Due to the attractor dynamics of the pose cells, a single visual scene is not enough to force an immediate change of pose; several consecutive and consistent views are required to update the pose. The attractor dynamics temporally and spatially filter the information from the local view cells, providing rejection of spurious loop closure events.

The connections between local view cells and pose cells are stored in a connection matrix β, where the connection between local view cell V_i and pose cell $P_{x',y',\theta'}$ is given by:

$$\beta^{t+1}_{i,x',y',\theta'} = \max\left(\beta^{t}_{i,x',y',\theta'}, \lambda V_i P_{x',y',\theta'}\right) \tag{5}$$

where λ is the learning rate. When a familiar visual scene activates a local view cell, the change in pose cell activity, ΔP, is given by:

$$\Delta P_{x',y',\theta'} = \frac{\delta}{n_{act}} \sum_i \beta_{i,x',y',\theta'} V_i \tag{6}$$

where the δ constant determines the influence of visual cues on the robot's pose estimate, normalized by the number of active local view cells n_{act}. Figure 1 represents the moment in time when a strongly active local view cell has injected sufficient activity into the pose cells to cause a shift in the location of the dominant activity packet. The previously dominant activity packet can also be seen, which is less strongly supported by a moderately activated local view cell.

2.3 Experience Mapping

The experience map combines the activity pattern of the pose cells with the activity of the local view cells to create a topologically consistent and semi-metric map. The pose cells' representation of space is distorted and aliased, making it unsuitable for path planning. Often when a loop closure event occurs, the odometric error introduces a discontinuity into the pose cells' representation of space, creating two sets of cells that might represent the same area in space. Similarly, the wrapped connectivity of the pose cell network leads to pose ambiguity, where a pose cell encodes multiple locations in the environment, forming the tessellated firing fields seen in grid cells in the rat. The experience map does not contain the discontinuities and ambiguities of the pose cells.

The experience map contains representations of places combined with views, called experiences, e_i, based on the conjunction of a certain activity state P^i in the pose cells and the local view cells V^i. Links between experiences, l^{ij}, describe the spatio-temporal relationships between places. Each experience is positioned at a location $\mathbf{p}^i$, a position that is constantly updated based on the spatial connectivity constraints imposed by the links. Consequently, the complete state of an experience can be defined as the triple:

$$e_i = \left\{P^i, V^i, \mathbf{p}^i\right\} \tag{7}$$

Figure 1 shows the region of pose cells and the single local view cell associated with the currently active experience A.

Transition links, l_{ij}, encode the change in position, $\Delta\mathbf{p}^{ij}$, computed directly from odometry, and the elapsed time, Δt^{ij}, since the last experience was active:

$$l^{ij} = \left\{\Delta\mathbf{p}^{ij}, \Delta t^{ij}\right\} \tag{8}$$

where l^{ij} is the link from the previously active experience e_i to the new experience e_j. The temporal information stored in the link provides the travel time between places in the environment. Path planning is achieved by integrating the time values in the transition links starting at the robot's current location to form a temporal map. The fastest path to a goal experience can be computed by performing steepest gradient ascent from the goal experience to the current location.

Creating Experiences and Closing Loops

At each time step, the current pose cell and local view cell states are compared with each experience. If a previously stored experience matches the current state, it is chosen as the 'active' experience, and represents the best estimate of the robot's location within the experience map. If the activity state in the pose cells or local view cells is not sufficiently described by any of the existing experiences, a new experience is created using the current pose and local view cell activity states. The odometry information defines the initial location space of a newly created experience relative to the previous experience:

$$e_j = \left\{P^j, V^j, \mathbf{p}^i + \Delta\mathbf{p}^{ij}\right\} \tag{9}$$

When loop closure occurs, the relative position of the two linked experiences in the map will typically not match the odometric transition information between the two, as shown in the discrepancy between experience A and A′ in Figure 1. A relaxation method seeks to minimize the discrepancy between odometric transition information and absolute location in experience space, by applying a change in experience location $\Delta\mathbf{p}^i$:

$$\Delta\mathbf{p}^i = \alpha\left[\sum_{j=1}^{N_f}\left(\mathbf{p}^j - \mathbf{p}^i - \Delta\mathbf{p}^{ij}\right) + \sum_{k=1}^{N_t}\left(\mathbf{p}^k - \mathbf{p}^i - \Delta\mathbf{p}^{ki}\right)\right] \tag{10}$$

where α is a correction rate constant, N_f is the number of links from experience e_i to other experiences, and N_t is the number of links from other experiences to experience e_i. Equation 10 is applied iteratively at all times during robot operation; there is no explicit loop closure detection that triggers map correction. The effect of the repeated application of Equation 10 is to move the arrangement of experiences in experience map space incrementally closer to an arrangement that averages out the odometric measurement error around the network of loops.

Experience Pruning

The experience map algorithm will continue to add experiences about the changing state of the world as the robot operates. In this way, the robot constantly updates and maintains its representations of the world. For example, if the robot proceeds along a corridor on one day with an adjoining door open, then the robot will build an experience that captures the open door. If the door is closed on the next day, then a new experience will be constructed with the door closed. The experiences will overlap in the experience map, by virtue of their connectivity to surrounding experiences, effectively mapping the same position with the alternate states of the door. This is not an extension of the existing algorithm; it is a property inherited from the algorithm's biological origins.

The difficulty with using this method of map maintenance is that the robot will remember all experiences from throughout its lifetime, creating an unmanageable list of experiences to search and update. The list of experiences must be pruned to a manageable level in order to attain the goal of lifelong navigation and mapping. Experience pruning consolidates parts of the experience map which exceed a maximum spatial density threshold. In this way the number of experiences grows in proportion to the size of the environment that has been explored, but not with time.

Figure 3 illustrates an experience being pruned from the map. The pruning algorithm uses a grid overlaid on the experience map to tag squares that contain more than one experience. Other methods for choosing the experience to delete were investigated (using measures such as recency and connectivity) but were found to have no effect on performance. Existing transitions to removed experiences are either deleted or updated to link to another experience. The odometric information for the new link is inferred from the current positions of the experiences in experience space, while the temporal information is calculated by dividing the inter-experience distance by the average robot speed. If, after the pruning process, any local view

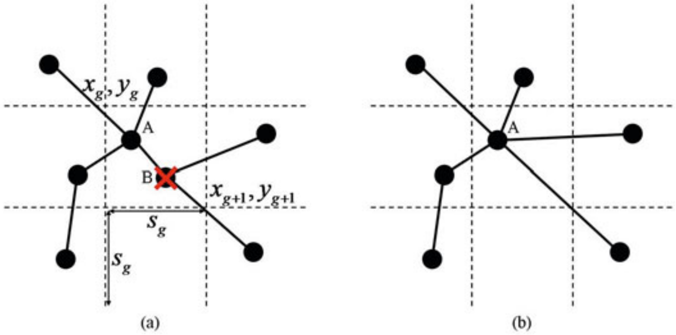


Fig. 3 Experience map pruning. Experiences are removed to maintain a one experience per grid square density. (b) Transitions to or from removed experiences are either deleted or reconnected to remaining experiences.

cells no longer define any experiences, the cells and the visual templates associated with them are removed. Any local view — pose cell links from these removed local view cells are also deleted.

3 The Office Delivery Challenge

The challenge set for the system was to perform the role of a delivery robot in a real world workplace over a two week period. The workplace consisted of floors in two different buildings at The University of Queensland in Brisbane, Australia, shown in Figure 2. The two buildings, Axon and General Purpose South (GP South), are typical research environments, moderately populated during the day and cleaned by janitors at night. The sizes of the two environments were 43 by 13 metres and 60 by 35 metres. The robot had to navigate through open plan office space, corridors, kitchens and laboratories. The environments were by no means static, with moving people, changing door states, re-arrangement of furniture and equipment, and a range of trolleys that intermittently appeared and disappeared during deliveries, maintenance and cleaning operation. Perceptually the environments also changed, with the most significant example being the day-night time cycles, which had an especially significant impact in areas with many windows.

We used a Pioneer 3 DX robot equipped with a panoramic imaging system, a ring of forward facing sensors, a Hokuyo laser range finder and encoders on the wheels (shown in Figure 2). All computation and logging was run onboard on a 2 GHz single core computer running Windows XP. The robot's typical operation speed was 0.3–0.5 ms^{-1}. The robot operated continuously for two to three hours between recharging cycles. Due to the need for supervision of the robot during operation (to ensure the integrity of the experiment), the total active time was limited to one to four recharge cycles in a typical day. In order to capture the effect of around-the-clock operation, experiments were conducted across all hours of the day and night.

3.1 Procedure

The experimenter first placed the charging station at a location in the Axon building. The experimenter then positioned the robot at a random location, turned it on, and initiated the software suite. The robot commenced exploration of the environment. After 117 minutes, the experimenter specified six desired delivery locations, by clicking on these locations in the experience map. Detecting the specification of delivery locations, the robot commenced a process of picking a delivery location at random and navigating to it to make a mock ‘delivery’. When the robot detected a low battery state, it navigated to the recharging dock and docked to recharge. During re-charging the robot powered down in a fashion that retained the map, but lost localization.

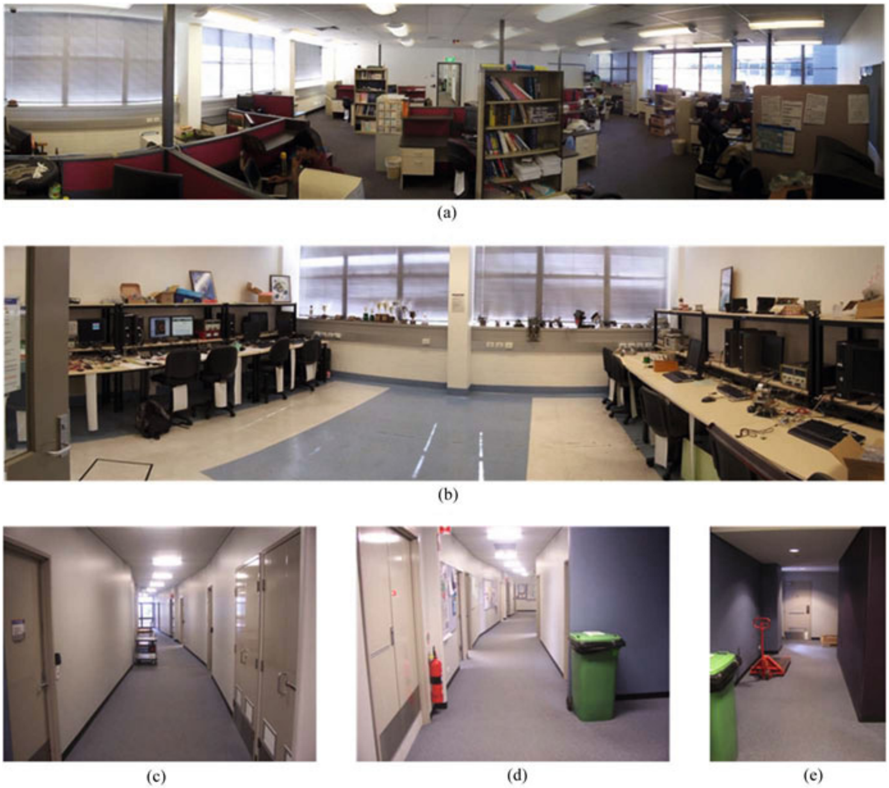


Fig. 4 Photos of the two environments. (a) Cluttered open plan office space in Axon building, note the number of windows. (b) Robotics laboratory in Axon building. (c-e) Corridors in GP South building. Panoramas generated using Autostitch software demo [3].

After eight days and 1017 delivery trials in the Axon environment, the experimenter turned off the robot and re-deployed it in the GP South environment. The experimenter also placed the charging station in the new environment, and then turned on the robot and initiated the software suite. To take advantage of the wider corridors, the experimenter made a single change to a movement parameter governing the top speed and obstacle clearance; this change was not required but enabled higher speed operation. The robot was not told that it had been put in a new environment, and since it had no path solutions to any existing delivery locations, commenced exploration. After 68 minutes, the experimenter specified five desired delivery locations, by clicking on the experience map. With delivery goals that were now accessible, the robot commenced random delivery navigation. After 54 delivery trials the robot returned to the charger to recharge, after which the robot was turned off and replaced in the original Axon environment, with no notification that it had changed environments. The robot commenced navigation to the original delivery locations. A further 72 delivery trials were conducted in the Axon building, before the experiment ended after 11 days.

4 Results

The results show the performance of the robot in its main task of navigating to delivery locations, and the stability of the performance over the 11 days of the trial. Results were obtained from analysis of more than nine gigabytes of data including video from the robot, logged every time the robot docked to recharge.

4.1 Delivery Performance

Over the entire experiment, the robot had 1177 successful navigation trials (including navigation for deliveries and for recharging) and only one failure. The robot failed to complete delivery trial number 579, in that it did not get to the delivery location within five minutes. This failure was due to an extended localization failure in the open space in the robotics laboratory pictured in Figure 4(b). The global navigation system provided a local path to the local navigation system which could not be fulfilled, and the robot became ‘stuck’ in the corner, before eventually timing out the delivery attempt and reporting a delivery failure. The robot laboratory was one of the most challenging environments for the mapping system, because robot motion was relatively unconstrained in the large open space, leading to many unique template sequences. The delivery accuracy was worst for the delivery location in this room.

Figure 5 shows the accuracy of the global navigation system with respect to ground truth when making deliveries. The global navigation system uses 0.1 m as the acceptable tolerance for a delivery to be deemed complete. Perfect localization at delivery would be shown as a delivery error of 0.1 m. Goal location 3 was the delivery point located in open space in the robot laboratory.

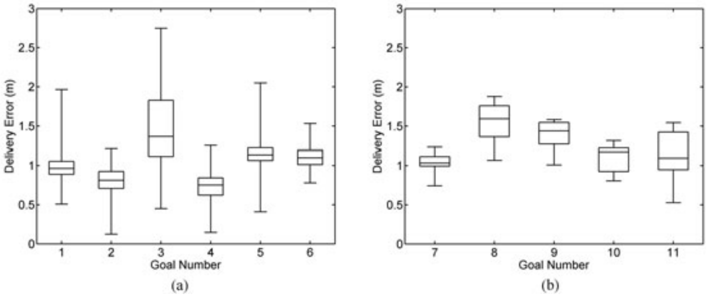


Fig. 5 Accuracy of delivery tasks, shown using a box whisker plot. Whiskers encompass the full range of delivery errors for each goal location.

4.2 Experience Maps

The experience map is the core global representation that forms the basis for navigation performance. In Figure 6, the experiences are plotted as a green circle at the (x, y) coordinates in the stored position $\mathbf{p}$. Linked experiences are joined with a blue line. The spatial integrity of the map is important for efficient operation of the pruning algorithm, while topological integrity is important for path planning and

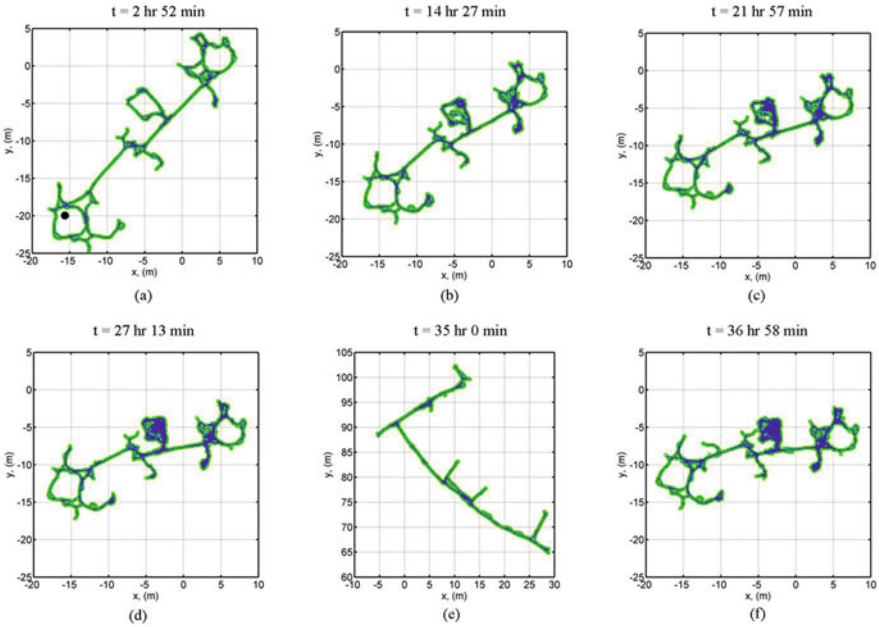


Fig. 6 Experience map progression for Axon and GP South buildings over the entire experiment. For the purposes of presentation, the experience map in the GP South building was initialized with an offset, to keep the maps separate.

navigation. The maps are not strictly accurate in a Cartesian sense, but need not be for effective navigation. For clarity, the map of GP-South in Figure 6(e) has been represented with an offset to the map of Axon. Note that there are no topological connections between the two buildings.

4.3 Stability

In order to address the question of the long term stability of the persistent navigation and mapping system, various measures were assessed to investigate trends in performance over time. The first indicator is the average time taken to navigate to each delivery location, plotted against time as in Figure 7. The delivery duration is dependent on the distance from the robot starting location and the randomly chosen delivery location. Durations are shown as averages over 100 deliveries, or over the period spent in the building whichever is shorter, smoothing the data sufficiently to obtain a measure of whether there is any increasing trend in navigation times as the map evolves. The graph shows no increase in delivery times, even after the robot has mapped another building.

The second indicator of stability used is the number of experiences and visual templates retained by the system over time, shown in Figure 8. The number of experiences and visual templates in use is stable after the first two hours, although the robot does gradually learn extra sections of the environment. In particular, the robotics laboratory (see Figure 4(b)) continued to build new experiences as the robot visited new areas of the room on subsequent visits. The extended coverage of the room is illustrated by the map evolution shown in Figure 6. About 1200 new experiences and 800 new visual templates were learnt in 90 minutes when the robot

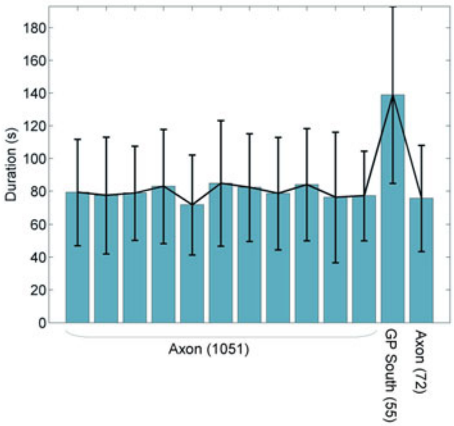


Fig. 7 Navigate-to-goal durations for the initial navigation trials in the Axon building (in groups of 100), General Purpose South building, and the final trials in Axon. Error bars show the standard deviation.

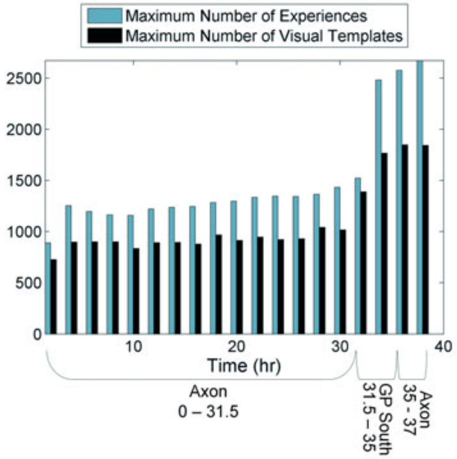


Fig. 8 Graph of the number of experiences and the number of view templates over the entire active duration of the experiment.

was moved to GP South, after which the number of experiences reaches a plateau of 2500. When placed back into Axon building at 35 hours, a small number of new experiences and templates are learned.

There were ample computational resources at all times during the eleven day experiment to maintain a 7 Hz rate of vision processing, localization and navigation planning. Spare compute time was used to run the experience map relaxation algorithm in Equation 10, which stabilized at 50 Hz in the initial Axon trials, and dropped to 25 Hz when the extra GP-South trials were included.

5 Conclusions

We presented the RatSLAM algorithm as a possible solution to the challenge of lifelong mapping and navigation. RatSLAM is significantly different to other SLAM algorithms: it relies on remembering sequences of observed features across a trajectory of poses, rather than geometric optimization of motion and feature measurements. Our biological analogy of memory for places is built in a system inspired by the neural computation believed to take place in the rodent hippocampus. The neural inspiration lends itself readily to a paradigm of learning, remembering and forgetting, rather than geometry, probability and optimization. The flexibility introduced by our paradigm shift enables the creation of maps that can adapt to the constant change found in real environments.

It is important to realize that functional integrity rather than absolute Cartesian accuracy is the key criteria for success in lifelong navigation and mapping. Performance metrics must be principally concerned with the measurement of goal attainment, rather than the accuracy of the underlying map. In this paper, we have presented a series of metrics that are of principal importance to the measurement of success of a lifelong navigation and mapping system. The rate of goal attainment ($> 99.9\%$) is the key metric, with metrics showing the stability of the number of retained elements in the map providing evidence that this system could continue to function with high reliability for months or possibly years.

References

1. Arleo, A., Gerstner, W.: Spatial cognition and neuro-mimetic navigation: A model of hippocampal place cell activity. Biol. Cybern. 83, 287–299 (2000)
2. Barrera, A., Weitzenfeld, A.: Biologically-inspired robot spatial cognition based on rat neurophysiological studies. Auton. Robot. 25, 147–169 (2008)
3. Brown, M., Lowe, D.: Autostitch, `http://www.cs.ubc.ca/mbrown/autostitch/autostitch.html`
4. Burgess, N., Donnett, J.G., Jeffery, K.J., O'Keefe, J.: Robotic and neuronal simulation of the hippocampus and rat navigation. Philos. Trans. R Soc. Lond. B Biol. Sci. 352, 1535–1543 (1997)
5. Davis, D.E., Emlen, J.T., Stokes, A.W.: Studies on home range in the brown rat. J. Mammal 29, 207–225 (1948)
6. Grisetti, G., Stachniss, C., Burgard, W.: Improved techniques for grid mapping with rao-blackwellized particle filters. IEEE Trans. Robot. 23, 34–46 (2007)
7. Hafting, T., Fyhn, M., Molden, S., Moser, M.-B., Moser, E.I.: Microstructure of a spatial map in the entorhinal cortex. Nature 11, 801–806 (2005)
8. Milford, M.J., Wyeth, G.: Mapping a suburb with a single camera using a biologically inspired slam system. IEEE Trans. Robot. 24, 1038–1053 (2008)
9. Milford, M.J., Wyeth, G.: Persistent navigation and mapping using a biologically inspired slam system. Int. J. Rob. Res. 11 (2009) (in press)
10. Milford, M.J., Wyeth, G., Prasser, D.: Ratslam: A hippocampal model for simultaneous localization and mapping. In: Proc. IEEE Int. Conf. Robot. Autom., New Orleans, USA (2004)
11. Prasser, D., Milford, M.J., Wyeth, G.: Outdoor simultaneous localisation and mapping using ratslam. In: Results Int. Conf. Field Serv. Robot., Port Douglas, Australia (2005)
12. Stringer, S.M., Rolls, E.T., Trappenberg, T.P., de Araujo, I.E.T.: Self-organizing continuous attractor networks and path integration: two-dimensional models of place cells. Netw. Comput. Neural Syst. 13, 429–446 (2002)
13. Thrun, S., Hahnel, D., Ferguson, D., Montemerlo, M., Triebel, R., Burgard, W., Baker, C., Omohundro, Z., Thayer, S., Whittaker, W.: A system for volumetric robotic mapping of abandoned mines. In: Proc. IEEE Int. Conf. Robot. Autom., Taipei, Taiwan (2003)
14. Thrun, S., Leonard, J.: Simultaneous localisation and mapping. In: Springer Handbook of Robotics, ch. 37, p. 871. Springer, Heidelberg (2008)

Multi-robot systems

Cédric Pradalier

Using multiple mobile robots appears as a natural evolution to the traditional robotic research. Introducing multiple agents increases the potential performance of the system: adding more degrees of freedom to manipulation tasks, adding parallelism to delivery tasks, adding more view-points for observation tasks. However, as shown in the following chapters, this comes with a significant increase in complexity of the control, planning, and data-fusion systems.

- In “Coordinating Construction of Truss Structures using Distributed Equal-mass Partitioning ”, Yun et al. propose a methodology for a multi-robot construction system, in particular in the context of multiple parts and specialised robots. The paper presents simulation results and discusses the stability and adaptation properties of the proposed method.
- In “Synthesis of Controllers to Create, Maintain, and Reconfigure Robot Formations with Communication Constraints ”, Ayanian et al. develop a framework for controlling groups of robots with high-bandwidth intragroup communication but limited inter-group communication. The chapter reports simulation results and discusses the complexity of the approach.
- In “Planning and Control for Cooperative Manipulation and Transportation with Aerial Robots ”, Fink et al. consider the problem of transporting a pay-load using multiple quad-copter. Individual robot control laws and motion plan are developed, and the chapter reports results both in simulation and with a set-up made of three quad-copters.
- In “Adaptive Highways on a Grid ”, Roozbehani and D'Andrea address the problem of planning paths for a multitude of real robots moving in a grid environment. This chapter considers the dynamic of the robots and presents results in simulation for several algorithms, in particular the Adaptive Highway Algorithm.
- In “Environment modeling for cooperative aerial/ground robotic systems”, Vidal et al. implement a SLAM approach suitable for the cooperation of multiple heterogeneous robots. Using a combination of ground and aerial platforms make this problem particularly challenging because of the extreme difference of view-point between the two types of robots.

- In “Energy-Efficient Data Collection from Wireless Nodes using Mobile Robots ”, Tekdas et al. address a different aspect of the synchronisation of multiple autonomous systems in the context of wireless sensor networks. In this case, the energy constraints are so high that a robot and a group of deployed sensors must synchronise their communication to optimise their global energy efficiency.

As mentioned earlier, the complexity increase due to the use of multiple robotic devices is significant. This is particularly visible in that many of the chapters in this part of the ISRR'09 proceedings only report results on simulated systems. Even when real platforms are used, rarely more than three are reported, with the exception of the Kiva systems mentioned in the work of Roozbehani and D'Andrea. A reason for the rarity of real multi-robot systems with number of robots with at least two digits is simply the complexity of maintaining, programming, and deploying a large number of robots, especially when the team is heterogeneous. Engineering solutions to this problem are provided by company such as Kiva in specific cases. Nonetheless, there is still a lot of space for research to provide generic solution to the communication problems, fault management and reporting, task planning and distribution, and the general programming and deployment problems.

Coordinating Construction of Truss Structures Using Distributed Equal-Mass Partitioning

Seung-kook Yun, Mac Schwager, and Daniela Rus

Abstract. This paper presents a decentralized algorithm for the coordinated assembly of 3D objects that consist of multiple types of parts, using a networked team of robots. We describe the algorithm and analyze its stability and adaptation properties. We instantiate the algorithm to building truss-like objects using rods and connectors. We implement the algorithm in simulation and show results for constructing 2D and 3D parts. Finally, we discuss briefly preliminary hardware results.

1 Introduction

We wish to develop cooperative robot systems for complex assembly tasks. A typical assembly scenario requires that parts of different types get delivered at the location where they are needed and incorporated into the structure to be assembled. We abstract this process with two operations: (1) tool and part delivery carried out by delivering robots, and (2) assembly carried out by assembling robots. In this paper, we consider how a team of robots will coordinate to achieve assembling the desired object. Tool and part delivery requires robots capable of accurate navigation between the part cache and the assembly location. Assembly requires robots capable of complex grasping and manipulation operations, perhaps using tools. Different assembling robots work in parallel on different subcomponents of the desired object. The delivering robots deliver parts (of different types) in parallel, according to the sequence in which they are needed at the different assembling stations. We consider the case where the parts are (a) rods of different lengths and (b) connectors for connecting the rods into truss-structured objects. The robots can communicate locally to neighbors. The delivering robots have the ability to find the correct part type in

Seung-kook Yun · Mac Schwager · Daniela Rus
Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA
e-mail: yunsk@mit.edu, schwager@mit.edu, rus@csail.mit.edu

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 607–623.
springerlink.com

the part cache, pick it up, and deliver it to the correct spot for the assembling process requesting the part, and return to the part cache for the next round of deliveries. The assembling robots have the ability to receive the part from a delivering robot and incorporate it into the assembly.

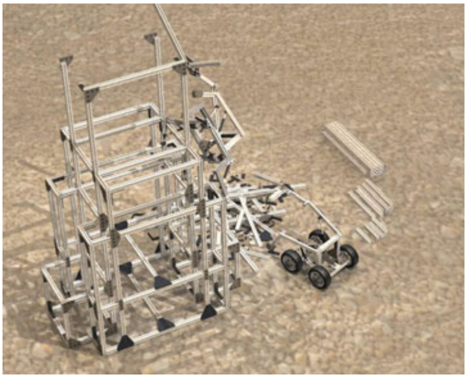


Fig. 1 Concept art for construction of a truss structure by mobile delivering robots and truss-climbing assembling robots. Reprinted with permission from Jonathan Hiller, Cornell University, USA.

We assume that the target object is given by a material-density function which encodes the object geometry and is known to all the robots. The construction process starts by a "coverage"-like process during which the assembling robots partition the target structure adaptively into sub-assemblies, such that each robot[1] is responsible for the completion of that section. To achieve this division, the robots locally compute a Voronoi partition, weighted by the mass of all the rods contained in the partition, and perform a gradient descent algorithm to balance the mass of the regions. The delivery robots also know the density function describing the target structure and the location of the parts. Each delivering robot carrying a part enters the assembling region and delivers the part to the region with the highest demanding mass. That is, the robot asks each assembling robot within communication range what is the current mass of the structure they have created and selects the site of the least completion. This ensures global and local balance for part delivery.

We describe decentralized control algorithms for the partition, part delivery, and assembling steps. The algorithms are inspired by the approach in [2, 9, 7] and use *equal-mass partitioning* as the optimization criterion. The algorithms rely on local information only (e.g. neighbors exchange information about their local mass). The task allocation and part delivery algorithms are provably stable. They are adaptive to the number of delivering robots and assembling robots as well as to the amount of source material. We implemented these algorithms in simulation. Several 2D and 3D truss-structures were created using our algorithms. We have started a hardware implementation using iCreate robots extended with Meraki communication and a CrustCrawler 4-dof robot arm. The part delivery algorithm has been implemented on these robots to demonstrate the coordination infrastructure of the system and the correctness of the delivery algorithm. Assembly execution is under development.

[1] The robot represents all the skills needed for each required assembly step; in some cases multiple robots will be needed, for example the connection of two rods with a screw is done by three robots, one robot holding each rod, and one robot placing the connector.

1.1 Related Work

This work combines distributed coverage and robotic construction. We follow the notion of locational optimization developed by Cortes et al. [2], who introduced distributed coverage with mobile robots. The same optimization criteria was used in a distributed coverage controller for real-time tracking by Pimenta et al. [8]. In our previous work, Schwager [9] used adaptive coverage control in which networked robots learn a sensory function while they are controlled for the locational optimization. This research inherits the distributed coverage concept, and pursues *equal-mass partitioning* in which every networked robot is controlled to have the same amount of construction (in our case, truss elements and connectors) to be built, rather than optimal sensing locations. Pavone et al. [7] have been independently working on equitable partitioning by the power diagram.

Algorithms and hardware have been developed for manipulator robots that climb and build a truss structure. SM^2, a truss-walking inspection robot, was developed for space station trusses [6]. Skyworker demonstrated truss-like assembly tasks [10]. Werfel et al. [11] introduced a 3D construction algorithm for modular blocks. Our previous work on truss assembling robots includes Shady3D [1, 4, 5] that utilizes a passive bar with active communication and may include itself in a truss structure, and is controlled by locally optimized algorithms. We also proposed a centralized optimal algorithm to reconfigure a given truss structure to a target structure [3]. This work introduces a framework in which robots are specialized as delivery and assembling robots, distributed algorithms control the assembly of a structure with multiple kinds of source materials.

2 Problem Formulation

We are given a team of robots, n of which are specialized as part delivering robots and the rest are specialized as assembling robots. The robots can communicate locally with other robots within their communication range. The robots are given a target shape represented as a mass-density function ϕ_t. We wish to develop a decentralized algorithm that coordinates the robot team to deliver parts so that the goal assembly can be completed with maximum parallelism.

Suppose for now that the robots move *freely* in an Euclidean space (2D and 3D). This assumption makes sense when the robots move in the plane to achieve a planar assembly. However, for 3D assemblies, factors such as gravity and connectivity of structure, as well as 3D motion for the robots, must be considered. We will generalize in Section 5.

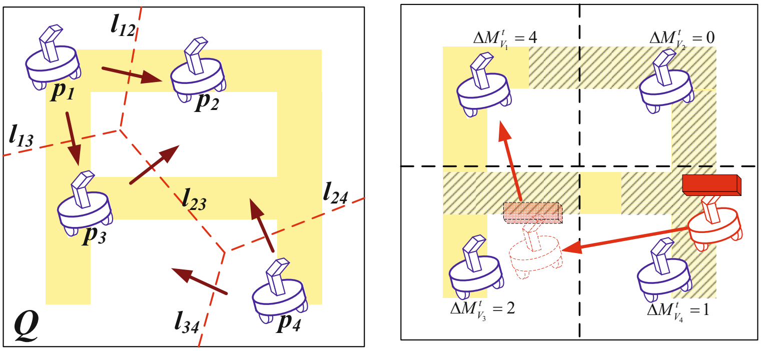
Algorithm 1 shows the main flow of construction in a *centralized* view. In the first phase, assembling robots spread in a convex and bounded target area $Q \subset \mathbb{R}^N (N = 2, 3)$ which includes the target structure. They find placements using a distributed coverage controller which assigns to each robot areas of the target structure that have approximately the same assembly complexity. In the second phase the delivering robots move back and forth to carry source components to the assembling robots. They deliver their components to the assembling robot with maximum *demanding mass*. The demanding mass is defined as the amount of a source component required for an assembling robot to complete its substructure. In this work, the source components include two types: truss elements and connectors. The truss elements are rods and they may be of different lengths. Details of the demanding mass for each type of the source components are presented in Section 4.1. After an assembling robot obtains a component from a delivering robot, it determines the optimal placement for this component in the overall assembly and moves there to assemble the component. The assembly phase continues until there is no source component left or the assembly structure is complete.

Algorithm 1. Construction Algorithm

```
Deploy the assembling robots in $Q$
Place the assembling robots at optimal task locations in $Q$ (Section 3)
**repeat**
    **delivering robots:** carry source components to the assembling robots (Section 4.2)
    **assembling robots:** assemble the delivered components (Section 4.1)
**until** task completed *or* out of parts
```

2.1 Example

Figure 2(a) shows a construction system with 4 assembling robots. Intuitively, robot 1 and robot 4 move towards the other robots in order to expand their partition, whereas robot 2 moves away from the other robots because it has the largest area.



(a) (b)

Fig. 2 Example of the equal-mass partitioning and delivery by the gradient of the demanding mass. 4 mobile manipulators (assembly robots) are displayed in a convex region Q that includes the A-shaped target structure. The yellow region has high density ϕ_t. The mass of a robot is the size of the total yellow region in its partition (Voronoi region.) $p_i (i = 1, 2, 3)$ denotes the position of the assembling robots and the red-dotted lines l_{ij} are shared boundaries of the partitions between two robots. $\Delta M^t_{V_i}$ is the demanding mass.

The moving direction of the robots is determined by combining the normals to the Voronoi edges. Figure 2(b) shows the red delivering robot carrying a red truss element driven by the gradient of the demanding mass. The yellow region denotes the target density function ϕ_t. The hashed region denotes completed assembly. The demanding mass of a region can be thought of as the difference between the area of yellow regions and the area of hashed regions.

Suppose a delivering robot is in the region of robot 4. Among its neighbors (robot 2 and 3) the maximum demanding mass is with robot 3. Thus the delivering robot moves to robot3. The delivering robot finds that robot 1 has the maximum demanding mass among robot 3's neighbors, therefore it advances to robot 1 and delivers the truss component. Following the maximum demanding mass gives a local balance for the target structure.

3 Task Allocation by Coverage with Equal-Mass Partitions

This section describes a decentralized equal-mass partitioning controller which is inspired by distributed coverage control [2, 9]. The algorithm allocates to each assembling robot the same amount of assembly work, which is encoded as the same number of truss elements. This condition ensures maximum parallelism. We continue with a review of the key notation in distributed coverage, then give the mass optimization criteria and end the section with the decentralized controller.

3.1 Equal-Mass partitioning

Suppose n assembling robots cover region Q with a configuration $\{\mathbf{p}_1, ..., \mathbf{p}_n\}$, where $\mathbf{p}_i$ is the position vector of the i^{th} robot. Given a point $\mathbf{q}$ in Q, the nearest robot to $\mathbf{q}$ will execute the assembly task at $\mathbf{q}$. Each robot is allocated the assembly task that included its Voronoi partition V_i in Q.

$$V_i = \{\mathbf{q} \in Q | \, \|\mathbf{q} - \mathbf{p}_i\| \leq \|\mathbf{q} - \mathbf{p}_j\|, \forall j \neq i\} \tag{1}$$

The target density function ϕ_t is the density of truss elements, and it is fixed during the construction phase. Given V_i, we define its mass property as the integral of the target density function in the area.

$$M_{V_i} = \int_{V_i} \phi_t(\mathbf{q}) d\mathbf{q} \tag{2}$$

We wish for each robot to have the same amount of assembly work. We call this *equal-mass partitioning*. The cost function can be modeled as the product of all the masses:

$$\mathcal{H} = \mathcal{H}_0 - \prod_{i=1}^{n} M_{V_i}, \tag{3}$$

where $\mathcal{H}_0$ is a constant and the bound of the product term as:

$$\mathcal{H}_0 = \left(\frac{1}{n}\sum_{i=1}^{n} M_{V_i}\right)^n = \left(\frac{1}{n}\int_Q \phi_t(\mathbf{q})d\mathbf{q}\right)^n. \tag{4}$$

The cost function is continuously differentiable since each M_{V_i} is continuously differentiable [8]. Minimizing this cost function leads to equal-mass partitioning, because of the relationship between the arithmetic mean and the geometric mean.

$$\frac{1}{n}\sum_{i=1}^{n} M_{V_i} \geq \sqrt[n]{\prod_{i=1}^{n} M_{V_i}}, \tag{5}$$

where the equality holds only if all the terms are the same. Therefore the prefect equal-mass partitioning makes the cost function zero. Using the cost function in (5), we have developed a decentralized controller that guarantees $\mathcal{H}$ converges to a local minimum.

3.2 Controller with Guaranteed Convergence

We wish for the controller to continuously decrease the cost function: $\dot{\mathcal{H}} \leq 0, t > 0$. Differentiating $\mathcal{H}$ yields

$$\dot{\mathcal{H}} = \sum_{i=1}^{n} \frac{\partial \mathcal{H}}{\partial \mathbf{p}_i}\dot{\mathbf{p}}_i. \tag{6}$$

When $\mathcal{N}_i$ is a set of neighbor robots of the i^{th} robot, each term of the partial derivatives is

$$\frac{\partial \mathcal{H}}{\partial \mathbf{p}_i} = -\sum_{j=i,\mathcal{N}_i} \frac{\partial M_{V_j}}{\partial \mathbf{p}_i} \prod_{k=\{1,\ldots,n\},k\neq j} M_{V_k} \tag{7}$$

$$= -\prod_{l\notin\{i,\mathcal{N}_i\}} M_{V_l} \sum_{j=i,\mathcal{N}_i} \frac{\partial M_{V_j}}{\partial \mathbf{p}_i} \prod_{k\in\{i,\mathcal{N}_i\},k\neq j}^{n} M_{V_k} \tag{8}$$

where

$$\frac{\partial M_{V_i}}{\partial \mathbf{p}_i} = \sum_{j\in\mathcal{N}_i} \mathcal{M}_{ij}, \quad \frac{\partial M_{V_j}}{\partial \mathbf{p}_i} = -\mathcal{M}_{ij} \tag{9}$$

$\mathcal{M}_{ij}$ is computed along the sharing edges (sharing faces in 3D) l_{ij} between V_i and V_j as in [8]:

$$\mathcal{M}_{ij} = \int_{l_{ij}} \phi_t(\mathbf{q}) \frac{\partial \mathbf{q}_{l_{ij}}}{\partial \mathbf{p}_i} \mathbf{n}_{l_{ij}} d\mathbf{q} = \int_{l_{ij}} \phi_t(\mathbf{q}) \frac{\mathbf{q} - \mathbf{p}_i}{\|\mathbf{p}_i - \mathbf{p}_j\|} d\mathbf{q} \tag{10}$$

where $\mathbf{n}_{l_{ij}}$ is a normal vector to l_{ij} as

$$l_{ij} = V_i \cap V_j, \quad \mathbf{n}_{l_{ij}} = \frac{\mathbf{p}_j - \mathbf{p}_i}{\|\mathbf{p}_i - \mathbf{p}_j\|}. \tag{11}$$

We can rewrite equation 6 as

$$\dot{\mathcal{H}} = -\sum_{i=1}^{n} \prod_{l \notin \{i, \mathcal{N}_i\}} M_{V_l} \sum_{j=i,\mathcal{N}_i} \frac{\partial M_{V_j}}{\partial \mathbf{p}_i} \prod_{k \in \{i,\mathcal{N}_i\}, k \neq j}^{n} M_{V_k} \dot{\mathbf{p}}_i \tag{12}$$

Let $\boldsymbol{J}_i$ denote the part of the partial derivative term $\frac{\partial \mathcal{H}}{\partial \mathbf{p}_i}$ which is related with the set $\{i, \mathcal{N}_i\}$.

$$\boldsymbol{J}_i = \sum_{j=i,\mathcal{N}_i} \frac{\partial M_{V_j}}{\partial \mathbf{p}_i} \prod_{k \in \{i,\mathcal{N}_i\}, k \neq j}^{n} M_{V_k} \tag{13}$$

Note that $\boldsymbol{J}_i$ is a vector. Given a velocity control for each robot, the decentralized controller that achieves task allocation is given by the control law:

$$\dot{\mathbf{p}}_i = k \frac{\boldsymbol{J}_i}{\|\boldsymbol{J}_i\|^2 + \lambda^2}, \tag{14}$$

where k is a positive control gain and λ is a constant to stabilize the controller even around singularities where $\|\boldsymbol{J}_i\|^2 = 0$.

Note that all the equations can be computed in a distributed way, since they only depend on the variables of the neighboring robots.

Theorem 1. *The proposed controller guarantees that $\mathcal{H}$ converges to either a local maximum or a global maximum.*

Proof. The proposed control input $\dot{\mathbf{p}}_i$ yields

$$\dot{\mathcal{H}} = -k \sum_{i=1}^{n} \frac{\|\boldsymbol{J}_i\|^2}{\|\boldsymbol{J}_i\|^2 + \lambda^2} \prod_{l \notin \{i,\mathcal{N}_i\}} M_{V_l}. \tag{15}$$

Since k and M_{V_l} are positive, each term of $\dot{\mathcal{H}}$ is always negative. In addition, the cost function is differentiable, and trajectories of robots are bounded in Q. Therefore, the controller keeps the cost function decreasing unless all the $\boldsymbol{J}_i$ are empty vectors (relocating the robots does not change the cost function), which implies a local minimum.[2]

[2] Pavone et. al [7] also developed equitable partitioning using power diagrams that are weighted generalized Voronoi diagrams. They used a different cost function as the average of inverse of the masses. They targeted a different application in the space of the multi-vehicle routing.

4 Delivery and Assembly Algorithms

Once the assembling robots are in place according to the equal-mass partitioning controller, construction may begin. State machines drive the delivering robots and the assembling robots. During construction we wish to distribute the source components (truss elements and connectors) to the assembling robots in a balanced way. Global balance is asymptotically achieved by a probabilistic target selection of delivering robots that uses ϕ_t as a probability density function. For local balance, the delivering robots are driven by the gradient of demanding mass defined as the remaining structure to be assembled by the robot. Robots with more work left to do get parts before robots with less work left. Each assembling robot waits for a new truss element or connector and assembles it to the most demanding location in *its Voronoi region*. Therefore, construction is purely driven by the density functions regardless of the amount of the source components and it can be done without an explicit drawing of the target structure. We ensure that all the processes of the controllers work in a distributed way and each robot needs to communicate only with neighbors. Details of the control algorithms are explained next.

4.1 Assembly Algorithm

Each assembling robot operates using a state machine as shown in Figure 3. The robot has the following states:

- IDLE
- WAITING: waiting for a new component
- MOVING: moving to the optimal location to add the part
- ASSEMBLING: adding the component to the assembly

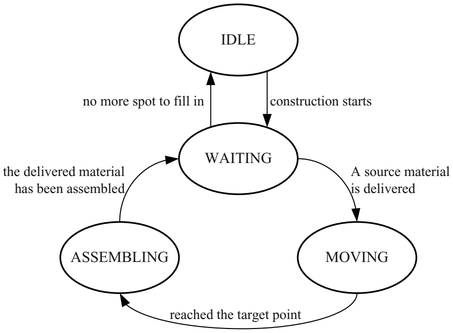


Fig. 3 The state machine for an assembling robot. Each assembling robot waits for the delivery of a source component, moves the component to the optimal spot and adds it to the structure. The robot's task is complete when there is no demanding mass left.

Each robot has a graph representation $G_i = (R_i, E_i)$ of the already built substructure. The graph is composed of sets of nodes and edges in the Voronoi region. For simplicity of exposition, we assume truss elements of two sizes: the unit-box size, and the unit box diagonal. The extension to multiple sizes is trivial. We design the density function according to a grid. The unit length of the grid is the length of the truss element. Vertices of the grid have density values equal to the number of truss elements at the vertex. The density of the intermediate points in the space is interpolated. The interpolated value is used in the coverage implementation only. We can generalize this cost function to be a continuous function that encodes the geometry of the object. The demanding mass is defined uniquely for each component type. As for a truss element, the demanding mass $\Delta M^t_{V_i}$ is computed as:

$$\Delta M^t_{V_i} = \int_{V_i} \phi_t(\mathbf{q}) d\mathbf{q} - \int_{V_i} \rho(\mathbf{q}) d\mathbf{q}, \tag{16}$$

where $\rho(\mathbf{q})$ is the density function of the built structure, which increases as a robot assembles truss elements. Note $\phi_t(\mathbf{q})$ of the target shape is fixed. Therefore, a bigger demanding mass means that more elements should be included in that area. The demanding mass for connectors $\Delta M^c_{V_i}$ is the number of required connectors Φ^c for the current structure G_i. Note that $\Delta M^c_{V_i}$ is a function of $\phi(\mathbf{q})$. The demanding masses drive a delivering robot according to gradients as in (Section 4.2). If a structure is composed of other components, we can define the demanding mass for each material.

Algorithm 2 shows the details of the state machine. When construction starts, an assembling robot initializes the parameters R, E, ρ, Φ^c and changes its state to WAITING. Once a new truss element is delivered, the robot finds the optimal place to add it to the structure using Algorithm 3. Since we want the structure to gradually grow, the optimal edge is chosen among a set of edges E_1 that are connected to G. Let E_2 be a set of edges that have maximum demanding mass in E_1. The demanding mass of an edge can be computed as the sum of masses of two nodes defining the edge. Each node of the edges in E_2 should have a density value greater

Algorithm 2. Control Algorithm of assembling robots

```
STATE: IDLE
$R = \varnothing, E = \varnothing$
$\rho(\mathbf{q}) = 0, \Phi^c = \varnothing$
state=WAITING
STATE: WAITING
if truss delivered then
    $e$=findOptimalEdge($R, E, \phi_t, \rho$) (Alg. 3)
    if $e \neq \varnothing$ then
        $\mathbf{t} = \mathbf{q}_{(node_1(e)+node_2(e))/2}$
        state=MOVING
    else
        state=IDLE
    end if
end if
if connector delivered then
    $v \leftarrow \Phi^c$
    $\mathbf{t} = \mathbf{q}_v$
    state=MOVING
end if
STATE: MOVING
if reached t then
    state=ASSEMBLING
else
    move to t
end if
STATE: ASSEMBLING
assemble the material
if the material = truss then
    update $\rho(e)$
    if $node_2 \in R$ and $node_i \notin \Phi^c$ then
        $\Phi^c \leftarrow node_i$
    end if
    $E \leftarrow e$
    $R \leftarrow \{node_1(e), node_2(e)\}$
end if
if the material = connector then
    $\Phi^c = \Phi^c - \{v\}$
end if
state=WAITING
```

than the threshold preventing the robot from assembling the component outside the target structure. In order to achieve a spreading-out structure, priority is given to unconnected edges. If no such edge exists, we choose another seed edge that is not connected to G and has the maximum demanding mass. This jump is required in case that the robot covers substructures which are not connected to each other. If the delivered material is a connector, the optimal location is a node $v \in \Phi^c$ that is connected to the largest number of edges in E. The state machine sets a target location $\mathfrak{t}$ according to the optimal location and changes the state to MOVING. In the MOVING state, an assembling robot moves to the target location $\mathfrak{t}$ and changes the state to ASSEMBLING when it arrives. Finally, a robot assembles the delivered material and updates the parameters. It adds a node of the optimal edge to Φ^c if the node $\notin \Phi^c$ and is connected to other edges. If the material is a connector, the robot removes the node from Φ^c. The state switches to WAITING again.

4.2 Delivery Algorithm

delivering robots operate by a state machine as shown in Figure 4. Each robot has the following states:

- IDLE
- ToSOURCE: moving to get a new element
- ToTARGET: moving to a picked point at the target area Q
- ToASSEMBLY: delivering the element to an assembling robot

Algorithm 3. Finding the Optimal Edge to Build

```
$E_1 = \varnothing, E_2 = \varnothing, E_3 = \varnothing$
if $E_1 = \varnothing$ then
    $e_{opt} = \text{argmax}_e(\phi_t(e) - \rho(e)) \cap (\phi_t(e) > \Delta\phi_{threshold})$
else
    $E_1 \leftarrow e, (e \notin E, node(e) \in R)$
    $E_2 \leftarrow \text{argmax}_{e \in E_1}(\phi_t(e) - \rho(e)) \cap (\phi_t(e) > \Delta\phi_{threshold})$
    if $E_2 = \varnothing$ then
        $e_{opt} = \text{argmax}_e(\phi_t(e) - \rho(e)) \cap (\phi_t(e) > \Delta\phi_{threshold})$
    else
        $E_3 \leftarrow e, (e \in E_2, \{node_1(e), node_2(e)\} \in \{R_i, R_{j,j \in \mathcal{N}_i}\})$
        if $E_3 \neq E_2$ then
            $e_{opt}$= random($E_2 - E_3$)
        else
            $e_{opt}$= random($E_2$)
        end if
    end if
end if
return $e_{opt}$
```

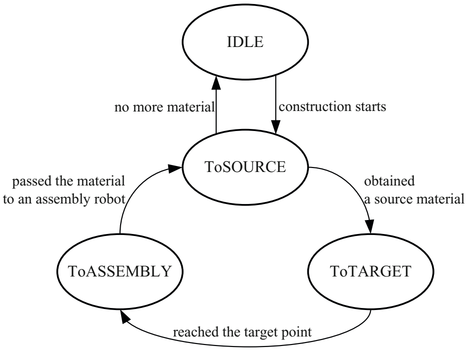


Fig. 4 The state machine for a delivering robot. A delivering robot repeatedly passes source components from the source location to an assembling robot. The initialization of construction causes the delivering robots to start moving. The robots finish working when there is no more source material left at the source location or the assembly is complete.

Algorithm 4 describes the details of the state machine.[3] Given an initially empty state, a delivering robot changes its state to ToSOURCE and moves to $\mathcal{S}$ (the source location). At $\mathcal{S}$, the robot picks a source component if one exists. Otherwise, it stops working. The state is switched to ToTARGET and the robot moves to a randomly chosen point in Q following the probability density function ϕ_t. Therefore, materials are more likely to be delivered to an area with a denser ϕ_t. After arrival at the chosen point, the robot changes the state to ToASSEMBLY and moves following the gradient of the demanding mass ΔM_{V_i} of assembling robots. Delivery by the gradient of the demanding mass yields a *locally* balanced mass distribution. Note that the global balance is maintained by the randomly chosen delivery with density ϕ_t. When the robot meets the assembling robot with the maximum demanding mass, it checks if the state of the assembling robot is WAITING and passes the material. The state changes to ToSOURCE and the robot repeats delivery.

5 Adaptation

We briefly discuss the adaptive features of the construction algorithm. Proofs, details and implementation will be in future work.

Theorem 2. *Continuous coverage during construction compensates for failure of robots*

In the proposed framework for robotic construction, a failure of an assembling robot is critical since the robot covers a unique region. Control that uses equal-mass partitioning continuously during the construction makes the remaining robots automatically compensate for the failed assembling robot. The assembling robots reconstruct the Voronoi regions when the surrounding network of the robots has changed (in implementation, assembling robots keep contact with the neighbor robots.) The

[3] The assembly and the delivery algorithms provably guarantee completion of the correct target structure. In the interest of space, the proof is omitted. Empirical results in Section 6 shows correctness of the algorithms since all the simulations with different initial conditions end up with the same final structure.

Algorithm 4. Control Algorithm of delivering robots

```
STATE: IDLE
state = ToSOURCE
$\mathbf{t} = \mathcal{S}$
STATE: ToSOURCE
if reached $\mathbf{t}$ then
    if source material remains then
        pick a material element
        $\mathbf{t} = \mathbf{q}, \mathbf{q} \sim \phi_t(\mathbf{q})$
        state = ToTARGET
    else
        state = IDLE
    end if
else
    move to $\mathbf{t}$
end if
STATE: ToTARGET
if reached $\mathbf{t}$ then
    state=ToASSEMBLY
else
    move to $\mathbf{t}$
end if
STATE: ToASSEMBLY
communicate with robot $r_i$ s.t. $\mathbf{q} \in V_i$
$deliveryID = \text{argmax}_{(k=i,j\in\mathcal{N}_i)} \Delta M_{V_k}$
$\mathbf{t} = p_{deliveryID}$
if reached $\mathbf{t}$ & state of $r_i$ = WAITING then
    pass the material
    state = ToSOURCE
    $\mathbf{t} = \mathcal{S}$
else
    move to $\mathbf{t}$
end if
```

assembling robots also need to update the parameters such as the graph of the built structure, the demanding mass, etc. The delivering robots achieve this transparently.

Theorem 3. *The algorithms are adaptive to construction in order.*

The construction algorithm is also adaptive to a time varying density function. This property has a nice side-effect: it enables construction *in order*, with connectivity constraints in 3D. For example, the robots can build a structure from the ground up by revealing only part of ϕ_t that is connected to the current structure.

Theorem 4. *The proposed algorithms can be extended for reconfiguration from an existing structure.*

The goal structure might change after or during construction. The construction algorithm can be to adapt the robots to build a new goal structure from the current structure. Equal-mass partitioning can be used with difference of the target density functions, assuming the assembling robot is capable of disassembly. The delivering robots grab source material from the part of the current structure that is unnecessary for the new goal structure.

6 Implementation

Algorithm 2, 4 and the *equal-mass partitioning* were implemented for building 2D and 3D structures. We use side truss elements and connectors that lie at a single source location. We have built several structures using these algorithms.

6.1 Building an A-shaped Bridge

The first simulation demonstrates the construction of a bridge from a single source location of trusses and connectors. The density function ϕ_t and the final Voronoi regions resulting from using the equal-mass partitioning controller for 4,6, and 10 assembling robots are shown in Figure 5. We use a discrete system so that ϕ_t is defined at every node (integer points). The unit length is the length of a truss element. At an arbitrary point $\mathbf{q}$, $\phi_t(\mathbf{q})$ is interpolated from 4 surrounding nodes by barycentric interpolation. The interpolation ensures continuity of ϕ that is required for the cost function $\mathcal{H}$. The robots are deployed from randomly selected starting positions. Figure 5 shows that each robot has approximately the same area of the yellow region. As expected, the masses converge to the same value as shown in Figure 6(b), and the cost function $\mathcal{H}$ approaches zero as in Figure 6(a). A little jitter in the masses and the cost function graphs comes from discrete numerical integrals.

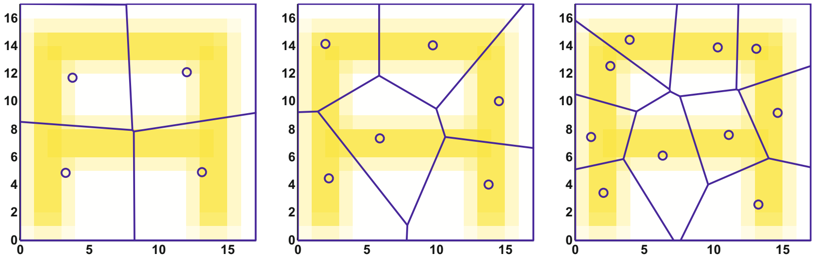


Fig. 5 Density function for an A-shaped bridge and coverage by the equal-mass partitioning. The blue circles are assembling robots. Yellow regions have dense ϕ_t.

Figure 7 shows snapshots from the simulation after partitioning. We use 4 robots for truss delivery and 4 robots for connector delivery. They deliver source materials which have 250 side truss elements and 150 connectors. The area with high density is gradually filled with truss elements and connectors. Because the controller

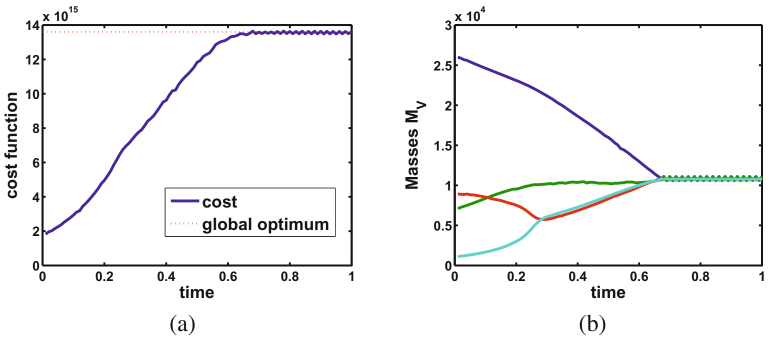


Fig. 6 Result from the equal-mass partitioning controller for 4 assembling robots. (a) Cost function $\mathcal{H}$ (b) Masses of four assembling robots

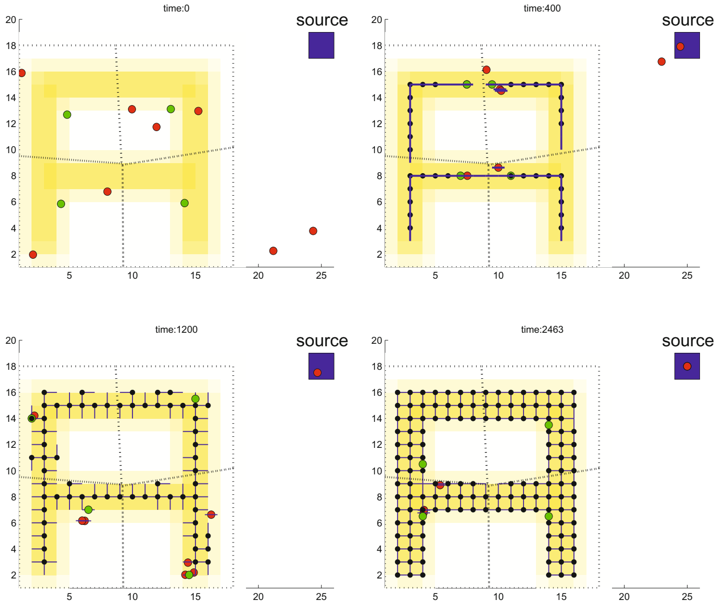


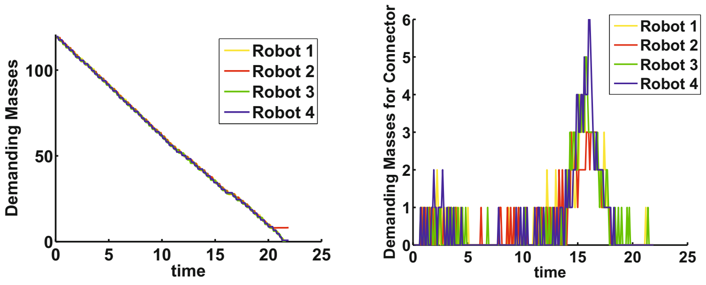
Fig. 7 Snapshots of simulation. Green circles denote assembling robots and red circles denote delivering robots. The blue line is a truss elements and the black dot is a connector. The blue box is the source location. The dotted lines in Q are boundaries of the Voronoi regions.

uses equal mass partitioning and the gradient of the demanding mass, the assembling robots maintain almost the same $\varDelta M_V$ all the time. Therefore, each Voronoi region has a balanced amount of truss elements. Note that the control algorithms do *not* depend on the amount of the source truss elements. With fewer elements, we obtain a thinner structure, while the availability of more truss element yields a denser structure. At the end of the simulation, the assembling robot that has built the least amount of the truss component has assembled 58 truss elements while the robot with the maximum amount has assembled 63. The robot with the minimum number of connectors assembled 33 connectors and the robot with the maximum number assembled 38.

Figure 8 shows the demanding masses for a truss part and a connector. All four curves are completely overlapped, meaning all the substructures have been balanced at all time. The demanding mass for a connector oscillates since it depends on the already built substructure.

6.2 Constructing an Airplane

Figure 9 shows snapshots of building an airplane. 3D grids are used and the target density functions are given and computed in the grids.



(a)

(b)

Fig. 8 (a) Demanding masses for a truss part and (b) a connector. 4 assembling robots and 8 delivering robots are used. The assembly time is set to ten times the velocity. All the graphs are almost overlapped.

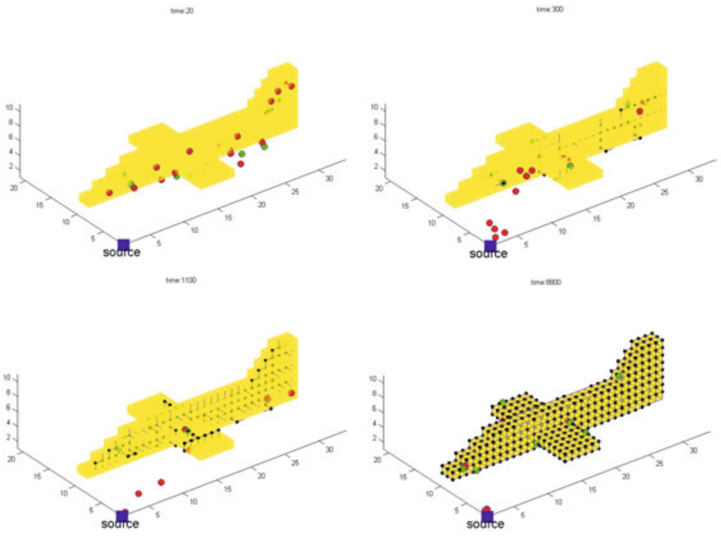


Fig. 9 Snapshots of building an airplane pyramid. There are 10 assembling robots, 20 delivering robots for truss parts and 10 robots for connectors. 1575 truss parts and 656 connectors are assembled.

6.3 Experiment

We have implemented Algorithm 4 using a team of robots. The robots are networked using the Meraki mesh networking infrastructure. The robots (Figure 10) combine an iRobot platform for navigation with a Crust Crawler 4-dof arm and use a Vicon system for location feedback. The setup allowed us to verify the coordination and

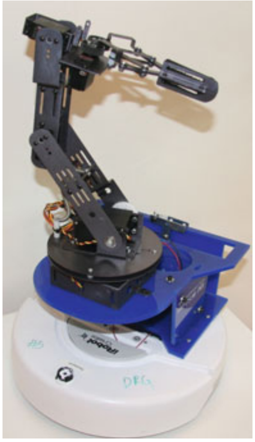


Fig. 10 iRobot platform with Crust Crawler 4-dof arm.

computation required by part delivery. Currently, we have a single type of source component as the screw in Figure 10. Two delivering robots have performed 20 times of delivery to three assembling robots.

7 Conclusion

We propose a framework for distributed robotic construction. Robots with specialized tasks (assembly and delivery of various parts) cover the target structure which is given by a density function, and perform their tasks with only local communication. To divide the structure in equally-sized substructures, the equal-mass partitioning controller is introduced, guaranteeing convergence of the cost function that is the product of the all the masses. An intuitive control criteria with probabilistic deployment and a gradient of the demanding masses is proposed to maintain a balance among the substructures. Implementation with two kinds of source materials (truss and connector) shows that the proposed algorithms assign an equal amount of construction work to the assembling robots, and effectively construct the target structures. This work has opened many interesting questions which we are pursuing as part of our on-going work. We are currently expanding the hardware experiment.

1. **Goal-driven structure.** A target structure can be given as an abstract goal such as connecting two points, not as a density function. In this case, each assembling robot should make the locally best decision of how to build a partial structure.
2. **Connectivity in sub-structure.** Assembling robots may be constrained to work only on the truss structure in practice. In this case, connectivity through each Voronoi region is critical, since a robot may not reach its own region if some part of the region is separated. We need to incorporate this constraint in distributed coverage.

Acknowledgements. This project has been supported in part by The Boeing Company, the U.S. National Science Foundation, NSF grant numbers IIS-0513755, IIS-0426838, CNS-0520305, CNS-0707601, the MAST project, MURI SWARMS project grant number W911NF-05-1-0219, and Emerging Frontiers in Research and Innovation (EFRI) grant #0735953. Seung-kook Yun is supported in part by Samsung Scholarship. We are grateful for this support.

References

1. Detweiler, C., Vona, M., Yoon, Y., Yun, S.-k., Rus, D.: Selfassembling mobile linkages. IEEE Robotics and Automation Magazine 14(4), 45–55 (2007)
2. Cortes, J., Martinez, S., Karatas, T., Bullo, F.: Coverage control for mobile sensing networks, 20th edn., pp. 243–255 (2004)
3. Yun, S.k., Hjelle, D.A., Lipson, H., Rus, D.: Planning the reconfiguration of grounded truss structures with truss climbing robots that carry truss elements. In: Proc. of IEEE/RSJ IEEE International Conference on Robotics and Automation, Kobe, Japan (May 2009)
4. S., Yun, S.k., Rus, D.: Optimal distributed planning for self assembly of modular manipulators. In: Proc. of IEEE/RSJ IEEE International Conference on Intelligent Robots and Systems, Nice, France, September 2008, pp. 1346–1352 (2008)
5. Yun, S.k., Rus, D.: Self assembly of modular manipulators with active and passive modules. In: Proc. of IEEE/RSJ IEEE International Conference on Robotics and Automation, pp. 1477-1482 (May 2008)
6. Nechyba, M.C., Xu, Y.: Human-robot cooperation in space: SM^2 for new spacestation structure. IEEE Robotics & Automation Magazine 2(4), 4–11 (1995)
7. Pavone, M., Frazzoli, E., Bullo, F.: Distributed algorithms for equitable partitioning policies: Theory and applications. In: IEEE Conference on Decision and Control, Cancun, Mexico (December 2008)
8. Pimenta, L.C.A., Schwager, M., Lindsey, Q., Kumar, V., Rus, D., Mesquita, R.C., Pereira, G.A.S.: Simultaneous coverage and tracking (scat) of moving targets with robot networks. In: Proceedings of the Eighth International Workshop on the Algorithmic Foundations of Robotics (WAFR), Guanajuato, Mexico (December 2008)
9. Schwager, M., Rus, D., Slotine, J.-J.E.: Decentralized, adaptive control for coverage with networked robots. International Journal of Robotics Research 28(3), 357–375 (2009)
10. Skaff, S., Staritz, P., Whittaker, W.L.: Skyworker: Robotics for space assembly, inspection and maintenance. In: Space Studies Institute Conference (2001)
11. Werfel, J., Nagpal, R.: International journal of robotics research. Three-dimensional construction with mobile robots and modular blocks 3-4(27), 463–479 (2008)

Synthesis of Controllers to Create, Maintain, and Reconfigure Robot Formations with Communication Constraints[*]

Nora Ayanian[**], Vijay Kumar, and Daniel Koditschek

Abstract. We address the synthesis of controllers for groups of multi-robot systems that enable them to create desired labelled formations and maintain those formations while travelling through an environment with obstacles, with constraints on communication. We assume that individuals in a group are capable of close coordination via high bandwidth communication, but coordination across groups must be limited because communication links are either sporadic or more expensive. We describe a method for developing feedback controllers that is entirely automatic, and provably correct by construction. We provide a framework with which navigation of multiple groups in environments with obstacles is possible. Our framework enables scaling to many groups of robots. While our paper mainly addresses groups of planar robots in $\mathbb{R}^2$, the basic ideas are extensible to $\mathbb{R}^3$.

1 Introduction

There are many types of tasks which require multiple groups working on subtasks concurrently. For example, building an automobile requires many subassemblies which are built in parallel for efficiency. When building a house, each wall of the wooden frame is built separately, then fastened together. These kinds of tasks require coordination of groups on different levels: we want each group to work closely to accomplish subtasks, but we also want them to coordinate on the inter-group level to ensure that they accomplish the higher-level task.

Nora Ayanian · Vijay Kumar · Daniel Koditschek
GRASP Laboratory, Philadelphia, Pennsylvania
e-mail: {nayanian,kumar,kod}@grasp.upenn.edu

[*] We gratefully acknowledge the support of NSF grants IIS-0413138 and IIS-0427313, ARO grant W911NF-05-1-0219, ONR grants N00014-07-1-0829 and N00014-08-1-0696.

[**] N. Ayanian gratefully acknowledges support from the NSF.

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 625–642.
springerlink.com

These types of tasks often occur in cluttered environments, where it may be costly (either in time, energy, or other currency) to allow groups to communicate large amounts of information to other groups. Exchanging information with unnecessary detail results in loss of time and energy, thereby increasing cost. It is imperative to exchange information between groups on a limited basis, while ensuring that ultimately the task is accomplished. We address the problem of synthesizing controllers for labelled robots working in multiple groups in the same workspace, merging and splitting to form a group of desired size and formation shape for a specific task. We will assume that individuals in a group are capable of close coordination via high bandwidth communication but coordination across groups has to be limited because communication links are either sporadic or more expensive.

The configuration space for an N robot system is the Cartesian product of each robot's configuration space, $\mathscr{C} = \mathscr{C}_1 \times \mathscr{C}_2 \times \cdots \times \mathscr{C}_N$. Planning in a space of such large dimension causes computational as well as combinatorial problems which are challenging to address. To decrease complexity, constraints on desired inter-robot distances or relative positions are typically added, reducing dimensionality. Instead, we propose an approach which allows more flexibility. Our goal is to construct controllers similar to navigation functions, which have desirable properties such as safety and convergence guarantees [15], while allowing for larger group sizes.

1.1 Related Work

Similar problems have been studied in the mobile robotics literature both in large and small groups of robots. In small sized groups we can provide guarantees and formal proofs that specific formations can be achieved and maintained, even in the presence of obstacles [2,5,14,4]. As the groups grow, however, formal proofs [4] or controller synthesis [2,15] become prohibitively complex.

In large groups, one cannot feasibly synthesize a specialized controller for each robot, thus achieving specific, labelled formations is not addressed. Some works use abstractions to control an entire group [3,12,19]; in this case, navigation and obstacle avoidance is at the abstraction level, decreasing computation significantly. However, we forfeit control over the network topology, which can change as the group moves. In [3], safety is not guaranteed: robots can collide and escape from the abstraction. Some limitations of [3] are addressed in [12], which still does not enable us to specify formations in the sense of exact shape and topology. A particular formation can be sepcified in [19], but the number of moments which must be supplied to specify a particular formation increases with the number of robots, and the method is not entirely automatic.

Flocking or schooling strategies also enable control of large groups of robots with relatively little computation [18]. These strategies stabilize the entire group's velocity to a single velocity. However, like the above large scale controllers, they lack the capability of specifying particular formations; the final shape of the formation depends on the initial conditions, and cannot be controlled directly.

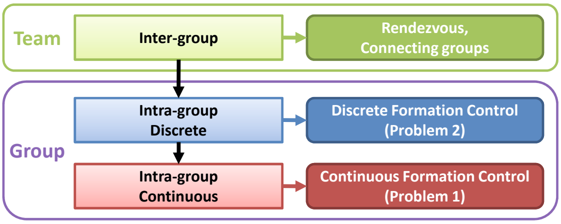

Team
Inter-group
Rendezvous, Connecting groups
Group
Intra-group Discrete
Discrete Formation Control (Problem 2)
Intra-group Continuous
Continuous Formation Control (Problem 1)

Fig. 1 The levels of hierarchy in our controller. We address group navigation at the team level, decoupling it from the formation problem. This allows us to limit the complexity of the problem.

In [7], large groups of robots are stabilized to shapes. However, the presence of obstacles in the workspace could cause local minima or deadlock. Additionally, there is no ordering to the robots on the shape; this is a function of initial conditions.

With sizes between small and large groups, one can provide some guarantees while taking advantage of some reduction in complexity. In [13], proofs are provided for creating and maintaining formations, but collision avoidance is guaranteed only in most cases, requiring careful parameter choices. In [10], undesirable local minima can occur if sufficient virtual leaders are not added. Additionally, the authors do not provide a stable way to switch between formations. In [17] a method is proposed for creating a formation and maintaining it during motion. However, no guarantees are made in the presence of obstacles, and formations must be unlabelled.

1.2 Proposed Approach

The method we propose is for large but finite-sized *groups* with labels; it provides global guarantees on shapes, communication topology, and relative positions of individual robots. We define a *group* of agents as a collection of agents which work simultaneously to complete a single task. Two or more groups act in a *team* to complete a task which requires completing multiple parallel subtasks [1]. Our contribution is twofold. First, we provide a framework for synthesizing controllers for multiple groups of robots in environments with obstacles with communication constraints. Second, we provide a method for *automatically* reconfiguring groups of robots into desired labelled formation shapes without local minima.

A key feature in our approach is a hierarchical decomposition of the problem of constructing feedback controllers (Fig. 1). This reduces the complexity of synthesizing individual robot controllers that meet such specifications as collision avoidance and shapes of formations. We design multi-group navigation controllers at the team level, and the control input for individual robots is derived by summing the inputs from the feedback controller for the robot within its group and the feedback controller for the group. The multi-group navigation problem is equivalent to the multi-robot navigation problem, so we focus in this paper on merging groups of robots into groups of arbitrary numbers of robots, and constructing desired formation shapes.

Our approach is as follows. First, the groups which are involved in the merge convene at some negotiated rendezvous area. We abstract the groups by enclosing them in deformable rectangles, centered at the group centroid and sized appropriately (we address size in Section 4). When the groups are within a pre-specified merging distance, they merge into one group and begin reconfiguration, ending at the desired formation shape. In the case that there are more robots than required for the task assignment, the extra robots split into a separate group. Once the desired formation is achieved, the newly-formed group(s) navigates toward the task.

In Section 2 we formulate the problem. In Section 3 we develop controllers for reconfiguration and formation maintenance. In Section 4 we describe and develop controllers on the abstraction. In Section 5 we describe the process of merging and splitting groups, and follow in Section 6 with MATLAB simulation results. We discuss complexity in Section 7, and conclude in Section 8.

2 Problem Formulation

Consider a team with multiple groups, $\mathscr{G}^i, i = \{1,\dots,m\}$, of N^i kinematic agents $V_A^i = \{a_j^i | j = 1,\cdots,N^i\}$. A group consists of a small number of robots, which can communicate with each other at high bandwidth, enabling centralization. While communication is facilitated by having a complete graph, it is not necessary since agents can exchange information about their neighbors. The team must form a group of $N^g \leq \sum_{i=1}^m N^i$ agents to accomplish a large task. Each agent has the configuration or state $x_j^i \in \mathbb{R}^2$ with the dynamics:

$$\dot{x}_j^i = U_j^i,\ x_j^i \in X_j^i \subset \mathbb{R}^2,\ j = 1,\dots,N^i. \tag{1}$$

Within groups, communication of state information occurs very frequently, so that control is centralized over the entire group. Communication across teams occurs less frequently. Long-range communication occurs rarely if at all, requiring decentralized control.

We use an abstraction on groups of robots to reduce the computational complexity of the problem.

Definition 1. An *abstraction* of a group of robots is a surjection

$$S : \mathscr{C}_T^i \to B,\ S(x^i) = b \tag{2}$$

so that the dimension of B is not dependent on the number of robots N^i.

The abstraction models the extent of the formation, which we call the *boundary*, while the controllers on the robot level ensure that the boundary is satisfied. Therefore, a group of robots can reconfigure from one formation to another knowing only the limits of the abstraction, decoupling the agents from the physical space. The specific abstraction we use is discussed in detail in Section 4.

Figure 2 is a graphical representation of the hierarchichal structure of our approach. At the top level, groups interact with limited knowledge of other groups. At the middle level, there is interaction between individual robots in order to maintain

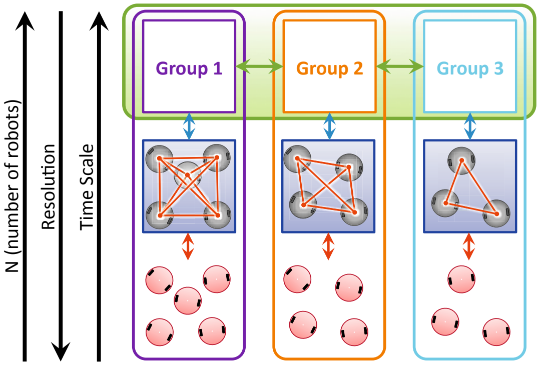


Fig. 2 Hierarchical structure. At the top level, groups interact with limited knowledge about other groups. Within each group, formations must be maintained. At the lowest level, individual robots implement the continuous controllers. As the number of robots increases, the spatial resolution required for planning and control decreases, and the time scale increases so that dynamics get faster.

the formation. At the lowest level, individual robots execute the continuous controller. In the examples we present, we assume that there are no obstacles within the group boundary. However, should an obstacle appear within a group boundary, the group can split appropriately, then rejoin in a location without obstacles.

The input to each agent is the sum

$$U_j^i = u_j^i + u_b^i \tag{3}$$

where u_j^i is the formation shape controller (Section 3), and $u_b^i \in \mathbb{R}^2$ is the abstraction controller (Section 4). As shown in Fig. 2, the two control inputs drive dynamics at two different time scales. Group dynamics (motion within a group) must evolve on a much faster time scale than the team dynamics (motion of the group). In other words, time required for robots to converge to a target formation within the group is much smaller than the motion of the group. This time-scale separation is necessary to guarantee convergence at all levels.

The number of agents required for the task, the goal formation shape, and the environment are known to all agents. We assume each agent is capable of synthesizing controllers (both for group navigation and reconfiguration), its group's abstraction, and whether certain criteria are satisfied (such merging criteria). This information propagates through the group rapidly through explicit communication, therefore we are not concerned with which agent is responsible for these calculations. Agents observe relative state of their neighbors and exchange this information with other neighbors to construct a complete group configuration. Groups are capable of long-range communication in short bursts to determine a rendezvous point. Once the rendezvous point is determined, they no longer use long-range communication.

3 Formation Shape Controllers

In this section we develop the formation shape controllers, which both reconfigure the robots and maintain the formation once it is achieved.

Let the set of all agents be $V_A \equiv V_A^1 \cup V_A^2 \cup \cdots \cup V_A^m$. (Hereafter, for simplicity, where we describe a property for $\cup_{i=1}^{m} \cup_{j=1}^{N^i} a_j^i$, we will drop the superscript i.) Connectivity between all agents V_A is modeled by a *robot formation graph*. Agents must maintain *proximity constraints*, which are represented by edges on the robot formation graph and the collision graph. Recall that a *graph* is a pair of sets $G=(V,E)$, where $V=\{v_1,...,v_n\}$ is the set of vertices or nodes and $E \subseteq [V]^2$ is the set of edges on the graph. Pairs of vertices for which $(v_i,v_j) \in E$ are called adjacent. A graph in which all pairs of vertices are adjacent is called a complete graph.

Definition 2. A *robot formation graph* is a graph $G_N^{\rho} = (V_A, E_N)$ where E_N is the set of edges which denote pairs of agents which directly communicate state information, and ρ is a metric for determining inter-agent distances. To enable communication, pairs $(a_j,a_k) \in E_N$ must be within a maximum distance $|x_j - x_k|_{\rho} \leq \delta_{max}$. The constraint can be written

$$\nu^{\rho}(x_j,x_k) \leq 0 \ \ \forall (x_j,x_j) \in E_N. \tag{4}$$

We call pairs of agents which are adjacent on this graph *neighbors*.

Definition 3. The *collision graph* on a group $\mathscr{G}_i$ of agents is a static graph $G_L^{i,\rho} = (V_A^i, E_L^i)$ where E_L^i is the set of all pairs of agents in V_A^i which cannot occupy the same coordinates simultaneously. Pairs $(a_j^i, a_k^i) \in E_L^i$ must maintain a nonzero minimum distance $|x_j^i - x_k^i|_{\rho} \geq \delta_{min}$. The constraint can be written

$$\lambda^{i,\rho}(x_j^i,x_k^i) \geq 0 \ \ \forall (x_j^i,x_k^i) \in E_L^i. \tag{5}$$

For homogeneous agents occupying the same space, this graph will be complete.

The proximity constraints specified by the robot formation graph $G_N^{\rho} = (V_A, E_N)$ and the collision graph $G_L^{i,\rho} = (V_A^i, E_L^i)$ are realized using metric ρ. For pairs of agents $(a_j^i, a_k^i) \in E_N \cap E_L^i$ the intersection of these constraints corresponds to an annulus in the relative space of two agents. One can choose any underestimation of the annulus, with a tessellation consisting of convex regions, and call this the metric ρ. A few options are shown in Fig. 3. Our metric of choice is the infinity norm, in Fig. 3e since it has fewer regions, decreasing complexity and allowing more flexibility in the formation. Hence, the proximity constraints become a square annulus in the relative space of two agents as in Fig. 4a (the shaded region denotes illegal configurations). Pairs $(a_j^i, a_k^i) \in E_N - E_L^i$ have only the maximum distance constraint (Fig. 4b), and pairs $(a_i, a_j) \in E_L - E_N$ have infinite annuli (Fig. 4c).

To accomplish the task, a specific formation shape is required of the new group. The formation shape can be provided in exact, continuous form, or approximate, discrete form. In defining the discrete robot formation shape, we desire a non-overlapping partition of the square annulus whose union is the entire region. The description must also contain information about connectivity.

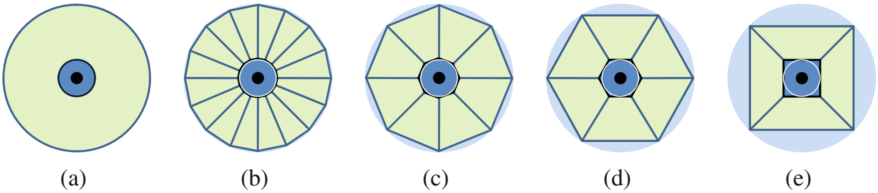


Fig. 3 The annulus in relative space of a pair of robots (a) and a few approximations (b)-(e). The dark blue center corresponds to the collision constraint, while the light green area depicts allowable configurations. Note that in panels (b)-(e) the dark blue polygons are overapproximations of the unsafe region (encircled in white), while the large green polygons underapproximate the safe region (shaded blue circle).

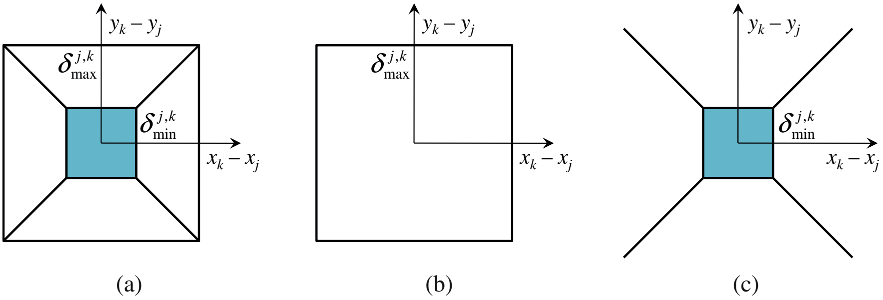


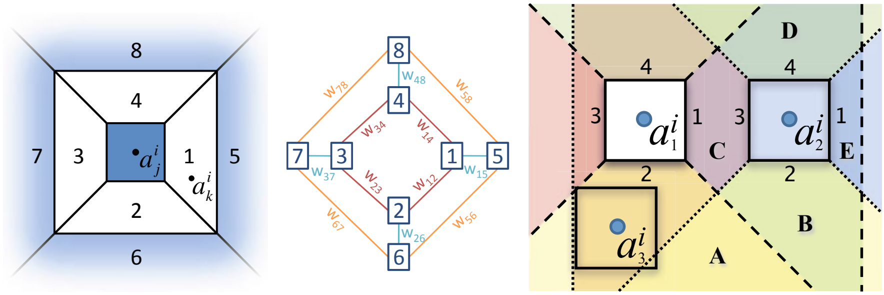
Fig. 4 Proximity constraints for pairs of robots. Shaded area indicates illegal configurations. (a) Neighbors with collision constraints. (b) Neighbors with no collision constraint. (c) Collision constraint on non-neighbors.

Definition 4. A *robot formation shape* $\mathscr{F}$ of N robots describes the relative locations of the set of robots, specified exactly using continuous shape variables. $\mathscr{F}$ is a $\binom{N}{2}$-tuple of vectors $\mathscr{F} = \{r_{(1,2)}, r_{(1,3)}, \dots, r_{(N-1,N)}\}$ where $r_{(j,k)} = x_k - x_j$. We use a superscript ($\mathscr{F}^i$) to refer to the subset corresponding to group $\mathscr{G}^i$.

Definition 5. The *discrete robot formation shape* $\mathscr{F}_d$ of N robots describes approximate relative locations of the set of robots using discrete shape descriptors. The shape descriptors are integers corresponding to regions of the tessellation of the annulus. $\mathscr{F}_d$ is a $\binom{N}{2}$-tuple of these integers, $\mathscr{F}_d = \{f_{(1,2)}, f_{(1,3)}, \dots, f_{(N-1,N)}\}$. For each pair of robots (a_j, a_k) $j < k$, $f_{(j,k)}$ describes the position of a_k with respect to a_j, according to Fig. 5a. Regions 1-4 correspond to communication between the pair (i.e. $(a_j, a_k) \in E_N$). Regions 5-8 correspond to no direct communication between the pair ($(a_j, a_k) \notin E_N$). We use ($\mathscr{F}^i_d$) to refer to the subset corresponding to group $\mathscr{G}^i$.

An example discrete formation shape for a group of three robots is shown in Fig. 5c. Here, $f^i_{(1,2)} = 1$, $f^i_{(1,3)} = 2$, and $f^i_{(2,3)} = 3$, so that $\mathscr{F}^i_d = (1,2,3)$.

To build the space on which we develop the controllers, we combine the proximity constraints with the configuration spaces of the robots.



(a) (b) (c)

Fig. 5 The shape descriptors (regions) for discrete robot formation shapes. (a) For pair (a^i_j, a^i_k) the shape descriptor represents the location of a^i_k with respect to a^i_j. Here, the region is 1. (b) We can use this discrete system to build an adjacency graph. (c) Overlapping proximity regions of a group of three robots. Dashed (dotted) lines correspond to boundaries of proximity regions for a^i_1 (a^i_2). Letters A-E correspond to possible polytopes through which a^i_3 would pass to get to $\mathscr{F}^i_d = (1,1,1)$.

3.1 The Task Configuration Space

Definition 6. The *configuration space* $\mathscr{C}^i_j$ of an agent a^i_j is the set of all transformations of a^i_j. The *free space* $\mathscr{C}^{i,free}_j$ of a^i_j is the set of all transformations of a^i_j which do not intersect with obstacles in the configuration space.

In this work, navigation (and therefore obstacles), are dealt with on the abstraction level. The abstraction constrains the free space of a group so that $\mathscr{C}^i_j \neq \mathscr{C}^{i,free}_j$. In fact, $\mathscr{C}^{i,free}_j$ is a local space, whose origin is the center of the abstraction.

Definition 7. The *group configuration space* is the Cartesian product of the free spaces of each agent in a group,

$$\begin{aligned} \mathscr{C}^i_{all} &= \mathscr{C}^{i,free}_1 \times \mathscr{C}^{i,free}_2 \times \cdots \times \mathscr{C}^{i,free}_{N^i} \\ x^i &= [x^i_1, \ldots, x^i_{N^i}] \in \mathscr{C}^i_{all}. \end{aligned} \tag{6}$$

Thus the configuration of a group $\mathscr{G}^i$ of N^i agents is described by a single point in $\mathscr{C}^i_{all} \subset \mathbb{R}^d$, $d = 2N^i$.

Definition 8. The *task configuration space* $\mathscr{C}^i_T$ for the group $\mathscr{G}^i$ is the set

$$\begin{aligned} \mathscr{C}^i_T &= \mathscr{C}^i_{all} \cap \mathscr{L}^i_\rho \cap \mathscr{N}_\rho, \\ \mathscr{L}^i_\rho &\equiv \{x^i | x^i \in \mathscr{C}^i_{all}, \lambda_\rho(x^i_j, x^i_k) \geq 0 \; \forall (a^i_j, a^i_k) \in E^i_L\}, \\ \mathscr{N}_\rho &\equiv \{x^i | x^i \in \mathscr{C}^i_{all}, \nu_\rho(x^i_j, x^i_k) \leq 0 \; \forall (a^i_j, a^i_k) \in E_N\}. \end{aligned} \tag{7}$$

$\mathscr{C}^i_T$ is a polytopic space in which robots cannot collide or lose communication.

Note that each group's task configuration space is independent of other groups; that is, robots in a group rely only on each other's positions. Thus, reconfigurations of a group from one formation to another is entirely decoupled from other groups.

Each discrete group formation corresponds to a unique polytope in $\mathscr{C}_T^i$. By planning and synthesizing controllers on $\mathscr{C}_T^i$, we drive the robots to the desired formation and keep them there as the group navigates the space.

Problem 1. For any initial group formation $\mathscr{F}_0^i$, consider the system (1) on $\mathbb{R}^{2N^i}$, with goal formation $\mathscr{F}_g^i$ and metric ρ. Find an input function $u^i : [0,T_0] \to \mathscr{U}$ for any $x_0^i \in \mathscr{F}_0^i \subset \mathscr{C}_T^i$ such that

1. for all time $t \in [0,T_0]$, $x^i \in \mathscr{C}_T^i$ and $\mathscr{F}_{T_0}^i = \mathscr{F}_g^i$,
2. $\dot{x}_j^i = u_j^i$,
3. $x^i(t) \in \mathscr{L}_\rho^i \cap \mathscr{N}_\rho$, $\forall t \in [0,T_0]$.

3.2 Shape Controllers on $\mathscr{C}_T^i$

In this section, we synthesize feedback controllers to solve Problem 1. We follow closely [2], specifying our modifications for the present problem. We choose to use a centralized version, since state information is shared among all connected agents. Unlike [2], we simultaneously build a discrete representation of the task configuration space and find a path in this representation. We then translate the path into feedback controllers. The key step in the first stage is to define an adjacency graph on the set of polytopes.

Definition 9. The *polytope graph* $G_P^i = (V_P^i, E_P^i)$ on the polytopes in $\mathscr{C}_T^i$ is the pair of sets $V_P^i = \{c_1^i, \dots, c_n^i\}$, where c_q^i is the Chebyshev[1] center of the q-th polytope P_q^i, and E_P^i, the set of all pairs of polytopes which share a facet.

By using the discrete group formation shape notation, we build the polytope graph online, on an as-needed basis. For example, in the three robot group formation $\mathscr{F}_d^i = (1,2,3)$ in Fig. 5c, we use the graph in Fig. 5b to generate adjacent formations by moving to adjacent regions. By changing the region corresponding to the location of a_2^i with respect to a_1^i, we get $\mathscr{F}_d^i = (2,2,3)$ and $\mathscr{F}_d^i = (4,2,3)$. By changing the integer corresponding to the location of a_3^i with respect to a_1^i, we get $\mathscr{F}_d^i = (1,1,3)$ or $\mathscr{F}_d^i = (1,3,3)$. Using this method, we build an adjacency graph online while concurrently minimizing cost via a graph search algorithm.

In Fig. 5b we have included separate weights w_{pq} for each edge. Thus the cost of moving from one formation[2] $\mathscr{F}_d^r$ to another adjacent formation $\mathscr{F}_d^{r+1}$ is

$$C\left(\mathscr{F}_d^r, \mathscr{F}_d^{r+1}\right) = \sum_{j=1}^{N} \sum_{\substack{k=1 \\ k>j}}^{N} w_{f_{(j,k)}^r f_{(j,k)}^{r+1}}. \tag{8}$$

[1] The Chebyshev center of a polytope is the center of the largest inscribed ball.

[2] Although this applies to a group $\mathscr{G}^i$, here we drop the subscript i for clarity.

This induces a heuristic cost for going from the initial to the goal formation. If $\mathscr{F}_d^0$ and $\mathscr{F}_d^g$ are the initial and final formations, the minimum cost of reconfiguring is

$$h\left(\mathscr{F}_d^0, \mathscr{F}_d^g\right) = \sum_{j=1}^{N^i} \sum_{\substack{k=1 \\ k>j}}^{N^i} w_{f^0_{(j,k)} f^g_{(j,k)}}.$$

In our simulations, we choose to minimize the number of transitions, and thus set all weights $w_{pq} = 1$.

It is important to note that not all nodes will be valid. From Fig. 5c it is obvious that $\mathscr{F}_d^i = (4,2,3)$ is an invalid formation: it is impossible for a_2^i to be in region 4 of a_1^i and simultaneously have a_3^i in region 3 of a_2^i. Therefore, while we anticipate nodes by using the graph in Fig. 5b, before adding nodes to G_P^i we check if the polytopes are empty, corresponding to invalid formations. Thus the cost heuristic can be used as an underestimate for the cost-to-goal in a best-first-search algorithm.

Problem 2. For the initial discrete group formation shape $\mathscr{F}_d^{i,0}$, find a path $t = \{c^i_{t^1}, \cdots, c^i_{t^g}\}$ on G_P^i to the goal discrete formation shape $\mathscr{F}_d^{i,g}$, such that we minimize the actual cost

$$h^*\left(\mathscr{F}_d^0, \mathscr{F}_d^g\right) = \sum_{r=0}^{g-1} C\left(\mathscr{F}_d^{i,r}, \mathscr{F}_d^{i,r+1}\right).$$

Theorem 1 (Necessary and Sufficient condition). *Problem 1 has a solution iff Problem 2 has a solution.*

Proof. $\mathscr{C}_T^i$ contains every allowable configuration x^i in our polytopic world model. G_P^i contains all the information about the connectivity of $\mathscr{C}_T^i$. Thus, if there is a solution to Problem 1, there must exist a path from the start node in G_P^i to the goal node. Conversely, if there is no path on the graph G_P^i between the start node and the goal node, there is no solution to Problem 1. □

Corollary 1 (Completeness). *Problem 2, and therefore Problem 1, has a solution if the start and goal nodes on the polytope graph G_p^i are connected.*

After a path to the goal is determined, we want to be able to synthesize feedback controllers to drive the system through those polytopes to the goal. We choose to use a controller developed by Lindemann and Lavalle which is applicable in high dimensions and results in smooth feedback [11].

We set the vector field on each facet except the exit facet to be a unit inward normal; on the exit facet, we set the vector field to the unit normal pointing outward. By using a bump function, vector fields on the faces of each transitional polytope are smoothly blended with an attractor field on the Generalized Voronoi Diagram (GVD) of the polytope. For each polytope on the path to the goal formation, we implement the controller using the Chebyshev center of the exit facet as the attractor.

In the goal polytope, if an exact formation shape is prescribed, we decompose the polytope so that the vertices of each polytope in the decomposition are the vertices of a facet along with the goal point. Then we set all facet vector fields to be pointing inward, and the field on the faces of the decomposition always pointing to the goal.

If a discrete formation shape is prescribed, we set the attractor field to always point to the Chebyshev center of the polytope (not the exit facet), satisfying the requirements set in [11] to guarantee convergence. There are several advantages to using the Chebyshev center. First, it allows a minimum radius around the attractor, providing some robustness (for disturbances or feedback linearization for nonholonomic robots). Second, the Chebyshev center lies on the GVD, so we do not need to use a separate decomposition that would force us to enumerate the vertices of the polytope, which is computationally expensive.

To smoothly stabilize the system at the Chebyshev center, we normalize the attractor field until it is within the radius of the Chebyshev ball. Once within the Chebyshev ball, we use a second bump function to drive the input to zero as we approach the center.

We use the controller for the goal polytope to maintain the desired group formation of the group as it moves through the space, inside the bounds of the abstraction.

4 Geometric Abstractions for Groups

The abstraction enables robots to operate in an obstacle-free (but limited) environment, while group navigation is handled by a higher-level controller for the abstraction. This controller is then summed onto the individual controllers as in (3). .

Definition 10. The *group abstraction* B^i is a pair

$$B^i = \left(x_b^i, s^i(N^i)\right) \in \mathbb{R}^4 \tag{9}$$

where $s^i(N^i)$ is a shape vector, and

$$x_b^i = \frac{1}{N^i} \sum_{j=1}^{N^i} x_j^i.$$

The center of the group abstraction is x_b^i, and the shape vector $s^i(N^i)$ represents the boundary and size of the abstraction which encloses the group of robots.

Although any boundary can be chosen for the group, we choose a rectangle for all groups boundaries since a rectangle corresponds well with our choice of the infinity norm. Therefore, the shape vector is a pair $s^i(N^i) = (s_w^i, s_h^i)$ where s_w^i corresponds to the width and s_h^i to the height of the rectangle. Knowing the shape of the abstraction, we can determine the minimum size of the abstraction so that the rectangle is large enough to contain the number of robots in the group,

$$N^i \leq \left(\lfloor s_w^i(N^i)/\delta_{\min}\rfloor + 1\right)\left(\lfloor s_h^i(N^i)/\delta_{\min}\rfloor + 1\right). \tag{10}$$

To ensure graph completeness in the abstraction, we can also enforce $\{s_w^i(N^i), s_h^i(N^i)\} \leq \delta_{\max}$. This induces a limit for the number of robots which can be in a group with a complete graph using our abstraction

$$N^{\max} = \left(\lfloor \delta_{\max}/\delta_{\min}\rfloor + 1\right)^2. \tag{11}$$

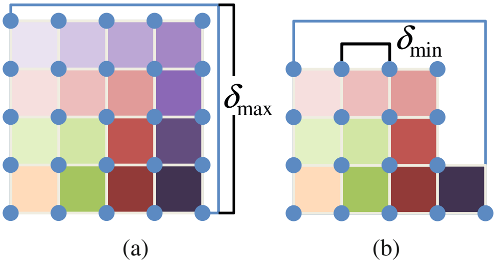


Fig. 6 The limits of the size of the abstraction. (a) The upper bound for number of robots in a group is a function of the ratio $(\delta_{\max}/\delta_{\min})$. (b) The minimum size of the square is a function of $\delta_{\min}$ and N^i.

An example of the maximum number of robots in a group is shown in Fig. 6a. Here, $\delta_{\max}/\delta_{\min} = 4$, so that $N^{\max} = (4+1)^2 = 25$ is the maximum number of robots in the group. An example of the minimum of $s^i(N^i)$ is in Fig. 6b. Here, $\lfloor\sqrt{N^i}\rfloor = 4$, so the minimum width is $4\delta_{\min}$.

We treat the abstraction as a single robot. Because multiple abstractions share one workspace, we need a multi-robot controller to ensure they do not collide.

To emulate the cost of sharing information across large spaces, we place restrictions on communication between groups of robots on three levels based on inter-group distances. The lowest level of communication occurs at the largest distances, above the threshold $\gamma^{\max}$,

$$|x_b^i - x_b^k|_\rho > \gamma^{\max}.$$

At this level, groups communicate as necessary to negotiate rendezvous points.

The mid-level of inter-group communication occurs below the threshold $\gamma^{\max}$,

$$|x_b^i - x_b^k|_\rho \le \gamma^{\max},$$

where groups perceive position relative to each other $(x_b^i - x_b^k)$.

Once two groups are close enough that explicit inter-group communication is established (i.e. a member from one group is able to communicate directly with a member of the other group), groups can share both the number and position of each group's agents $(x_j^i, j = 1, \ldots, N^i)$ by passing this information through the robot formation graph. Once the group is within a pre-specified distance of the other groups,

$$|x_b^i - x_b^k|_\rho \le \gamma^{\min},\ \forall\, i,k \in \{1,\cdots,m\} \tag{12}$$

the groups are able to commence the merging process.

As discussed in Section 1.1, there are many controllers applicable to multi-robot problems. However, the inter-group communication restrictions as well as the real-time nature of this problem require a decentralized controller to bring the groups to the rendezvous area. (In our simulations, we have used path-planning to determine a path for each group to the rendezvous point.) Once the groups are close enough to know relative position ($x_b^i - x_b^k \leq \gamma^{\max}$), a more demanding controller may be used to drive them close enough to communicate and merge (until they satisfy (12)). Then, they must reconfigure into the desired formation, and continue to the task location while maintaining that formation.

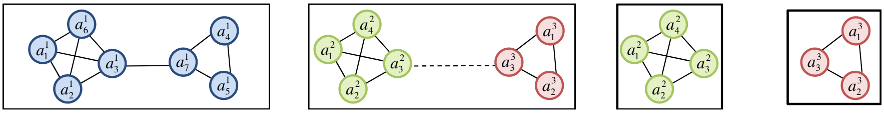
5 Merging and Splitting Groups

In this section we describe the process of merging groups. Once the groups have satisfied (12), they combine their boundaries into the smallest boundary of the specified shape that contains all of the groups' boxes. This will now be the boundary for the reconfiguration discussed in Sec. 3.2.

The desired formation can be either connected or disconnected. If we have just the right number of robots, the resulting graph is connected. If we have too many, the formation graph is disconnected, and a group of robots break away.

If we have the exact number of robots required for the task, once they have reconfigured into the desired formation, the boundary size must be adjusted. The boundary is resized to within some small ε of the smallest rectangle centered at the centroid of the group and enclosing all the robots. If it is possible to resize directly to the desired size (in the case of a rectangle, $s_w^i(N^i) \times s_h^i(N^i)$), then we are done, and the group can continue to the task. If not, we allow the robots to stabilize to the Chebyshev center of the formation using the new boundary size. Then, we re-evaluate the boundary size, and iterate until we get to the desired size.

If we have more robots than required, the group reconfigures into a formation shape such that the required robots' subgraph is that required for the task, and the rest of the robots are connected to that subgraph by one edge. An example of this is shown in Fig. 7. The dotted line in the example depicts where the group will split. The desired formation is achieved after reconfiguration in Fig. 7a. The discrete formation shape integer relating a_3^1 to a_7^1 is $f_{3,7}^1 = 1$. In Fig. 7b, the groups separate, until $f_{3,7}^1 = 5$, meaning there is no direct communication between a_3^1 and a_7^1. This is when the group splits into two as in Fig. 7c.



(a) Desired formation shape (b) Preparing to split (c) After splitting

Fig. 7 An example of a desired robot formation shape and the splitting process when the task requires less robots than the total number in all groups.

In summary, the algorithm for our approach involves the following six steps.

Algorithm 1

1. *Construct the goal controller for formation maintenance in $\mathscr{C}_T^i$ for each group of robots.*
2. *Drive the groups toward each other in the space.*
3. *When (12) is satisfied, solve Problem 2 while selectively constructing the task configuration space for the joint group of robots.*
4. *Solve Problem 1 on all polytopes on the path, and solve for a goal controller in the goal polytope.*
5. *If $\sum_{i=1}^{m} N^i > N^g$, break the team into two separate groups and construct the task configuration space and goal polytope controller for the new groups.*
6. *Drive the newly formed group(s) to the task location while using the new goal polytope controller to maintain the formation.*

6 Simulations

We simulate a two-group example where the groups merge into the correct number of robots required for the task, and a three-group example where there are more robots than necessary to complete the task. The simulations run on MATLAB, using the Multi-Parametric Toolbox for polytope computations [9].

Although our simulations run on MATLAB, by using a MATLAB interface for PLAYER and GAZEBO, we can transition this controller directly to a three-dimesional dynamic simulator (GAZEBO) or perform experiments on robots using PLAYER, parts of the PLAYER/STAGE/GAZEBO project [6]. As in [2], we can use feedback linearization to provide controllers for nonholonomic robots.

6.1 Two Groups Merging and Continuing to Task Location

The goal in this example (Fig. 8) is to join groups of four and three robots into a single group for a task which requires seven robots. Once the groups are merged and in the desired boundary shape and size, they proceed to the task location. We used as parameters $\delta_{\max} = 2.5$, $\delta_{\min} = 0.2$, $\gamma^{\max} = 5$, $\gamma^{\min} = 0.2$, $\varepsilon = 0.1$, and $s_h^i = s_w^i = 2.5 - 2/N^i$.

6.2 Three Groups Merging and Splitting

In this example (Fig. 9), two tasks require seven and two robots each. Nine robots are available across three groups of three. The three groups merge, then split into two groups of seven and two robots. The groups now proceed to their respective task locations. Here we used the same parameters as above (except $\gamma^{\min} = 0.5$).

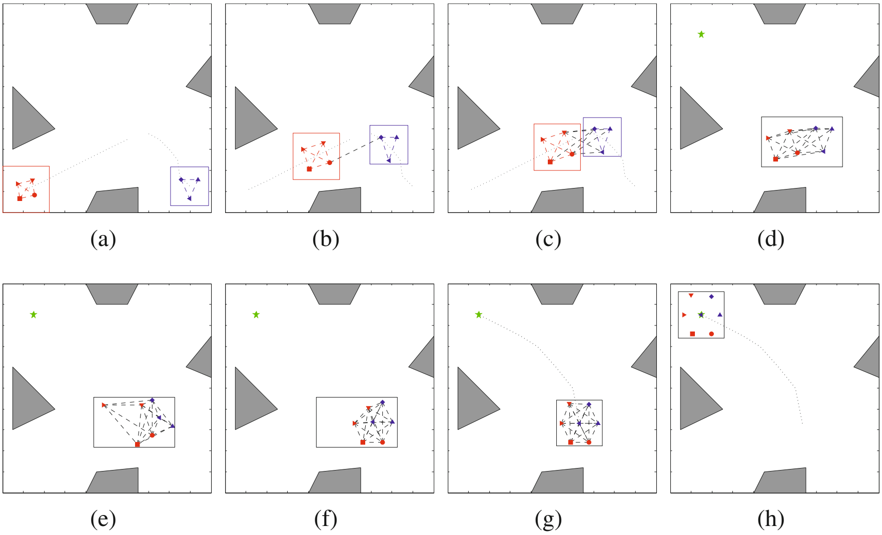


Fig. 8 A two group simulation. Dashed lines represent communication links (omitted in (h)), dotted lines represent the path, and the star represents the task location. (a) Initial condition. (b) Direct inter-group communication established. (c) Merging criterion satisfied. (d) A single group is formed. (e) Mid-reconfiguration. (f) At the desired formation. (g) The box is reshaped. (h) At the task location.

7 Complexity

In this section we discuss the complexity of our method. For each pair of agents with collision constraints we have one annulus with 8 regions, resulting in a maximum

$$P_{max} = 8^{N^i(N^i-1)/2}$$

polytopes in $\mathscr{C}_T^i$. Although this scales exponentially with the number of robots in the group, we only construct the polytopes as we expand nodes in the polytope graph.

To solve Problem 2, we use an A^* algorithm. In an A^* algorithm, the number of nodes expanded is exponential in the actual path length, unless the error of the heuristic grows no faster than the logarithm of the actual cost [16]

$$\left| h\left(\mathscr{F}_d^0, \mathscr{F}_d^g\right) - h^*\left(\mathscr{F}_d^0, \mathscr{F}_d^g\right) \right| \leq O(\log h^*(n)).$$

Although we do not have a bound for our heuristic error, empirically we have found that there exists a path to the goal of the heuristic cost. If there exists a path to the goal, then there likely exist other paths of the same length to the goal (though in the case of differently weighted transitions, equal length may not correspond to equal cost). For example, if the start formation is $\mathscr{F}_d^0 = \{1,2,3\}$ and the goal formation $\mathscr{F}_d^g = \{1,1,1\}$, $h(\mathscr{F}_d^0, \mathscr{F}_d^g) = 3$, there exist three paths (ABE, CBE, CDE in Fig. 5c), with cost $h^*(\mathscr{F}_d^0, \mathscr{F}_d^g) = 3$. In general, since the graph is cyclic, it is likely that a path

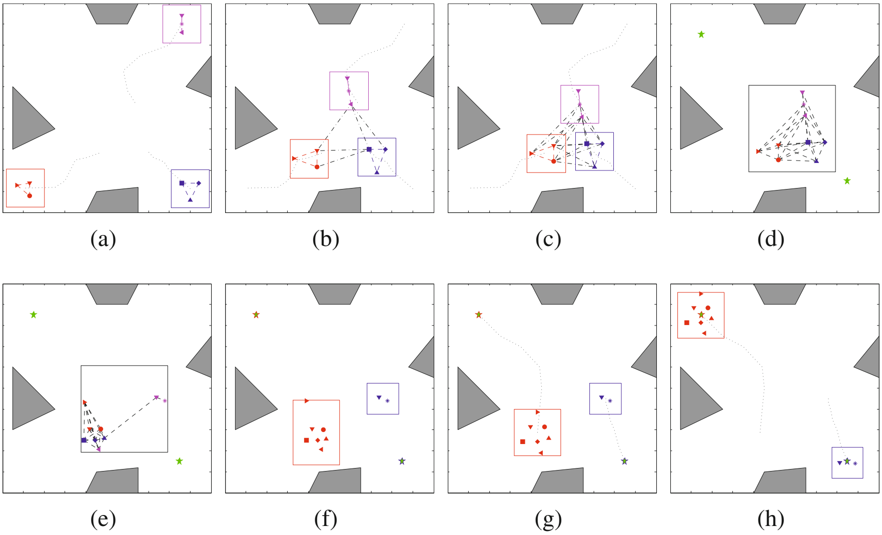


Fig. 9 A three group simulation. Dashed lines represent communication links (omitted (g)-(j) where graphs are complete), dotted lines represent the path, and stars represent task locations. (a) Initial conditions. (b) Inter-group direct communication established. (c) Merging criterion satisfied. (d) One group is formed. (e) Prior to disconnection. (f) Groups split. (g) Boxes are resized. (h) At the task locations.

exists with the exact cost of the heuristic. (If the graph is weighted such that each edge is not of equal cost, this is not generally the case).

We construct a controller in each polytope in $t = \{c^i_{t1}, \cdots, c^i_{tg}\}$ to solve Problem 1. The vector field can be computed in $O(\sum_{d=0}^{2N^i} \Phi_d)$ time, where Φ_d is the total number of d-dimensional cells in the GVD [11]. For bounds on Φ_d for a certain class of polytopes, see [8].

The complexity increases linearly in the number of concurrent merging and reconfiguring processes, since each group computes its own controllers.

8 Conclusion

We have presented a method for controlling multiple groups of robots to create, reconfigure, and maintain formations under communication constraints. We provide guarantees of safety, preventing inter-robot collisions and collisions with obstacles in the workspace. Our controller is entirely automatic, and requires information about the space, the desired formation, and the task location. We have discussed briefly the complexity of our approach, which in the worst case scales exponentially in N^i and $h^* \left(\mathscr{F}^0_d, \mathscr{F}^g_d\right)$.

The algorithm is complete based on the choice of abstraction boundary. Since the abstraction is an overestimate of the area occupied by the robots, it is possible that some solutions will be lost. This is especially the case when fixing the boundary of the abstraction while the group is navigating through the space. It should be possible to enforce a minimum area of the space inside the boundary, and maintain discrete formations under some deformations. Further study is required for the case where the abstraction can be reshaped to allow for navigation through cluttered spaces.

Finally, our approach lends itself to planning and control for heterogeneous teams. This is an issue of future research.

References

1. Anderson, C., Franks, N.R.: Teams in animal societies. Behav. Ecol. 12(5), 534–540 (2001)
2. Ayanian, N., Kumar, V.: Decentralized feedback controllers for multi-agent teams in environments with obstacles. In: Proc. IEEE Int. Conf. Robot. Autom, Pasadena, CA, pp. 1936–1941 (May 2008)
3. Belta, C., Kumar, V.: Abstraction and control for groups of robots. IEEE Trans. Rob. 20(5), 865–875 (2004)
4. Desai, J.P., Ostrowski, J.P., Kumar, V.: Modeling and control of formations of nonholonomic mobile robots. IEEE Trans. Rob. Autom. 17(6), 905–908 (2001)
5. Egerstedt, M., Hu, X.: Formation constrained multi-agent control. IEEE Trans. Rob. Autom. 17(6), 947–951 (2001)
6. Gerkey, B., Vaughan, R.T., Howard, A.: The player/stage project: Tools for multi-robot and distributed sensor systems. In: Proc. of Int. Conf. on Advanced Robotics (June 2003)
7. Hsieh, M.A., Loizou, S., Kumar, V.: Stabilization of multiple robots on stable orbits via local sensing. In: Proc. IEEE Int. Conf. Rob. Automat., pp. 2312–2317 (April 2007)
8. Klee, V.: On the complexity of d-dimensional Voronoi diagrams. Archiv der Mathematik 34(1), 75–80 (1980)
9. Kvasnica, M., Grieder, P., Baoti, M.: Multi-parametric toolbox (mpt) (2004), http://control.ee.ethz.ch/~mpt/
10. Leonard, N.E., Fiorelli, E.: Virtual leaders, artificial potentials and coordinated control of groups. In: Proc. IEEE Conf. Decis. Contr., vol. 3, pp. 2968–2973 (2001)
11. Lindemann, S.R., LaValle, S.M.: Smoothly blending vector fields for global robot navigation. In: Proc. IEEE Conf. Decis. Contr., Seville, Spain (2005)
12. Michael, N., Kumar, V.: Controlling shapes of ensembles of robots of finite size with nonholonomic constraints. In: Proceedings of Robotics: Science and Systems, pp. 157–162 (June 2008)
13. Ogren, P., Egerstedt, M., Hu, X.: A control lyapunov function approach to multi-agent coordination. In: Proc. IEEE Conf. Decis. Contr., vol. 2, pp. 1150–1155 (2001)
14. Olfati-Saber, R., Murray, R.M.: Distributed cooperative control of multiple vehicle formations using structural potential functions. In: Proc. of IFAC World Congress, Barcelona (July 2002)

15. Rimon, E., Koditschek, D.E.: Exact robot navigation using artificial potential functions. IEEE Trans. Robot. Autom. 8(5) (October 1992)
16. Russel, S.J., Norvig, P.: Artificial Intelligence: a Modern Approach, 1st edn. Prentice Hall, Englewood Cliffs (1995)
17. Smith, B.S., Wang, J., Egerstedt, M.B.: Persistent formation control of multi-robot networks. In: Proceedings of the IEEE Conference on Decision and Control, pp. 471–476 (December 2008)
18. Tanner, H.G., Jadbabaie, A., Pappas, G.J.: Flocking in fixed and switching networks. IEEE Trans. Automat. Contr. 52(5), 863–868 (2007)
19. Yang, P., Freeman, R.A., Lynch, K.M.: Multi-agent coordination by decentralized estimation and control. IEEE Trans. Automat. Contr. 53(11), 2480–2496 (2008)

Planning and Control for Cooperative Manipulation and Transportation with Aerial Robots

Jonathan Fink, Nathan Michael, Soonkyum Kim, and Vijay Kumar

Abstract. We consider the problem of controlling multiple robots manipulating and transporting a payload in three dimensions via cables. Individual robot control laws and motion plans enable the control of the payload (position and orientation) along a desired trajectory. We address the fact that robot configurations may admit multiple payload equilibrium solutions by developing constraints for the robot configuration that guarantee the existence of a unique payload pose. Further, we formulate individual robot control laws that enforce these constraints and enable the design of non-trivial payload motion plans. Finally, we propose two quality measures for motion plan design that minimize individual robot motion and maximize payload stability along the trajectory. The methods proposed in the work are evaluated on a team of aerial robots in experimentation.

1 Introduction

Aerial transport of payloads by towed cables is common in emergency response, industrial, and military applications for object transport to environments inaccessible by other means. Examples of aerial towing range from emergency rescue missions where individuals are lifted from dangerous situations to the delivery of heavy equipment to the top of a tall building. Typically, aerial towing is accomplished via a single cable attached to a payload. However, only limited controllability of the payload is achievable with a single attachment point [9].

In this work we consider the problem of cooperative aerial manipulation. Unlike aerial towing, which addresses the issue of controlling a point-model payload, aerial manipulation considers the control of a payload with six degrees of freedom. The cooperative aerial manipulation problem can be shown to be analogous to

Jonathan Fink · Nathan Michael · Soonkyum Kim · Vijay Kumar
GRASP Laboratory, University of Pennsylvania, Philadelphia, PA
e-mail: {jonfink,nmichael,soonkyum,kumar}@grasp.upenn.edu

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 643–659.
springerlink.com

cable-actuated parallel manipulators in three dimensions, where in the former the payload pose is affected by robot positions and in the latter pose control is accomplished by varying the lengths of multiple cable attachments. The analysis of the mechanics of cable-driven parallel manipulators is a special case of the analysis of the general problem of payloads suspended by n cables in three dimensions. The $n = 6$ case is addressed in the literature on cable-actuated payloads, where the payload pose is fully specified. When $n = 5$, if the line vectors are linearly independent and the cables are taut, the line vectors and the gravity wrench axis must belong to the same linear complex [5]. The payload is free to instantaneously twist about the reciprocal screw axis. When $n = 4$, under similar assumptions on linear independence and positive tension, the line vectors and the gravity wrench must belong to the same linear congruence. The unconstrained freedoms correspond (instantaneously) to a set of twists whose axis lie on a cylindroid. The $n = 3$ case admits solutions where all three cables and the gravity wrench axis lie on the same regulus - the generators of a hyperboloid which is a ruled surface [10].

Of note are the unilateral constraints imposed by cables that can only admit positive forces. As the tension on a cable can only be positive and the distance between the two end-points of a cable cannot be more than its free length, and further the tension is non-zero only when the distance is equal to the free lengths – each cable introduces a *complementarity constraint* of the type:

$$s \geq 0, \quad \lambda \geq 0, \quad \lambda s = 0 \tag{1}$$

where s is the slack in the cable and λ is the tension in the cable. The solutions to systems of linear equations subject to such constraints, the so-called *Linear Complementarity Problem* (LCP), have been studied extensively [3]. Even though all the terms in the LCP are linear in the unknowns, the system can have multiple solutions or no solutions at all.

In prior work, we formulated the mechanics of aerial manipulation (reviewed in Sect. 2) [7]. We focused this development for an under-actuated system with three robots and presented pose control of a payload to a sequence of desired poses. Our prior work did not address a significant challenge resulting from the under-actuation which is of great relevance when considering the problem of planning non-trivial manipulation tasks: the fact that for a given robot configuration, the physical system may admit multiple payload equilibrium solutions.

In this paper, we build upon our prior work to develop individual robot control laws and motion plans that permit the control of the payload along a desired trajectory. We address the fact that robot configurations may admit multiple payload equilibrium solutions by developing constraints for the robot configuration that guarantee the existence of a unique payload pose. We formulate individual robot control laws that enforce these constraints and enable the design of non-trivial payload motion plans.

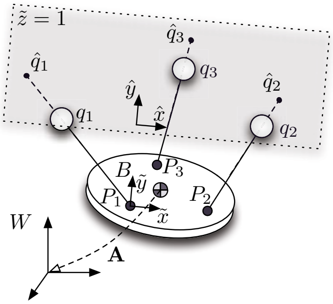


Fig. 1 A team of three point-model robots manipulate a payload in three dimensions. The coordinates of the robots in the inertial frame W are $q_i = [x_i, y_i, z_i]$ and in the body-fixed frame (attached to the payload) B are $\tilde{q}_i = [\tilde{x}_i, \tilde{y}_i, \tilde{z}_i]$. The rigid body transformation from the payload body frame aligned at the center of mass to W is $\mathbf{A} \in SE(3)$. Additionally, we denote the projection of the robot position $\tilde{q}_i$ along $q_i - p_i$ to the plane $\tilde{z} = 1$ as $\hat{q}_i = [\hat{x}_i, \hat{y}_i, 1]$.

2 Mechanics of Cooperative Manipulation

2.1 Model

We begin by considering the problem with three robots in three dimensions carrying an object (although the analysis readily extends to n robots). We consider point-model robots for the mathematical formulation and algorithmic development although the experimental implementation requires us to consider the full twelve-dimensional state-space of each quadrotor and a formal approach to realizing these point-model abstractions, which we provide in Sect. 6.1. Thus, our configuration space is given by $\mathscr{Q} = \mathbb{R}^3 \times \mathbb{R}^3 \times \mathbb{R}^3$. Each robot is modeled by $q_i \in \mathbb{R}^3$ with coordinates $q_i = [x_i, y_i, z_i]^{\mathrm{T}}$ in an inertial frame, W (Fig. 1). Define the robot configuration as $q = [q_1, q_2, q_3]^{\mathrm{T}}$. The i^{th} robot cable with length l_i is connected to the payload at the point P_i with coordinates $p_i = \left[x_i^p, y_i^p, z_i^p\right]^{\mathrm{T}}$ in W. Let $p = [p_1, p_2, p_3]^{\mathrm{T}}$ denote all attachment points. We require P_1, P_2, and P_3 to be non-collinear and span the center of mass. The payload has mass m with the center of mass at C with position vector $r = [x_C, y_C, z_C]^{\mathrm{T}}$. We denote the fixed Euclidean distance between attachment points P_i and P_j as $r_{i,j}$. The payload's pose $\mathbf{A} \in SE(3)$ can be locally parameterized using the components of the vector r and the Euler angles with six coordinates: $[x_C, y_C, z_C, \alpha, \beta, \gamma]^{\mathrm{T}}$. The homogeneous transformation matrix describing the pose of the payload is given by:

$$\mathbf{A} = \begin{bmatrix} \mathbf{R}(\alpha, \beta, \gamma) & \begin{pmatrix} x_C \\ y_C \\ z_C \end{pmatrix} \\ 0 & 1 \end{bmatrix}. \tag{2}$$

Note that $\mathbf{R}$ is the rotation matrix going from the object frame B to the world frame W (as depicted in Fig. 1). Additionally, for this work we follow the *Tait-Bryan* Euler angle parameterization for $\{\alpha, \beta, \gamma\}$.

The equations of static equilibrium can be written as follows. The cables exert zero-pitch wrenches on the payload which take the following form after normalization:

$$\mathbf{w}_i = \frac{1}{l_i} \begin{bmatrix} q_i - p_i \\ p_i \times (q_i - p_i) \end{bmatrix}.$$

The gravity wrench takes the form:

$$\mathbf{g} = -mg \begin{bmatrix} e_3 \\ r \times e_3 \end{bmatrix},$$

where g is the acceleration due to gravity and $e_3 = [0,0,1]^{\mathrm{T}}$. For static equilibrium:

$$\mathbf{W}\,\lambda = \begin{bmatrix} \mathbf{w}_1 \ \mathbf{w}_2 \ \mathbf{w}_3 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{bmatrix} = -\mathbf{g} \tag{3}$$

where $\lambda_i \geq 0$ is the tension in the i^{th} cable.

In order for (3) to be satisfied (with or without non-zero tensions), the four line vectors or zero pitch wrenches, $\mathbf{w}_1$, $\mathbf{w}_2$, $\mathbf{w}_3$, and $\mathbf{g}$ must belong to the same *regulus*. The lines of a regulus are points on a 2-plane in $\mathbb{PR}^5$ [11], which implies that the body is under constrained and has three degrees of freedom. Instantaneously, these degrees of freedom correspond to twists in the *reciprocal screw system* that are reciprocal to $\mathbf{w}_1$, $\mathbf{w}_2$, and $\mathbf{w}_3$. They include zero pitch twists (pure rotations) that lie along the axes of the *complementary regulus* (the set of lines each intersecting all of the lines in the original regulus). Geometrically, (3) simply requires the gravity wrench to be reciprocal to the reciprocal screw system, a fact that will be exploited in our calculations in the next section.

We will make the following simplifying assumptions:

1. The payload is a homogeneous, planar object and the center of mass lies in the plane of the attachment points.
2. The robots are positioned initially on one side of the plane of the attachment points.
3. The mass of the object is sufficiently small that three robots are able to lift the object.
4. The system is quasi-static and we may neglect the transients associated with the payload converging to an equilibrium position.

For the analysis, we will use a local frame, B, attached to the payload that is defined with the origin at P_1, the x axis pointing toward P_2 and the $x-y$ plane coincident with the plane formed by P_1, P_2, and P_3. In this local coordinate system, the components are denoted by $(\tilde{\cdot})$ and given as:

$$P_1 = [0, 0, 0]^{\mathrm{T}}, \qquad P_2 = [\tilde{x}_2^p, 0, 0]^{\mathrm{T}}, \qquad P_3 = [\tilde{x}_3^p, \tilde{y}_3^p, 0]^{\mathrm{T}};$$
$$\tilde{q}_1 = [\tilde{x}_1, \tilde{y}_1, \tilde{z}_1]^{\mathrm{T}}, \qquad \tilde{q}_2 = [\tilde{x}_2, \tilde{y}_2, \tilde{z}_2]^{\mathrm{T}}, \qquad \tilde{q}_3 = [\tilde{x}_3, \tilde{y}_3, \tilde{z}_3]^{\mathrm{T}}.$$

Note that without loss of generality, we assume that $\tilde{x}_2^P > \tilde{x}_3^P$, restricting the possible permutations of P_i.

2.2 Kinematics

In this section, we determine the pose of the payload for given positions of the quadrotors (the direct problem) and the positioning of the quadrotors for a desired pose of the payload (the inverse problem). The direct and inverse problems are analogous to the direct and inverse kinematics for manipulators. However, they are different in that one must incorporate the equations of static equilibrium in order to determine the answer.

2.2.1 The Direct Problem

With hovering robots and cables in tension, we can treat the object as being attached to three stationary points through rigid rods and ball joints to arrive at the constraints

$$\|q_i - p_i\| = l_i, \tag{4}$$

for $i = \{1, 2, 3\}$. In addition, we impose the three equations of static equilibrium, (3), to further constrain the solutions.

Problem 1 (Direct Problem). Given the positions of the robots, q_i, $i = \{1, 2, 3\}$, find the position(s) and orientation(s) of the payload satisfying the kinematics of the robots-cables-payload system (4) and the equations of equilibrium (3).

We solve this problem by finding the three screws (twists) that are reciprocal to the three zero pitch wrenches. Define the 6×3 matrix $\mathbf{S}$ of twists with three linearly independent twists such that the vectors belong to the null space of $\mathbf{W}^{\mathrm{T}}$:

$$\mathbf{W}^{\mathrm{T}}\mathbf{S} = 0.$$

$\mathbf{S}$ is an algebraic function of the positions of the three pivot points, P_i. In order to satisfy (3), p_i must satisfy the three algebraic conditions:

$$\mathbf{S}(x_i^P, y_i^P, z_i^P)^{\mathrm{T}}\mathbf{g} = 0. \tag{5}$$

Finally, we require that there exist a *positive*, 3×1 vector of multipliers, λ, that satisfies (3). We find the multipliers using the Moore-Penrose inverse, $\mathbf{W}^{\dagger}$, of $\mathbf{W}$,

$$\lambda = -\mathbf{W}^{\dagger}\mathbf{g},$$

This allows us to eliminate equilibrium poses corresponding to tensions that violate the non-negativity constraints on the multipliers.

Thus, the direct problem reduces to solving (4, 5). Accordingly, the object has three degrees of freedom. Imposing the three equilibrium conditions in (5) we can, in principle, determine a finite number of solutions for this analog of the direct kinematics problem.

2.2.2 The Inverse Problem

The inverse problem reduces to the problem of solving for the three-dimensional set $Q_c \subset \mathscr{Q}$ by solving for the nine variables $\{(\tilde{x}_i, \tilde{y}_i, \tilde{z}_i), i = 1, 2, 3\}$ subject to (4, 5).

Problem 2 (Inverse Problem). Given the desired payload position and orientation (2), find positions of the robots (q_i) that satisfy the kinematics of the robots-cables-payload system and the equations of equilibrium (3).

We now restrict our attention to a reduced space of possible configurations based upon the assumptions $(1, 2)$. We introduce the notion of normalized components, denoted by $(\hat{\cdot})$, with $\hat{q}_i = [\hat{x}_i, \hat{y}_i, 1]$ to define the position of the i^{th} robot projected to a constant height $\tilde{z}_i = 1$ above the payload. Based on this simplification (we now only need to solve for three planar positions), we redefine the static equilibrium condition as

$$\mathbf{S}(\hat{x}_i, \hat{y}_i, 1)^{\mathrm{T}} \tilde{\mathbf{g}} = 0. \tag{6}$$

Solving the system of equations in (6) yields algebraic solutions for $\{\hat{x}_2, \hat{x}_3, \hat{y}_3\}$ as functions of $\{\hat{x}_1, \hat{y}_1, \hat{y}_2\}$. Note that any solution to (6) is indeed a solution to (5). Further, we may compute the position of the robots $\tilde{q}_i$ given the normalized coordinates $\hat{q}_i$ and the kinematic constraint (4) as

$$\tilde{q}_i = l_i \frac{\hat{q}_i}{\|\hat{q}_i\|} + P_i.$$

By using $\hat{q}_i$ as coordinates, we can obtain closed-form analytic solutions for the positions of all robots that respect the kinematic constraints and the condition for static equilibrium.

3 Solving the Direct Problem

In this section, we consider an optimization-based formulation to determining solutions to the direct problem. We begin by stating the optimization problem that minimizes the potential energy of the rigid payload.

Problem 3 (Optimization-Based Formulation of the Direct Problem)

$$\begin{array}{ll} \min\limits_{p_i} & z_1^p + z_2^p + z_3^p \\ \text{s.t.} & \|q_i - p_i\| \leq l_i, \quad i = \{1, 2, 3\} \\ & \|p_i - p_j\| = r_{i,j}, \quad i, j = \{1, 2, 3\}, i \neq j. \end{array}$$

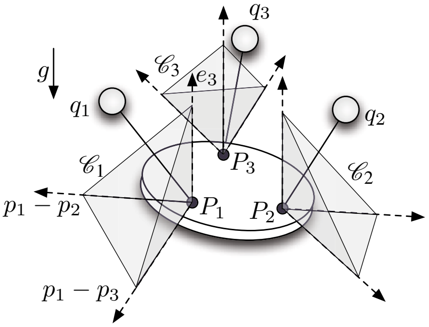


Fig. 2 A graphical depiction of the cone constraints presented in Proposition 1.

We have replaced the equations of static equilibrium for cables in tension, (4), with inequalities (a more accurate restatement of Problem 1). The minimization in Problem 3 explicitly prohibits local maxima while ensuring $\lambda_i \geq 0$, allowing for $\lambda_i = 0$ (the i^{th} cable becoming slack) as modeled in the complementarity constraints (1).

Problem 3 is non-convex due to the payload rigidity, which shows up as the equality constraint $\|p_i - p_j\| = r_{i,j}$. However, by relaxing this condition, we see that the program becomes convex.

Problem 4 (Convex Formulation of the Relaxed Direct Problem)

$$\begin{aligned} \min_{p_i} \quad & z_1^p + z_2^p + z_3^p \\ \text{s.t.} \quad & \|q_i - p_i\| \leq l_i, \quad i = \{1, 2, 3\} \\ & \|p_i - p_j\| \leq r_{i,j}, \quad i, j = \{1, 2, 3\}, i \neq j. \end{aligned}$$

Clearly, this formulation relaxes the rigidity requirement of the payload. However, the problem of solving for the equilibrium pose of the payload is now a Second Order Cone Program (SOCP) [6] and convex.

We now seek conditions on robot positions that enforce the rigidity of the payload while preserving the structure of the SOCP in Problem 4.

Proposition 1 (Conditions for Unique Solutions to the Direct Problem). *The solution to Problem 3 is unique provided that $q_i - p_i$ lies strictly inside the convex cone $\mathscr{C}_i$, defined by the vectors $(p_i - p_j)$, $(p_i - p_k)$, and e_3 ($j \neq i, k \neq i, j \neq k$), where $e_3 = [0, 0, 1]^T$.*

Proof. We begin by considering the SOCP in Problem 4. The First Order Necessary Conditions (FONC) are:

$$\begin{aligned} \mu_1(p_1 - q_1) + \mu_4(p_1 - p_2) + \mu_6(p_1 - p_3) + \frac{1}{2}e_3 = 0 \\ \mu_2(p_2 - q_2) + \mu_4(p_2 - p_1) + \mu_5(p_2 - p_3) + \frac{1}{2}e_3 = 0 \\ \mu_3(p_3 - q_3) + \mu_5(p_3 - p_2) + \mu_6(p_3 - p_1) + \frac{1}{2}e_3 = 0 \end{aligned} \tag{7}$$

with

$$\|p_i - q_i\| \leq l_i, \quad \mu_i \geq 0, \quad \mu_i(\|p_i - q_i\|^2 - l_i^2) = 0 \quad \text{for} \quad i = \{1,2,3\},$$

and

$$\begin{aligned} \|p_1 - p_2\| \leq r_{1,2}, \quad \mu_4 \geq 0, \quad \mu_4(\|p_1 - p_2\|^2 - r_{1,2}^2) = 0 \\ \|p_2 - p_3\| \leq r_{2,3}, \quad \mu_5 \geq 0, \quad \mu_5(\|p_2 - p_3\|^2 - r_{2,3}^2) = 0 \\ \|p_3 - p_1\| \leq r_{3,1}, \quad \mu_6 \geq 0, \quad \mu_6(\|p_3 - p_1\|^2 - r_{3,1}^2) = 0. \end{aligned}$$

Eliminating the condition when the payload is in a vertical orientation, $p_i \times p_j \cdot e_3 \neq 0$, for all $i \neq j$, from (7) it is clear that μ_1, μ_2, and μ_3 are all strictly positive. The variables μ_i correspond to forces in the physical system, with $i = \{1,2,3\}$ representing the cable tensions. If we can show that μ_4, μ_5, and μ_6, the three internal forces between the anchor points of the triangle, are strictly positive, the inequalities $\|p_i - p_j\| \leq r_{i,j}$ satisfy the rigidity (strict equality) requirement of Problem 3.

Consider the FONC for q_1,

$$q_1 - p_1 = \frac{\mu_4}{\mu_1}(p_1 - p_2) + \frac{\mu_6}{\mu_1}(p_1 - p_3) + \frac{1}{2\mu_1}e_3. \tag{8}$$

This equation is the force balance equation at P_1 up to a scalar multiplier. If we require that $q_1 - p_1$ lies *strictly inside* the convex cone $\mathscr{C}_1$, generated by $(p_1 - p_2)$, $(p_1 - p_3)$, and e_3:

$$q_1 - p_1 = \eta_{1,1}(p_1 - p_2) + \eta_{1,2}(p_1 - p_3) + \eta_{1,3}e_3, \tag{9}$$

with $\eta_{1,i} > 0$ for $i = \{1,2,3\}$, the coefficients $\frac{\mu_4}{\mu_1}$, $\frac{\mu_6}{\mu_1}$, and $\frac{1}{2\mu_1}$ must be strictly positive. A similar argument holds for q_2 and q_3 to show that μ_4, μ_5, and μ_6 are all strictly positive under this requirement.

Thus, if $q_i - p_i$ lies *strictly inside* the convex cone $\mathscr{C}_i$, the solution to the SOCP is also a solution to Problem 3. Further, as an SOCP is convex, the solution to Problem 3 is unique with the restriction on robot positions defined by (9). □

The result of Proposition 1 is a set of conditions on individual robot positions with respect to the payload that guarantee a unique solution to the direct problem (as depicted in Fig. 2). We will call the conditions $q_i - p_i \in \mathscr{C}_i$ the *cone constraints*. In the next section we develop individual robot control laws that ensure these constraints are preserved while driving the robot configuration, q, to a desired state.

4 Robot Control Laws

We begin by formulating (9) for the i^{th} robot as

$$q_i = A_i \eta_i + p_i,$$

where

$$\eta_i = \left[\eta_{i,1}, \eta_{i,2}, \eta_{i,3}\right]^{\mathrm{T}}$$

and

$$A_i = \left[p_i - p_j,\ p_i - p_k,\ e_3\right],\ j \neq i, k \neq i, j \neq k.$$

A_i is full rank and invertible, permitting the computation of η_i given the robot position q_i and payload configuration p:

$$\eta_i = A_i^{-1}(q_i - p_i). \tag{10}$$

Note that η_i is unique for any q_i and p up to a rigid body translation and rotation of the payload about e_3. Additionally, for the development that follows we treat the system as quasi-static (the fourth assumption stated in Sect. 2.1).

We wish to design control laws for individual kinematic agents, $u_i = \dot{q}_i$, that drive the full system q and p to a desired time-invariant configuration defined by q^d and p^d while respecting the cone constraints. The easiest way to guarantee convergence to a time-invariant q_i^d (and thus q^d) is to require the error, $q_i^e = (q_i^d - q_i)$, to converge exponentially to zero:

$$\dot{q}_i^{\star} = K_i q_i^e, \tag{11}$$

where K_i is any positive definite 3×3 matrix. From (11), we obtain robot velocities that guarantee globally asymptotic convergence to the desired robot configuration. However, due to the conditions on $\eta_{i,j}$ for the i^{th} robot with $j = \{1, 2, 3\}$, we must also consider the position of each robot with respect to the payload pose p.

We relax the requirement of exponential convergence to a desired robot configuration and replace it with a slightly different notion of convergence in order to accommodate the cone constraints. Specifically instead of insisting on (11), we find the solution closest to the velocities corresponding to the exponential solution which also enforce the cone constraints; and thus guarantee a unique solution to the direct problem.

We require that the error in the robot state decrease *monotonically*:

$$q_i^{e\mathrm{T}} K_i \dot{q}_i \geq 0, \tag{12}$$

treating K_i as a diagonal matrix. Further, from (10), we see that the cone constraints of Proposition 1 are met when $\eta_{i,j} > 0$. Therefore, we introduce active constraints that enforce this requirement when $\eta_{i,j} < \delta_{i,j}$, where $\delta_{i,j}$ is a value selected near the constraint boundary. We will require

$$\dot{\eta}_{i,j} \geq 0 \tag{13}$$

when $\eta_{i,j} < \delta_{i,j}$. Differentiating (10),

$$\dot{\eta}_{i,j} = \dot{A}_i^{-1}(q_i - p_i) + A_i^{-1}(\dot{q}_i - \dot{p}_i).$$

Recalling that $A_i A_i^{-1} = I_{3\times 3}$ and thus $\dot{A}_i A_i^{-1} = -A_i \dot{A}_i^{-1}$, (13) is equivalently written as

$$\dot{q}_i \geq \dot{A}\eta_i + \dot{p}_i. \tag{14}$$

Proposition 2 (Decentralized Robot Control Law). *Equation (15) is a decentralized control law that selects a unique control input that is instantaneously closest to*

(11) while satisfying the monotonic convergence inequality and enforcing the cone constraints.

$$\dot{q}_i = \underset{\dot{\bar{q}}_i}{\arg\min} \|\dot{q}_i^\star - \dot{\bar{q}}_i\|^2, \quad s.t. \quad (12,14) \tag{15}$$

Proof. The constraints in (12,14) provide the monotonic convergence condition and enforcement of the cone constraints. The function being minimized is the discrepancy from the exponentially convergent solution. Since the inequality constraints are linear in $\dot{q}_i$ and the function being minimized is a positive-definite, quadratic function of $\dot{q}_i$, (15) is a convex, quadratic program (QP) with a unique solution. Further, as each robot only relies on its own state and knowledge of the payload pose, it is a decentralized control law. □

Remark 1. The requirement of the cone constraints implies that robots will not collide. Therefore, inter-robot collision avoidance is maintained through (14) as enforced by (15).

Convergence properties of (15) To investigate the global convergence properties, we introduce the Lyapunov function

$$V(q) = \frac{1}{2} q^{e\mathrm{T}} q^e,$$

for the system of robots, q. Recalling from Proposition 1 that the enforcement of $\eta_{i,j} > 0$ implies uniqueness of p, it is sufficient to consider the convergence of q to q^d to show $p = p^d$. Since the solution of (15) must satisfy the inequality (12), we know that $q^{e\mathrm{T}} K \dot{q} \geq 0$. If K is chosen to be a diagonal 9×9 matrix with positive entries, this condition also implies $q^{e\mathrm{T}} \dot{q} \geq 0$. In other words, $\dot{V}(q) = -q^{e\mathrm{T}} \dot{q} \leq 0$. Clearly $V(q) \to \infty$ as $\|q\| \to \infty$. Further, $V(q)$ is globally uniformly asymptotically stable. Therefore, from LaSalle's invariance principle, we know that the robot configuration will converge to the largest invariant set given by $q^{e\mathrm{T}} \dot{q} = 0$. Additionally, we know that $\dot{q} = 0$ only when $\dot{q}_i = 0$ for $i = \{1, 2, 3\}$. Thus the invariant set is characterized by the set of conditions that lead to the system of inequalities given by (12, 14) to have $\dot{q}_i = 0$ as the only solution for $i = \{1, 2, 3\}$.

5 Planning for Aerial Manipulation

In this work, we take a decoupled approach to motion planning for the payload subject to the constraints of aerial manipulation. That is, we first plan a collision-free path in $SE(3)$ for a conservative approximation of the full robots-cables-payload system and then generate coordinated robot trajectories that enact that plan. Here we focus on the task of choosing robot trajectories for a given payload path.

Consider the manipulation workspace $Q_M \subset Q_c$, where Q_c is a function of the payload orientation and is computed in Sect. 2.2.2 to be the set of robot positions such that there is positive tension in each cable and the payload is in equilibrium.

Other factors that further constrain feasible robot configurations are maximum allowable tension ($\lambda_i < \lambda_{max}$) and the cone constraints ($\eta_i > 0$) presented in Proposition 1. While these constraints are easily computable, they are non-smooth in nature and generate a nontrivial workspace for planning.

As the constraints on Q_M do not reduce it to a unique solution for a given payload pose, there are in fact sets of configurations parameterized by $\{\hat{x}_1, \hat{y}_1, \hat{y}_2\}$ that can be chosen to stabilize the payload to a given orientation. The planning problem is: (a) to find feasible robot configurations at each point along the desired payload trajectory and (b) exploit the redundancy in configurations to optimize specified secondary design goals or quality measures. Examples of optimizing quality measures include minimizing the distance traveled by individual robots or maximizing the natural frequency of the robots-cables-payload system to lead to faster attenuation of disturbances.

5.1 Algorithm and Quality Measure

The motion planning problem we are considering is formalized as follows. Given a payload path specified by a series of waypoints $\{\mathbf{A}^0, \mathbf{A}^1, \ldots, \mathbf{A}^n\}$ where $\mathbf{A}^k \in SE(3)$, find a series of robot poses $\{q^0, q^1, \ldots, q^n\}$ where $q^k = \{q_1^k, q_2^k, q_3^k\} \in Q_M(\mathbf{A})$ such that the payload under the quasi-static assumption achieves the desired pose, $\mathbf{A}^k$. Since there are generally a set of feasible configurations q for a given payload pose $\mathbf{A}$ but no analytic representation of the set $Q_m(\mathbf{A})$, we use a sampling-based approach to evaluate candidate configurations. Selection of $\{\hat{x}_1, \hat{y}_1, \hat{y}_2\}$ yields a point in $Q_C(\mathbf{A})$ which is mapped to q and verified with respect to $\eta_i > 0$ and $\lambda_i < \lambda_{max}$ to be in $Q_M(\mathbf{A})$. Configurations are chosen to minimize a metric $D(q^{k+1}, q^k)$.

Since we have no formal completeness guarantees for this algorithm, we make the following assumption. The set of feasible payload configurations $\mathbf{A}_{feasible}$ such that $Q_M(\mathbf{A}_{feasible}) \neq \emptyset$ is defined by bounds on the roll and pitch angles of the payload and $Q_M(\mathbf{A}_{feasible})$ is sufficiently large that an initial feasible sample is always found. The space $Q_M(\mathbf{A})$ is explored with uniform sampling.

As with any sampling-based planning method, attention must be paid to the edges *connecting* states. In this work, we naively consider states to be connected by straight lines in Euclidean space and necessary deviations from this plan are computed by the individual robot control laws in (15).

We focus on two candidate quality measures for manipulation planning: a distance-based measure that seeks to minimize robot paths and a natural frequency-based measure that maximizes natural frequency of the payload around the stable equilibrium. For the distance-based quality measure, a candidate configuration is evaluated according to the function

$$D_{distance}(q^{k+1}, q^k) = ||q^{k+1} - q^k||.$$

The natural frequency-based quality measure is computed as

$$D_{frequency}(q^{k+1}, q^k) = -\text{eig}_0(\mathscr{H}(q^{k+1}))$$

where $\mathscr{H}(q)$ is the hessian of the potential energy as computed in [8] and $\mathrm{eig}_0(M)$ is the smallest eigenvalue of the matrix M.

5.2 Coordinated Control

In order for the payload to follow the desired trajectory $\mathbf{A}^k$, the individual robots must follow their computed plans in a coordinated way. That is, the control of the robots along their paths must occur in $\mathbb{R}^9$. Given the piecewise linear interpolation $q(s)$ of robot waypoints q^k and a current robot configuration q, we compute an instantaneous goal for the system as $q(s^* + \Delta s)$ where $q(s^*)$ is the closest point along the path to the current configuration and Δs relates to the velocity along the path by $v_{path} = \frac{\Delta s}{\Delta t}$ where v_{path} is chosen to be small enough to maintain our quasi-static assumption. If $q(s^* + \Delta s)$ is not in Q_M, a sweep along Δs is performed to find the next feasible point (the next waypoint q^{k+1} in the worst case). The individual robot control law (15) navigates around these infeasible regions.

6 Experimentation

Here, we evaluate the individual robot control laws and motion planning strategies presented in Sects. 4-5. In the presentation that follows we study the performance of the system shown in Fig. 3 to control along desired trajectories or to desired configurations via the previously discussed methods.

The experiments are conducted with the AscTec Hummingbird quadrotor [1], from Ascending Technologies GmbH, with localization information being provided by a Vicon motion capture system [2] running at 100 Hz with millimeter accuracy. Control commands are sent to each robot via Zigbee at 20 Hz. The quadrotor is specified to have a payload capacity of 0.2 kg.

Robots are positioned inside the 6.7 m × 4.4 m × 2.75 m workspace with an accuracy of approximately ±0.05 m in each direction. However, as the robots experience the effects of external loads and interactions during manipulation, this accuracy decreases to approximately ±0.15 m.

The payload is a rigid frame with cable attachment points given by $\tilde{x}_2^p = 1\,\mathrm{m}$, $\tilde{x}_3^p = 0.5\,\mathrm{m}$, and $\tilde{y}_3^p = 0.87\,\mathrm{m}$ with mass $m = 0.25\,\mathrm{kg}$. Therefore, the mass of the payload is more than the payload capacity of a single robot. The cable lengths are equal with $l_i = 1\,\mathrm{m}$.

6.1 The Quadrotor Robots

We begin by considering an aerial robot in $\mathbb{R}^3$ with mass m and position and orientation, $q = [x, y, z] \in \mathbb{R}^3$, and $\mathbf{R}(\alpha, \beta, \gamma) \in SO(3)$, respectively. We do not have direct access to the six control inputs for linear and angular force control, but instead have access to four control inputs related to these values. Therefore, we must restrict our interaction with the robot to the control inputs $\nu = [\nu_1, \ldots, \nu_4]$ defined over the intervals $[\nu_1, \nu_2, \nu_3] \in [-1, 1]$ and $\nu_4 \in [0, 1]$.

Based on system identification and hardware documentation, we assume the following transformation from the control inputs to the force control running on the robot (in the *body* frame of the robot):

$$\begin{aligned}\tau_x &= -K^q_{v,x}\dot{\alpha} - K^q_{p,x}(\alpha - \nu_1\alpha_{max})\\ \tau_y &= -K^q_{v,y}\dot{\beta} - K^q_{p,y}(\beta - \nu_2\beta_{max})\\ \tau_z &= -K^q_{v,z}\dot{\gamma} - K^q_{p,z}\nu_3\gamma_{inc}\end{aligned}$$

where τ is the torque applied by the robot in the body frame. Thrust along the z-axis in the body frame is defined as $f_z = \nu_4 f_{max}$. Through system identification, we determined values for these parameters. Therefore, in the discussion that follows, we consider the following desired inputs:

$$\left[\alpha^d, \beta^d, \gamma^d, f^d_z\right] = \left[\nu_1\alpha_{max}, \nu_2\beta_{max}, \nu_3\gamma_{inc}, \nu_4 f_{max}\right], \tag{16}$$

noting that we can compute the control inputs, ν_i, given the desired values with known parameters.

We now derive a transformation from point-model control in the world frame to the hardware-dependent control inputs, ν_i. Define the robot dynamics in the world frame, but only as a function of position ($q \in \mathbb{R}^3$), as

$$mI_3\ddot{q} = F_g + \left[F_x, F_y, F_z\right]^{\mathrm{T}}.$$

If we consider the forces applied to the system in the robot body frame,

$$\left[0, 0, f_z\right] = \left[F_x, F_y, F_z\right]\mathbf{R}(\alpha, \beta, \gamma), \tag{17}$$

we find a well-posed equation for α, β, and f_z given the current γ and forces F_x, F_y, and F_z in the world frame. Solving the system in (17), considering that $f_z \geq 0$ and physical system performance, yields a piece-wise smooth function such that we may define a force F in the world frame that is transformed into appropriate control input via (16) (see [7] for further details). To this end, we compute these values in implementation based on proportional-integral-derivative (PID) feedback control laws determined by the desired robot positions and velocities. For the purposes of this work, we additionally control $\gamma = 0$.

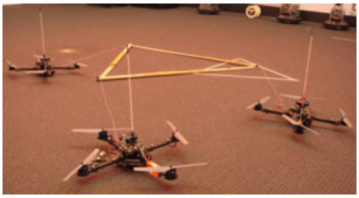


Fig. 3 The aerial robots and manipulation payload for experimentation. The payload is defined by $m = 0.25\,\mathrm{kg}$ and $\tilde{x}^p_2 = 1\,\mathrm{m}$, $\tilde{x}^p_3 = 0.5\,\mathrm{m}$, and $\tilde{y}^p_3 = 0.87\,\mathrm{m}$, with $l_i = 1\,\mathrm{m}$.

6.2 Results

Here we present three specific examples of aerial manipulation. The first highlights the manipulation of the payload along a complex trajectory. The second considers the effect of different quality measures on planning robot configurations. The final evaluation studies the capability of the control laws to maintain cone constraints when necessary. A *C* implementation of the SOCP allows us to solve the direct problem and efficiently construct and validate complex motion plans for the robots given a desired payload trajectory. Given these motion plans, we compute individual robot controls (15) via a *C*++ QP solver [4] to follow the desired trajectory with a 20 Hz update rate.

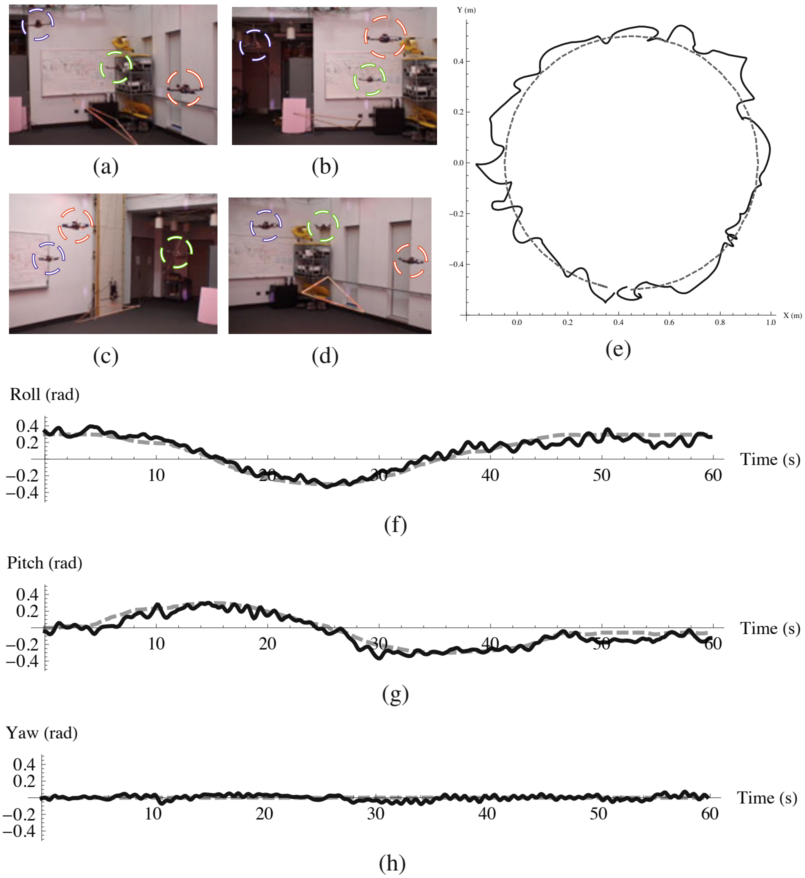


Fig. 4 Pose of the payload while being manipulated along the circular trajectory of Sect. 6.2.1. Figures 4(a)–4(d) are snapshots from an experimental trial. The robots control the payload in experimentation (solid line) along a desired trajectory (dashed line) (Figs. 4(e)–4(h)). A video of a trial run is available at http://kumar.cis.upenn.edu/movies/ISRR2009.flv

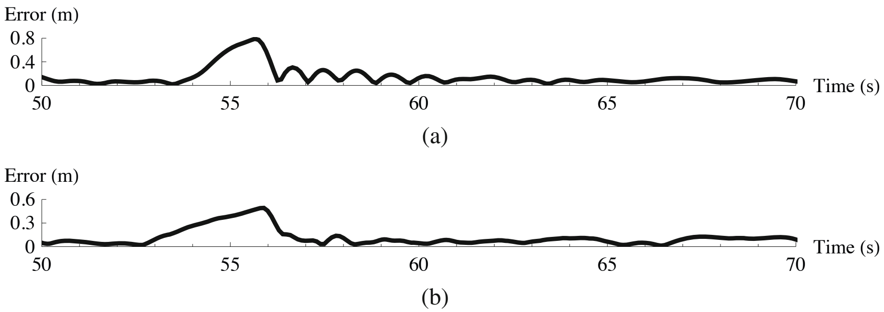


Fig. 5 A disturbance is applied to the payload at time 55 s. Notice that the configuration in Fig. 5(b), chosen to maximize natural frequency, attenuates error more quickly than the configuration in Fig. 5(a) where robot motions are minimized.

6.2.1 Trajectory Control

Depicted in Figs. 4(a)–4(d), this trial consist of a circular trajectory defined for the payload center of mass with varying roll and pitch such that

$$x_c(\theta) = 0.5\sin(\theta) + 0.45, \quad y_c(\theta) = 0.5\cos(\theta), \quad z_c = 1.0,$$
$$\alpha(\theta) = 0.3\cos(\theta), \quad \beta(\theta) = 0.3\sin(\theta), \quad \gamma = 0.$$

The motion plan is generated with $\theta = n\pi/16$ for $n = \{0,\ldots,32\}$ using the $D_{distance}$ metric that seeks to minimize the distance traveled by the robots.

Figures 4(e)–4(h) depict the trajectory followed by the payload and the control error of the robots following their planned trajectories. In this trial, a root-mean-square (RMS) error on robot control of approximately 9 cm yields RMS error on the payload of 7 cm for the center of mass and 3° for the orientation. We have observed similar robot control and payload error over many trials as detailed in Table 1. Each waypoint in the motion plan is selected to satisfy equilibrium conditions and maximum tension and cone constraints.

6.2.2 Effect of Planning Quality Measure

In this test, the payload is carried along a translational trajectory designed with both $D_{distance}$ and $D_{frequency}$. During transport, an external disturbance is applied to the payload and we observe the response. Figure 5 depicts the behavior of the motion plans for each measure and shows that the system configuration determined by the natural frequency-based measure attenuates payload error more quickly than the equivalent configuration specified by the distance-based measure.

6.2.3 Non-trival Waypoint Navigation

For a final experimental evaluation, we demonstrate the utility of the individual robot control laws for navigation between two waypoints when the straight-line

Table 1 RMS error of robot positions and payload position and orientation as compared to the desired robot-payload configuration. Note that the sixth trial corresponds to Fig. 4.

Trial	1	2	3	4	5	6
Robot Error (*m*)	0.095	0.102	0.106	0.103	0.098	0.087
Payload Center of Mass Error (*m*)	0.075	0.087	0.084	0.089	0.086	0.067
Payload Orientation Error (deg)	4.13	3.95	4.32	3.74	4.05	2.89

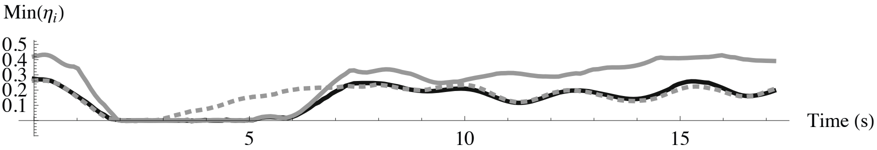


Fig. 6 Robot velocities are chosen to maintain $\eta_i > 0$ via the individual robot control law (15).

interpolated path is not feasible. The trial consists of desired waypoints requiring a rotation of the system by 120° about e_3. Figure 6 shows that the cone constraints are maintained throughout the trial and illustrates in areas of free-space that the planner need only focus on gross motions of the payload and robot configurations at the endpoints as the online controller will find the constraint-free robot trajectories.

7 Conclusion and Future Work

We presented individual robot control laws and motion plans that permit the control of the payload along a desired trajectory. We address the fact that robot configurations may admit multiple payload equilibrium solutions by developing constraints for the robot configuration that guarantee the existence of a unique payload pose. We formulate individual robot control laws that enforce these constraints and enable the design of non-trivial payload motion plans. We propose two quality measures for motion plan design that minimize individual robot motion and maximize payload stability along the trajectory. The methods proposed in this work are evaluated on a team of aerial robots in experimentation.

We are currently addressing limiting factors relating to system performance including the rate and effects of sensing and actuation errors on the control of the robots. To this end, we are implementing on-board estimation and control methods that consider the stochasticity of the system, increase control rates, and reduce our reliance on global localization. Additionally, we are exploring the limitations of the quasi-static assumption and the effect of transients on our individual robot control laws.

References

1. Ascending Technologies, GmbH, http://www.asctec.de
2. Vicon Motion Systems, Inc., http://www.vicon.com
3. Cottle, R.W., Pang, J., Stone, R.E.: The Linear Complementarity Problem. Academic Press, London (1992)
4. Fischer, K., Gartner, B., Schonherr, S., Wessendorp, F.: Linear and quadratic programming solver. In: Board, C.E. (ed.) CGAL User and Reference Manual (2007)
5. Hunt, K.: Kinematic Geometry of Mechanisms. Oxford University Press, Oxford (1978)
6. Lobo, M.S., Vandenberghe, L., Boyd, S., Lebret, H.: Application of second-order cone programming. Linear Algebra and Its Applications 284, 192–228 (1998)
7. Michael, N., Fink, J., Kumar, V.: Cooperative manipulation and transportation with aerial robots. In: Robotics: Science and Systems, Seattle, WA (2009)
8. Michael, N., Kim, S., Fink, J., Kumar, V.: Kinematics and statics of cooperative multi-robot aerial manipulation with cables. In: ASME Int. Design Engineering Technical Conf. & Computers and Information in Engineering Conf., San Diego, CA (2009) (to appear)
9. Murray, R.M.: Trajectory generation for a towed cable system using differential flatness. In: IFAC World Congress, San Francisco, CA (1996)
10. Phillips, J.: Freedom in Machinery, vol. 1. Cambridge University Press, Cambridge (1990)
11. Selig, J.M.: Geometric Fundamentals of Robotics. Springer, Heidelberg (2005)

Adaptive Highways on a Grid

Hajir Roozbehani and Raffaello D'Andrea

Abstract. The primary objective of this paper is to introduce the *adaptive highways* algorithm, a path planning algorithm for vehicles moving on a grid. We consider a workspace that consists of a symmetric grid and a large number of vehicles that move on the grid to accomplish a certain task. Each vehicle is assigned the task of visiting a set of randomly selected locations, which are updated over time. The dynamics of the vehicles are described by a constrained linear double-integrator model. The objective is to find, in real time, a set of trajectories that maximize the average speed of the vehicles while ensuring safety. The trajectory optimization problem is solved locally, whereas a central entity is employed for distribution of information. Safety guarantees are provided through a space reservation mechanism. Several algorithms are presented and compared in terms of performance.

1 Introduction

This paper is primarily concerned with path planning for vehicles that move on a grid to accomplish a certain task. The objective is to move the vehicles along collision-free trajectories from their starting points to their final destinations as quickly as possible. Various algorithms are explored in this context and evaluated in terms of robustness and performance.

A reservation mechanism is used to ensure collision-free trajectories. In other words, vehicles must reserve space before using it. Collision-free navigation can then be achieved if the vehicles are always contained within their reserved areas. Hence, regardless of the algorithm used for trajectory

Hajir Roozbehani
Automatic Control Laboratory, EPFL, Switzerland
e-mail: hajir.roozbehani@epfl.ch

Raffaello D'Andrea
Institute for Dynamic Systems and Control, ETHZ, Switzerland
e-mail: rdandrea@ethz.ch

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 661–680.
springerlink.com

optimization, the vehicles always remain safe. Three algorithms are explored for trajectory optimization: Reservation Based Algorithm (RBA), Fixed Highways Algorithm (FHA), and Adaptive Highways Algorithm (AHA).

The proposed algorithms quantify the desirability of a given candidate path in different ways. The RBA favors trajectories that do not pass through the current space reserved by other vehicles at any given time. The FHA builds on the RBA by further imposing a predefined structure that allows the vehicles to move only in certain directions in such a way that excludes the possibility of two vehicles running into head-to-head conflicts. The AHA achieves avoiding head-to-head conflicts without imposing any predefined structure; instead, it favors paths where the traffic flows in only one direction. As a result, highways emerge and periodically change direction. This paper compares the RBA, FHA, and AHA algorithms in simulated experiments and shows that the AHA demonstrates better performance over the other methods.

In addition to its promising performance in simulations with hundreds of vehicles, the AHA is further shown to be a real-time algorithm that can be implemented on systems comprising of large number of autonomous vehicles. The Kiva Mobile Fulfillment System (Kiva MFS), is one such system. In the Kiva MFS, hundreds of robots (see Fig.1) navigate around a warehouse to pick inventory items and carry them to fixed stations for packing, leading to increased productivity [2] [3] [11]. In such systems, a robot must complete its assigned task in the shortest possible time, while avoiding conflicts with all other robots. However, finding conflict-free minimum-time paths for such large number of robots in real time is a difficult problem. The AHA is shown to be able to tackle such large-scale problems in contexts similar to that of the Kiva MFS.

The underlying path planning problem has been widely studied in the literature and dates back to early research in robotics motion planning. A recent book that covers the vast field of algorithms and their applications to path planning is referenced in [4]. Path planning for autonomous vehicles in the presence of dynamic obstacles is relatively more recent. For example, a method based on rapidly-exploring random trees [5] is presented in [1]. The algorithm is based on randomly choosing intermediary waypoints that might generate feasible trajectories to the destination. The algorithm is powerful in adversarial settings where the robots have to generate trajectories on-line in unknown environments. Path planning has also been of high interest in co-operative settings. For instance, Raffard et al. [9] have developed an iterative algorithm based on decomposition of the global problem into a sequence of tractable sub-problems. A method based on decentralized receding horizon control is presented in [10]. Safety is guaranteed through maintaining a reachable loiter pattern in all intermediate planning steps. While keeping in mind that these results can potentially be extended to large-scale systems, most of the approaches focus on applications with small number of vehicles. Real-time implementation of such approaches, which rely in one way or another on expensive iterations of an optimization process, is not a trivial task. Other

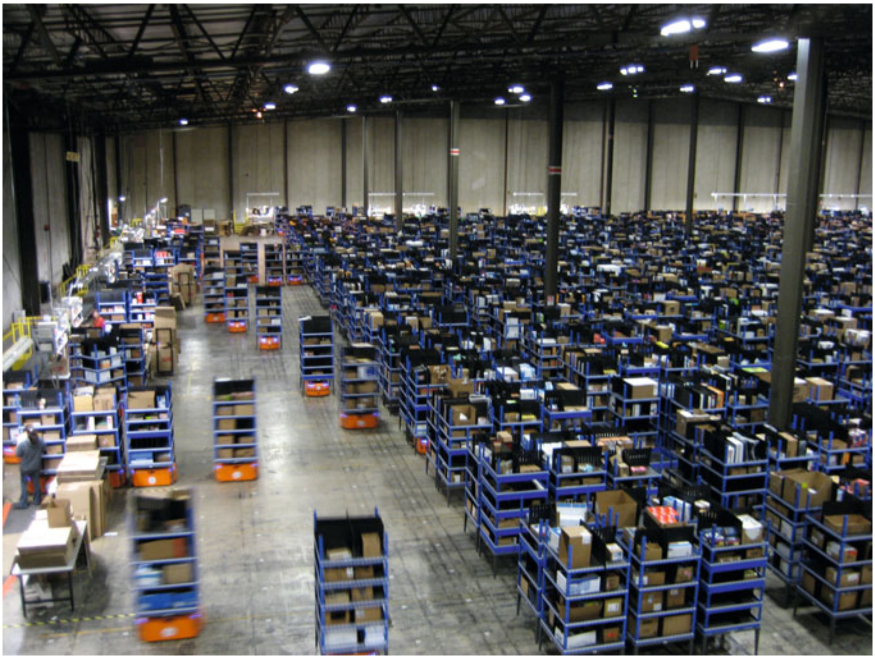


Fig. 1 A Kiva warehouse. The blue shelves hold the inventory and can be moved by the robots. The workers stay at fixed stations and the robots move around the warehouse to pick up the inventories and bring them to the stations. The underlying navigation grid can be inferred from the marks left by the robot wheels.

approaches that move toward full decentralization provide promising scalability features. These, however, come at the cost of decreased performance. For instance, a fully decentralized approach to multi-vehicle path planning is presented in [8]. Note, however, that the proposed algorithm behaves rather conservatively since it perceives any intersecting paths as a conflict without considering the time course of the vehicles on their paths.

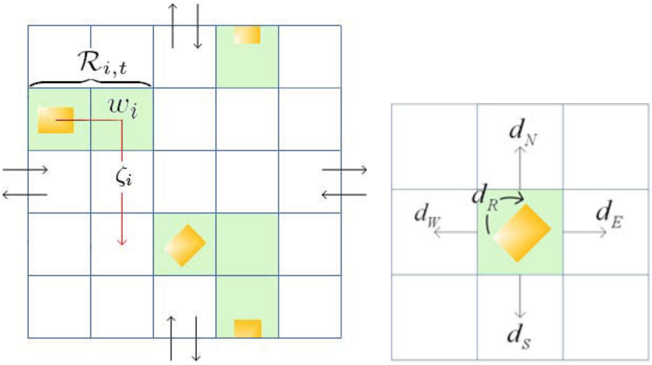
The AHA is based on predicting the potential conflicts and evaluating the expected time loss they incur. To do so, the algorithm explores an extra dimension in its search space to determine if any conflict occurs along a candidate path based on a time estimation of the available trajectories. Furthermore, the AHA algorithm distinguishes three different types of conflicts: head-to-head, head-to-side, and head-to-back. This enables it to favor more moderate conflicts (such as head-to-side) to tighter ones (such as head-to-head) on a candidate trajectory.

2 Preliminaries

This works considers the optimization problem of finding the minimum time paths for a group of N vehicles with independent dynamics moving under both coupled and individual constraints. The vehicles are assumed to navigate on a workspace $\mathcal{W} \subset \mathbb{R}^2$ and follow target points drawn from a spatially uniform distribution. The workspace, as shown in Fig.2a, is a symmetric 2-torus. The working region is further discretized into a set of M identical partitions, called cells $\mathcal{W}_1, ..., \mathcal{W}_M$, such that $\bigcup_{j=1}^{M} \mathcal{W}_j = \mathcal{W}$ and $\mathcal{W}_i \bigcap_{i \neq j} \mathcal{W}_j = \emptyset$. Every cell $\mathcal{W}_j$ represents a unit square:

$$\mathcal{W}_j = \{(x, y) \in \mathbb{R}^2 \mid \ \bar{x}_j - \frac{1}{2} \leq x < \bar{x}_j + \frac{1}{2},\ \bar{y}_j - \frac{1}{2} \leq y < \bar{y}_j + \frac{1}{2}\}$$

with $(\bar{x}_j, \bar{y}_j) \in \{\frac{1}{2}, \frac{3}{2}, ..., \frac{\sqrt{M}}{2}\} \times \{\frac{1}{2}, \frac{3}{2}, ..., \frac{\sqrt{M}}{2}\}$ being the center of the cell. It is assumed that a vehicle can be contained within a cell. More details on the vehicles will be given later. Once a vehicle is contained in the interior of its destination cell, a new target point is assigned to it.



(a) The world model (b) The atomic decisions

Fig. 2 *a) The world model*: The navigation map is a symmetric grid with wraparound. If a vehicle leaves the map from one edge, it enters it from the opposite side, similar to The Game of Life or the PacMan video game from the 80's. $\mathcal{R}_{i,t}$ denotes the reserved space (see 2.4) of vehicle i at step t. ζ_i is a discrete trajectory (see 2.2). w_i denotes the current destination, which is a corner point (or breakpoint) located along the robot's trajectory. *b) The atomic decisions*: The atomic decisions either impose a rotation or guide the vehicle toward one of its neighboring cells.

2.1 Vehicle Dynamics

The position of every vehicle is denoted by $q_i = (x_i, y_i) \in \mathcal{W}$ and the heading is represented by $\theta_i \in [0, \frac{\pi}{2}]$, measured with respect to the global reference frame. We denote by $v_i \in \mathcal{V} = [-\bar{v}, \bar{v}]$ the linear velocity, and by $\omega_i \in \Omega = [-\bar{\omega}, \bar{\omega}]$ the rotational velocity of vehicle i. The admissible linear and rotational accelerations are represented by $u_i \in \mathcal{U} = [-\bar{u}, \bar{u}]$ and $\alpha_i \in \vartheta = [-\bar{\alpha}, \bar{\alpha}]$, respectively. The vehicle dynamics are modeled by a double integrator system under input and velocity constraints (see Fig. 3). It is assumed that, at any given time, the vehicle can either rotate or move in only one direction, and that transitions between horizontal ($\theta = 0$) and vertical ($\theta = \frac{\pi}{2}$) states must go through a rotational state:

$$\ddot{x}_i = u_i,\ \dot{x}_i \in \mathcal{V},\ \dot{y}_i = 0,\ \dot{\theta}_i = 0,\ \theta_i = 0,\ y_i \in \{\bar{y}_j\} \tag{1a}$$

$$\ddot{\theta}_i = \alpha_i,\ \dot{\theta}_i \in \Omega,\ \dot{x}_i = 0,\ \dot{y}_i = 0,\ (x_i, y_i) \in \{\bar{x}_j, \bar{y}_j\} \tag{1b}$$

$$\ddot{y}_i = u_i,\ \dot{y}_i \in \mathcal{V},\ \dot{x}_i = 0,\ \dot{\theta}_i = 0,\ \theta_i = \frac{\pi}{2},\ x_i \in \{\bar{x}_j\} \tag{1c}$$

These constraints are incorporated in the task specification process and fulfilled during task implementation. It is assumed throughout the paper that the input and state constraints are $\bar{u} = \bar{v} = 1$ for translational motions and $\bar{\alpha} = \bar{\omega} = 2\pi$ for rotational motions. Note that the above choice of dynamics in combination with the gridded workspace resembles the navigation scheme of the Kiva MFS shown in Fig.1.

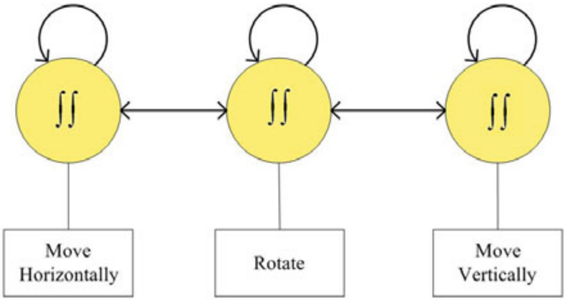


Fig. 3 The three states of Eq.1. In order to transit from moving horizontally (Eq.1a) to moving vertically (Eq.1c), the vehicle must visit an intermediary rotational state (Eq.1b).

2.2 Trajectories

The motion characteristics of the vehicle can be abstracted into a finite state model. This is useful for the path planning algorithms in Section 4. Continuous variables q_i and θ_i can be mapped into the discrete state variable $\sigma^i = \{(\mathrm{x}^i, \mathrm{y}^i), \mathrm{R}^i\}$. The pair $(\mathrm{x}^i, \mathrm{y}^i)$ represents the cell position of q_i (i.e., the center of the cell that contains q_i). For example, $(0.75, 0.5)$ is mapped to $(\frac{1}{2}, \frac{1}{2})$. Furthermore, the discrete rotational state R^i is defined by:

$$\mathrm{R}^i = \begin{cases} \mathrm{H} & 0 \leq \theta_i < \frac{\pi}{4} \\ \mathrm{V} & \frac{\pi}{4} \leq \theta_i \leq \frac{\pi}{2} \end{cases}$$

In other words, a vehicle is perceived to be in a horizontal/vertical state H/V if its orientation is closer to the horizontal/vertical axis. The set of admissible decisions is denoted by $\mathcal{D} = \{d_N, d_S, d_E, d_W, d_R, d_\emptyset\}$. The atomic decisions presented in Fig.2b define discrete steps to move either horizontally $\{d_E, d_W\}$ or vertically $\{d_N, d_S\}$, to rotate d_R, or to wait and do nothing $d_\emptyset$. Naturally, the discrete states evolve according to the adopted decisions. This can be described with a discrete dynamical system:

$$\sigma^i_{k+1} = \mathcal{F}(\sigma^i_k, d^i_k) \tag{2}$$

For example, $\mathcal{F}(\frac{1}{2}, \frac{1}{2}, \mathrm{H}, d_E) = (\frac{3}{2}, \frac{1}{2}, \mathrm{H})$. Furthermore, note that vertical/horizontal moves are not allowed when the vehicle is facing horizontally/vertically.

A discrete trajectory ζ_i (see Fig.4), which leads vehicle i from its starting coordinate to its target, is defined as follows:

Definition 1. A discrete trajectory,

$$\zeta_i = \big\{(\sigma^i_0, t^i_0, d^i_0), (\sigma^i_1, t^i_1, d^i_1), .., (\sigma^i_n, t^i_n, d_\emptyset)\big\}$$

is a finite sequence whose elements are the vehicle's discrete position and orientation σ^i_k, a time estimate t^i_k, and an atomic decision d^i_k with $d^i_n = \emptyset$.

2.3 Central Hub

Although the trajectory optimization problem is solved locally, the vehicles can rely on a central entity for communication purposes. We assume that an omniscient central hub $\mathcal{H}$ exists that is aware of the reserved space and discrete trajectory of every vehicle. This central hub can be considered as a blackboard on which every vehicle can write about its intentions and status. Furthermore, the central hub imposes a fixed ordering on the vehicles for communicating to the hub, reserving space, and path planning. Note that

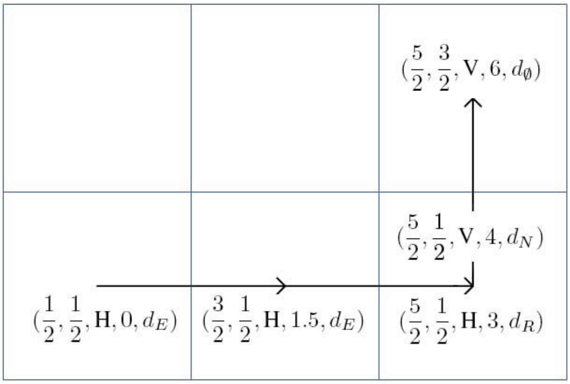

$(\frac{5}{2}, \frac{3}{2}, \mathrm{V}, 6, d_\emptyset)$
$(\frac{5}{2}, \frac{1}{2}, \mathrm{V}, 4, d_N)$
$(\frac{1}{2}, \frac{1}{2}, \mathrm{H}, 0, d_E)$
$(\frac{3}{2}, \frac{1}{2}, \mathrm{H}, 1.5, d_E)$
$(\frac{5}{2}, \frac{1}{2}, \mathrm{H}, 3, d_R)$

Fig. 4 A discrete trajectory. The vehicle starts in a horizontal state $\mathrm{R}_0^i = \mathrm{H}$ at time $t_0^i = 0s$. Given the constraints on the maximum velocities and accelerations, it follows that rotations and horizontal/vertical moves take $1s$. Note that each acceleration/deceleration takes an extra $0.5s$.

the information stored in the hub contains discrete states of the vehicles and is updated at a relatively slow rate. Hence, the communication and memory requirements of the central hub are modest. Distribution of information can be achieved in a decentralized fashion as well, by exploiting geometrical distribution of the vehicles.

2.4 Ensuring Collision-Free Paths

A reservation mechanism ensures that the vehicles move on collision-free paths. In short, the mechanism requires the vehicles to reserve space as they move. The reserved space of each vehicle (the green areas in Fig. 2a) is a sequence of cells located on its trajectory that contain the vehicle itself. As long as a cell is reserved by some vehicle, no other vehicle can use it. Furthermore, it is required that each vehicle can stop within its reserved area without requiring any more reservations. Hence, the vehicles are always contained within their exclusive reserved areas, ensuring safe navigation. In addition, once a vehicle is finished using its reserved space, it releases that space so that other vehicles can use it.

To express the reservation mechanism more formally, we let the collision region $P_{i,t} \subset \mathcal{W}$ indicate the smallest union of cells that fully contains vehicle i at time t. Collision avoidance requires that $P_{j,t} \bigcap_{i \neq j} P_{i,t} = \emptyset$ for all t. At any time $t \in [t_k^i, t_{k+1}^i]$, the collision region is a subset of the cells corresponding to $\{\sigma_k^i, \sigma_{k+1}^i\}$. If the vehicles could instantly transfer into a stationary state in their $P_{i,t}$ zones, safety would be guaranteed by making the $P_{i,t}$ zone always unreachable to $j \neq i$. A conflict zone comprised of all cells on the discrete trajectory that lie on vehicle i's path until it can fully stop is considered, in order to ensure safety under dynamical constraints,. This gives a lower bound on the number of cells that each vehicle must reserve. In the example shown in Fig.4, the conflict zone at $t = 0s$ is the cell centered at $(\frac{1}{2}, \frac{1}{2})$ and at $t = 0.5s$ is the set of cells centered at $(\frac{1}{2}, \frac{1}{2})$ and $(\frac{1}{2}, \frac{3}{2})$. As mentioned earlier, it is required that every vehicle i be completely contained inside its reserved area $\mathcal{R}_{i,t}$ at all times. Therefore, collision-free navigation is guaranteed if $\mathcal{R}_{j,t} \bigcap_{i \neq j} \mathcal{R}_{i,t} = \emptyset$.

From a single vehicle point of view, the optimal reservation policy for vehicle i is to request a space that contains its entire trajectory. However, this prevents other vehicles from using the available cells. Hence, for performance reasons, the request from the central hub contains just enough space so that the vehicle can be safe while moving at maximum speed. Every vehicle sets a candidate goal point $\tilde{g}_i \in \mathbb{R}^2$:

$$\tilde{g}_i = q_i + ({}^{\bar{v}^2}\!/_{2\bar{u}} + 2h\bar{v} + \frac{1}{2})\mathrm{sgn}(w_i - q_i).$$

where h is a discrete sampling time and w_i denotes the current destination. Note that since diagonal moves are not allowed, the vector $\mathrm{sgn}(w_i - q_i)$ always has one zero element. Next, the vehicle evaluates its next target point:

$$g_i = \min_{r \in \{\tilde{g}_i, w_i\}} \{\|q_i - r\|\}.$$

When the next target point is not included in the vehicle's reserved area $g_i \notin \mathcal{R}_{i,t}$, a request is sent to the central hub. The maximum number of cells to be requested is then:

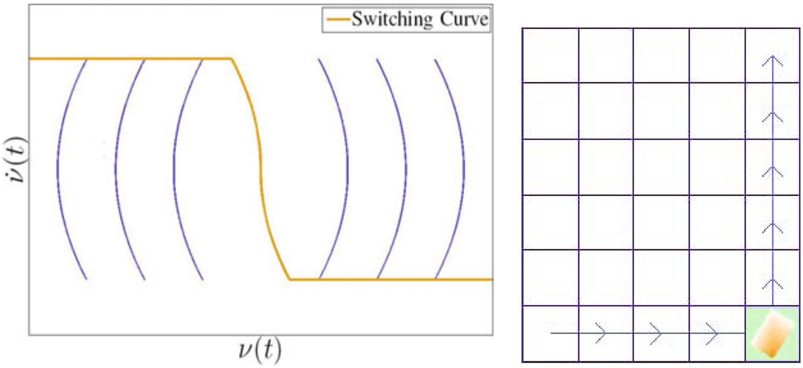
$$\mathrm{ceil}({}^{\|q_i - g_i\|}\!/_{l})$$

Given that $\bar{v} = \bar{u} = 1$, the reserved area $\mathcal{R}_{i,t}$, with $t \in [t^i_k, t^i_{k+1}]$, is a subset of the cells corresponding to $\{\sigma^i_k, \sigma^i_{k+1}, \sigma^i_{k+2}\}$. Also, note that during rotations we have: $w_i = q_i$ and the vehicle is safe by remaining in its own cell. After receiving a response packet from the central hub, the reserved space $\mathcal{R}_{i,t}$ is updated and the next target point is modified such that it can be contained within the reserved area. Therefore, $g_i = \min_{r \in \mathcal{R}_{i,t}} \{\|w_i - r\|\}$. At this time, the vehicle also releases the unneeded portion of its reserved space.

The above scheme ensures that the vehicles remain within their reserved spaces at all times.

2.5 Problem Statement

Problem 1: Consider a group of N vehicles that follow the dynamics given in (2), can communicate with a central entity $\mathcal{H}$, and comply with the rules of the reservation mechanism. Given a fixed ordering defined by the central hub, find for each $i \in N$, a discrete trajectory ζ^*_i that minimizes the required time to reach its destination.



(a) Phase portrait of the minimum-time path (b) The generated trajectory

Fig. 5 The continuous, minimum-time trajectory: *a) Phase portrait of the minimum-time path:* The optimal controller leads the vehicle toward its current destination with at most two switches, which occur on the switching curve (the orange line). *b) The generated trajectory:* A vehicle, its minimum-time trajectory, and its reserved zone are shown in an obstacle free environment.

3 Control Architecture

A two-layer control architecture is proposed. At the higher layer, a discrete trajectory is extracted from a discrete search algorithm implemented on a local planner. Then, a low-level controller provides a continuous execution in order to implement and preserve the properties of the discrete trajectory. Note that the planners act centrally in the sense of information retrieval, but solve the path planning problem locally based on the information they have about other vehicles.

Once the discrete trajectory is decided by the planner, the vehicle looks for break points in the trajectory, where it must come to a full stop. Hence, every vehicle heads to a current destination $w_i \in \mathcal{W}$ with the intent of reaching a final destination denoted by $f_i \in \mathcal{W}$. The vehicles adopt a bang-bang control [6] policy in order to traverse between current destinations as well as to rotate.

The minimum time trajectories for a single vehicle scenario are shown in Fig. 5. As can be seen from the figure, the optimal controller has no more than two switching points. Note that each degree of freedom is controlled independently and the above figure corresponds to a general coordinate ν_i. Each vehicle traverses on the minimum time trajectories with the maximum acceleration until it reaches its switching curve. At this point, the vehicle starts coasting at maximum possible speed. Once the turning point is reached, the vehicle moves toward its current destination with maximum deceleration. Interested readers can refer to [7] for a thorough analysis of the adopted control policy. Note that in the presence of multiple vehicles the discrete trajectories extracted from the planner might be suboptimal. Nevertheless, the optimal policy at the controller level remains the same.

4 Path Planning Algorithms

This section is concerned with presenting three path planning algorithms in the scope of *Problem 1.* In short, the algorithms estimate the required time for a transition in discrete states, and further compute a feasible combination of transitions that minimizes the accumulated estimated time to reach a final state. The estimated time for transitions varies from one algorithm to another.

In the RBA (Reservation Based Algorithm), in addition to the traveled distance, the current reserved areas of the vehicles are taken into account. For each vehicle, the algorithm penalizes a candidate path for passing through a cell reserved by other vehicles based on the current distance between the vehicle and the reserved cell. In the FHA (Fixed Highways Algorithm), in addition to accounting for reservations, a highway structure dividing the workspace into odd and even lanes is imposed. The vehicles are allowed to move in only one direction on each lane. Hence, the vehicles never encounter one another in a head-to-head situation. In other words, the algorithm prunes any subset of the search space that may potentially lead to head-to-head conflicts. The drawback is that a large portion of the search space, which could otherwise be exploited by the vehicles, is omitted in a conservative way. The AHA (Adaptive Highways Algorithm), however, is based on estimating whether a conflict occurs on a candidate path or not. This is achieved by comparing the time estimations and atomic decisions on the candidate path with those available in the trajectories of other vehicles. By taking the time and decisions into account, the AHA is able to omit only unpromising candidate paths rather than a large subset of the search space. In the rest of this section, we provide details on how the algorithms work.

4.1 Description of Cost-to-Go

The time required for transition from a state σ_k^i to a neighbor state σ_{k+1}^i is denoted by $\mathcal{L}(\sigma_k^i, d_k^i)$. A lower bound on $\mathcal{L}(.,.)$ is given by:

$$\mathcal{L}(\sigma_k^i, d_k^i) = \tau_{min}(\mathcal{F}(\sigma_k^i, d_k^i), \sigma_k^i)$$

with $\tau_{min}(\mathcal{F}(\sigma_k^i, d_k^i), \sigma_k^i) = 1/\bar{v}$ if the vehicle is to move either horizontally or vertically, i.e., $\mathrm{R}_k^i = \mathrm{R}_{k+1}^i$. If a rotation must be performed, a minimum of $\tau_{rot}(\mathcal{F}(\sigma_k^i, d_k^i), \sigma_k^i) = 2\sqrt{\pi/2\bar{\alpha}}$ must be considered in the cost. Therefore, if the transition is compatible with the safety requirements posed by the reservation mechanism, the transition time is restricted only by the kinematic and dynamic constraints. Furthermore, the heuristic cost-to-go to the terminal state σ_f^i is simply:

$$\lambda(\sigma_k^i, \sigma_f^i) = \frac{1}{\bar{v}} d_{man}(\sigma_k^i, \sigma_f^i) + \tau_{rot}(\sigma_f^i, \sigma_k^i)$$

where $d_{man}(\sigma_k^i, \sigma_f^i)$ is the manhattan distance to the destination cell and $\tau_{rot}(\sigma_f^i, \sigma_k^i)$ is a lower bound on the potential rotations that might take place in the future. Here, we account only for one such rotation. Moreover, the stage cost is defined as follows:

$$\phi(\sigma_0^i, \sigma_k^i) = \sum_{j=0}^{k} \mathcal{L}(\sigma_j^i, d_j^i) + \lambda(\sigma_k^i, \sigma_f^i)$$

With the above ingredients, a standard A* algorithm can be applied to find the shortest path, i.e., to minimize $\phi(\sigma_0^i, \sigma_f^i)$.

4.2 Reservation Based Algorithm

The minimum-time optimization problems considered for each vehicle in *Problem 1* are coupled through the reservations. This has several implications on the design of the algorithms. In fact, algorithms that react only myopically to avoid collisions with other vehicles, fail to demonstrate an adequate level of performance in multi-vehicle settings. This can be understood by noting that, in the presence of multiple vehicles, the system may get stuck in a local minima or in an oscillatory trajectory. Fig. 6 shows a scenario where the vehicles myopically account for the position of each other without taking the reserved areas or future decisions of each other into account. In this case, the transition cost penalizes trajectories that pass through the current cells of other vehicles:

$$\mathcal{L}(\sigma_k^i, d_k^i) = \tau_{min}(\mathcal{F}(\sigma_k^i, d_k^i), \sigma_k^i) + \delta(\sigma_k^i, d_k^i, \sigma_0^{-i}) \tag{3}$$

where σ_0^{-i} denotes the initial discrete states of other vehicles $j \neq i$ at the start of the planning, which is always assumed to be at $t = 0s$ for convenience. In this algorithm, $\delta(.,.,.)$ is defined as follows:

$$\begin{cases} \delta(\sigma_k^i, d_k^i, \sigma_0^j) = \hat{\delta} & \text{if discrete position of } \sigma_k^i = \text{discrete position of } \sigma_0^j \\ \delta(\sigma_k^i, d_k^i, \sigma_0^j) = 0 & \text{otherwise} \end{cases}$$

where $\hat{\delta}$ is a non-zero constant. In other words, if the node σ_k^i expanded at the k-th layer of the search algorithm is occupied by another robot, the transition cost increases by a constant. It can be shown that there exists a critical cost δ_c such that for $\hat{\delta} < \delta_c$ the vehicles can get stuck in a local minima and for $\hat{\delta} \geq \delta_c$ they oscillate between two cells. If the vehicles take the reserved spaces

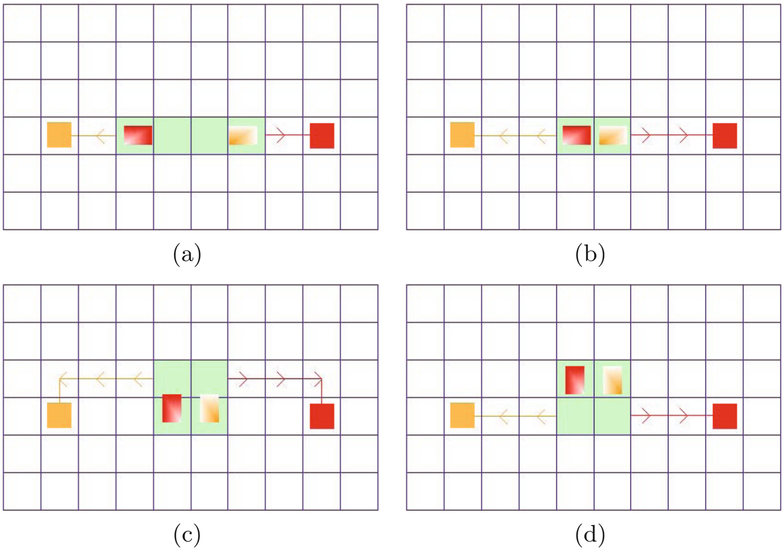


Fig. 6 A two vehicle scenario with myopic planning. The big squares are the destination cells, the rectangles are the vehicles, the arrows are the trajectories, and the green cells are the reservations. The red vehicle (on the left) is heading to the red square (on the right). Note that the reservations cover the arrows. *a)* The vehicles move toward their destinations. *b)* The vehicles encounter a head-to-head conflict and cannot further reserve the space needed to move on their trajectories. They start re-planning. *c)* First, the red vehicle plans a trajectory around the orange one. The trajectory goes up and continues to the right as shown in the figure. The orange vehicle starts planning immediately after the red vehicle and avoids its current position. Consequently, both vehicles go up. *d)* They meet again and the scenario repeats. Therefore, the vehicles demonstrate vertical oscillations. Re-planning can not resolve the oscillation deadlock.

of one another into account, the oscillation deadlocks are resolved as shown in Fig.7a. The transition cost is modified to:

$$\mathcal{L}(\sigma_k^i, d_k^i) = \tau_{min}(\mathcal{F}(\sigma_k^i, d_k^i), \sigma_k^i) + \delta(\sigma_k^i, d_k^i, \mathcal{R}_{-i,0}) \qquad (4)$$

where

$$\begin{cases} \delta(\sigma_k^i, d_k^i, \mathcal{R}_{j,0}) = \hat{\delta} & \text{if discrete position of } \sigma_k^i \in \mathcal{R}_{j,0} \\ \delta(\sigma_k^i, d_k^i, \mathcal{R}_{j,0}) = 0 & \text{otherwise} \end{cases}$$

In addition, the utilized information $\mathcal{R}_{-i,0}$ is valid only within a short time horizon close to the start of planning. The reserved areas that are located at a relatively far distance from σ_0^i are likely to have been released by the time the vehicle wants to use them. For this reason, an estimation horizon $\hat{e}$ can be introduced in the planner such that:

$$\delta(.,.,.) = \begin{cases} \delta(.,.,.) & \mathrm{dist}(\sigma_0^i, \sigma_k^i) \leq \hat{e} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

However, the performance may still be substantially influenced by conflicts. This can be better understood by realizing that, in this scheme, up to four equally optimal trajectories might exist. Hence, if a vehicle can predict that a conflict might occur on one of its shortest paths, it can choose an alternative, conflict-free trajectory. Moreover, the second order nature of the vehicles' dynamics penalizes switches on the control input. Hence, predicting the conflicts, especially in scenarios which involve a head-to-head encounter similar to Fig. 6, enables the vehicles to traverse with a smaller number of switchings on their control trajectory.

4.3 Fixed Highways Algorithm

A highway structure can be imposed on the RBA explained above. This leads to a navigation scheme prohibiting head-to-head encounters such as the scenario in Fig.7. In the FHA, the lanes are numbered with integers and the vehicles are allowed to move in only one direction on each lane based on the lane number being odd or even.

Although the highway structure resolves the head-to-head conflicts in a subtle way, the vehicles travel a longer distance on average. Furthermore, deadlocks similar to Fig. 8a may still happen. In order to circumvent the deadlock, the vehicles exploit the modified cost defined in either Eq. 3 or Eq. 4 combined with the estimation horizon defined in Eq. 5. The transition cost $\hat{\delta}$ must be high enough to force the vehicles to deviate from their paths in a head-to-side conflict. However, if there is no deadlock, the deviation

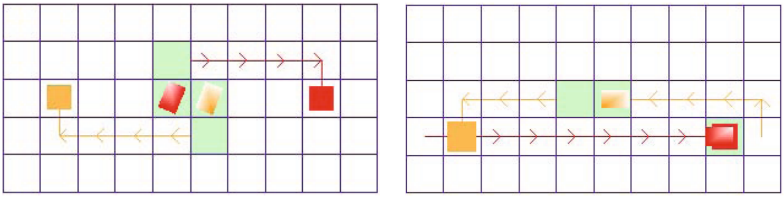

(a) Reservation Based Algorithm (b) Adaptive Highways Algorithm

Fig. 7 Comparing the RBA with the AHA in a two vehicle scenario. *a) Reservation Based Algorithm:* The vehicles first meet in a head-to-head situation and start re-planing similar to Fig. 6b. The vehicles proceed to reserving space immediately after planning. Hence, the symmetry in the conflict is broken, which leads to the resolution of the deadlock. However, the conflict has an adverse effect on the performance of both vehicles. *b) Adaptive Highways Algorithm:* The red vehicle plans first and the orange vehicle plans after it. Once the red vehicle plans a straight trajectory to its destination, the orange vehicle takes a detour to avoid any head-to-head conflicts.

diminishes the performance of the algorithm in head-to-side encounters (see Fig.9) where deviation is likely to be suboptimal.

4.4 Adaptive Highways Algorithm

In the AHA algorithm, each vehicle attempts to predict the future conflicts with other vehicles along a candidate path based on the list of trajectories (cell positions, decisions, and time estimates) stored in the central hub. The inclusion of time estimates in the algorithm requires augmenting a time element to the states of the search space, which further induces a discrete search on a hyper graph with a time dimension. To augment the states by time, a discrete planning step can be defined to incrementally increase the state of the vehicle as a function of time during the planning. Note that the dynamic constraints can be used to significantly reduce the dimension of the search problem and make the algorithm faster.

The actual time required for any transition through the states of a trajectory depend on whether or not the vehicle can reserve the corresponding cells. Moreover, the availability of the space $\mathcal{R}_{i,k}$ requested at the k-th layer of a trajectory ζ_i depends on ζ_j for $j \neq i$, more conveniently denoted by ζ_{-i}. Hence, by comparing the pairs $\{\sigma_k^i, t_k^i, d_k^i\}$ on a candidate path with those in ζ_{-i} during planning, we can more accurately determine if any conflict occurs along the candidate path. The transition cost can then be defined as follows:

$$\mathcal{L}(\sigma_k^i, d_k^i) = \tau_{min}(\mathcal{F}(\sigma_k^i, d_k^i), \sigma_k^i) + \delta(\sigma_k^i, d_k^i, \zeta_{-i}) \tag{6}$$

Note that the direction of the expected encounter can be inferred from the decision elements of the trajectories. Since the time loss due to a conflict depends on the direction of the conflict, it is likely that one has to distinguish the constraints from each other, depending on the direction of the expected encounters. For instance, the optimal cost for head-to-head encounters is likely to be higher than that of head-to-side encounters since it must be high enough to encourage deviations as can be seen in Fig.7b.

4.5 Congestion Avoidance

The cost can be further adapted to account for static vehicles. The main idea is to scan the level of traffic at each cell and associate an integral gain with the cells that contain static vehicles. A first order system is considered for the congestion equation:

$$\dot{c}(\sigma) = \frac{1}{\tau}(-c(\sigma) + c_{\max}\hat{c}_{in}(\sigma))$$

$$\hat{c}_{in}(\sigma) = \begin{cases} 1 & \text{if } v_i = 0 \text{ and } i \text{ is located at } \sigma \\ 0 & \text{otherwise} \end{cases}$$

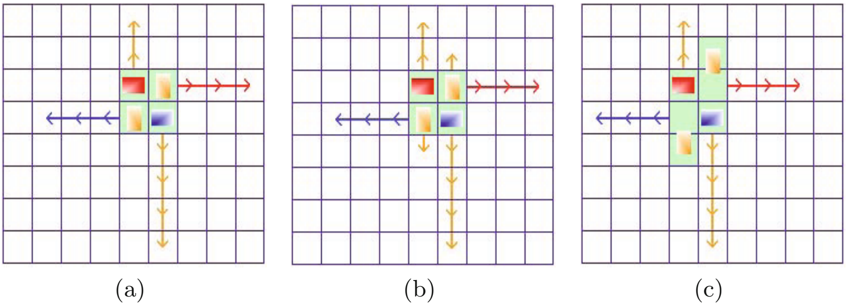


Fig. 8 A deadlock scenario resolved with the AHA. *a)* Four vehicles are put in a deadlock scenario. Each vehicle blocks the path of its counter-clockwise neighbor. (The blue vehicle blocks the path of the orange vehicle, which itself blocks the path of the red one, etc.). *b)* The vehicles re-plan. The two orange vehicles plan on moving backwards one step and heading to their destinations after the others pass. *c)* The conflict is resolved. Note that the solution is based on accurate timing between the vehicles. Solving the same deadlock involves deviations in both the RBA and the FHA.

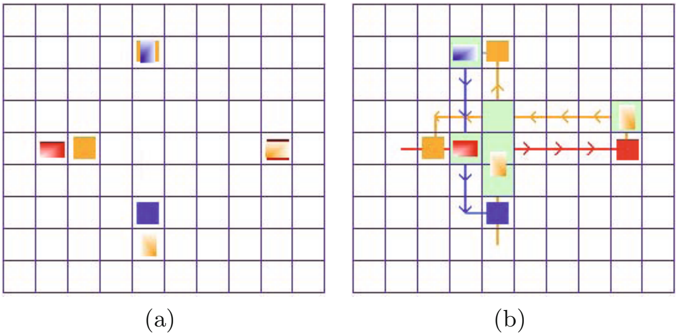


Fig. 9 A four vehicle scenario using the AHA. The big squares are the destinations and the rectangles are the vehicles. *a)* The vehicles face a mixture of head-to-head and head-to-side conflicts *b)* The vehicles deviate to avoid head-to-head conflicts and wait on their paths to resolve head-to-side conflicts.

requiring two more parameters τ, c_{max} to be optimized. The transition cost is further modified to:

$$\mathcal{L}(\sigma_k^i, d_k^i) = \tau_{min}(\mathcal{F}(\sigma_k^i, d_k), \sigma_k^i) + \delta(\sigma_k^i, d_k^i, \zeta_{-i}) + c(\sigma_k^i)$$

The main impact of the congestion avoidance is to add robustness to the proposed algorithms.

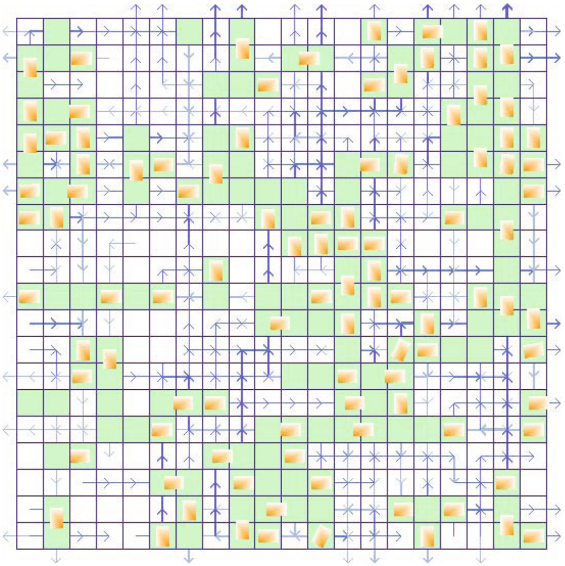


Fig. 10 A view of the platform in operation: the green areas are the reserved spaces and the arrows represent the trajectories. The thickness of the lines are proportional to number of trajectories passing through a certain cell.

5 Results

We developed a C++ platform (see Fig. 10) to study the performance of the proposed algorithms. The simulation environment is a test-bed that captures the path planning aspects of a large-scale multi vehicle system. To examine the efficacy of the proposed algorithms, we conducted several systematic experiments. Results are provided in this section. Fig. 11 demonstrates the performance of different algorithms presented above. Experiments were carried out on a 20×20 grid, on systems ranging from twenty to two hundred vehicles in size. The parameters of each algorithm were optimized at each density using a line search algorithm. It should be pointed out that, in all algorithms, if a vehicle does not move for a certain amount of time $\hat{t}$, it replans again to avoid congestions. As can be seen from the results, the AHA demonstrates superiority over the other methods. Hence, we consider it as the main approach for further analysis. The parameters required to be optimized for the algorithms are presented in Table 1 followed by a description for each parameter.

Furthermore, the algorithm is promising from the computation time point of view. As shown in Table 2, the algorithm performs seven times faster than real time in an experiment on a 50×50 grid with 625 vehicles, corresponding to a density of 0.25. The computation times provided in Table 2 can give some insight on the real-time implementability of the algorithms. Experimental

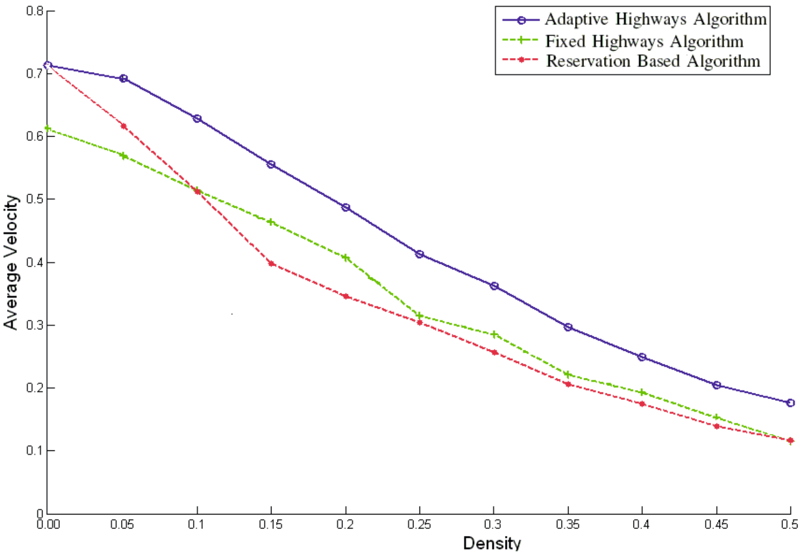


Fig. 11 Comparing the performance of different algorithms in simulation.

Table 1 The parameters of the algorithms. The optimization is done based on the average speed of the vehicles after 200 seconds of simulated time using a line search algorithm at a density of 0.25 on a 20 × 20 grid. $\hat{t}$ is the maximum wait time before re-planing as explained earlier in this section. τ and c_{max} are the time constant and maximum cost of congestion. $\hat{\delta}$ is the cost associated with conflicts in the RBA and the FHA. $\hat{e}$ is the estimation horizon. $\hat{\delta_h}$ is the cost associated with head-to-head conflicts in the AHA. As expected, this value is higher than $\hat{\delta_s}$, which corresponds to head-to-side conflicts. $\hat{\delta_0}$ corresponds to head-to-back encounters where vehicles plan to move toward the same direction. An optimal value of 0 implies that the algorithm encourages platooning.

Algorithm	$\hat{t}$	τ	$\hat{e}$	c_{max}	$\hat{\delta}$	$\hat{\delta_h}$	$\hat{\delta_s}$	$\hat{\delta_0}$
RBA	$2s$	$1s$	2	15	10	-	-	-
FHA	$2s$	$1s$	1	15	8	-	-	-
AHA	$2s$	$1s$	-	10	-	13	2	0

evidence suggests that the complexity of the algorithm grows almost linearly with respect to the number of the vehicles and the size of the grid.

Fig. 12 shows the robustness of the AHA with respect to individual vehicle failures. The conducted experiment concerns the performance of the AHA in the presence of inactive vehicles, i.e., the vehicles that do not move and do not interact with the central hub. In the studied case every vehicle becomes inactive once in every five missions and stays inactive for one mission time. The mission time is the average time required for the vehicles to move from a starting cell to a destination cell.

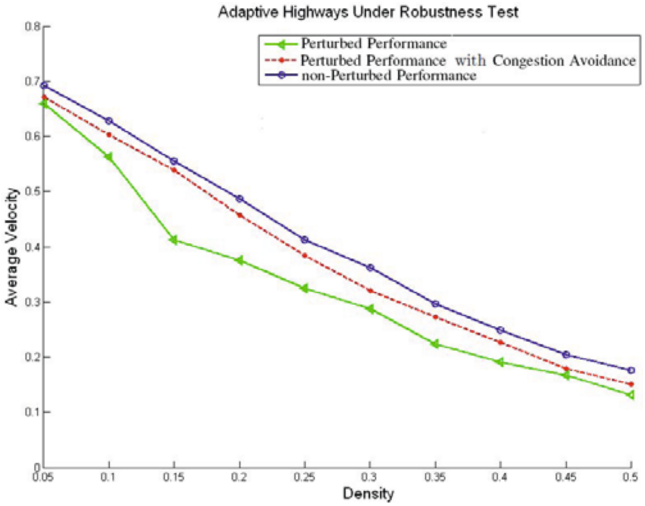


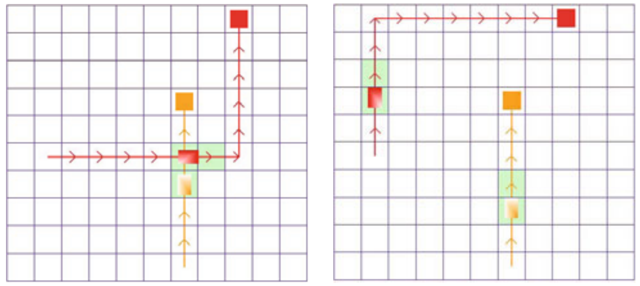
Fig. 12 The robustness test. The congestion avoidance scheme can help retrieve the performance in the presence of vehicle failures. Note that the performance is averaged over the working vehicles, and that the inactive vehicles are not counted. The small loss in performance is inevitable as the congestion is perceived with some delay.

Table 2 Comparing the running time of the algorithms. This experiment lasts 200 seconds of simulated time and is conducted on a laptop with 2.2GHZ Intel CPU and 2GB memory. The provided numbers are the ratio of the simulation time with respect to the running time of each algorithm. The simulation is carried out in a 50 × 50 environment with 625 vehicles, corresponding to a density of 0.25. The experiment suggests that the AHA can be considered as a real time algorithm, although it is slower than its fixed counterpart.

Algorithm	Simulation Time/Real Time
RBA	50
FHA	28
AHA	7

6 Discussion

While the AHA is shown to be promising in the studied context, there is no claim on the global optimality of the algorithm. Simple scenarios can be constructed to show that policies extracted from the algorithm correspond to a local minima of the global optimization problem. Fig.13a shows such an example. Iterative planning is further introduced here to deal with such cases (see Fig.13b). In this case, the vehicles plan in multiple iterations. Initially, the costs (δ_h,δ_s, and δ_0) are set to zero and a trajectory is generated and



(a) Adaptive Highways Algorithm (b) Adaptive Highways Algorithm with iterative planning

Fig. 13 The role of iterative planning. *a) Adaptive Highways Algorithm:* The red vehicle, which has two equally good paths to reach its destination, plans first. The orange vehicle plans next. The solution requires the orange vehicle to wait in the middle of its trajectory until the red vehicle passes. Observe that the fixed planning order in the AHA leads to sub-optimal strategies. *b) Adaptive Highways Algorithm with iterative planning :* Each vehicle plans twice. In the first iteration, all costs are zero. Therefore, each vehicles plans as if there is no other vehicle around, leading to the same trajectories as those in Fig.13a. In the second iteration, the cost for the head-to-side conflict increases to 1. The red vehicle predicts the conflict and chooses the alternative conflict-free trajectory. The orange vehicles keeps its previous trajectory, which is now conflict-free. In this scenario, iterative planning yields the globally optimal solution.

stored in the central hub for each vehicle. The costs are then increased by some constant and this process repeats until the costs reach their optimal value.

Finally, despite its relative complexity compared to other algorithms studied in this work, the AHA is still a real-time algorithm that can be implemented on systems comprised of large number of autonomous vehicles such as the Kiva MFS. The flexibility of the algorithm to work with a scalable number of vehicles can be further exploited to design even larger systems through distribution of computation and communication load. Furthermore, the robustness test suggests that the AHA algorithm, when combined with congestion avoidance, demonstrates a high level of robustness with respect to vehicle failures–a feature that is of high interest in practical applications.

References

1. Frazzoli, E., Dahleh, M.A., Feron, E.: Real-time motion planning for agile autonomous vehicles. AIAA Journal of Guidance, Control, and Dynamics 25(1), 116–129 (2002)
2. Guizzo, E.: Three engineers, hundreds of robots, one warehouse. IEEE Spectrum 45(7), 22–29 (2008)

3. Kator, C.: Staples' robotic retrievers offer new take on break-pack picking. In: Modern Materials Handling (2007)
4. Lavalle, S.M.: Planning Algorithms. Cambridge University Press, Cambridge (2006)
5. LaValle, S.M., Kuffner, J.J.: Randomized kinodynamic planning. In: Proceedings of the 1999 IEEE International Conference on Robotics and Automation, Denver, CO (August 2000)
6. Nagy, T.K., D'Andrea, R., Ganguly, P.: Near-optimal dynamic trajectory generation and control of an omnidirectional vehicle. Robotics and Autonomous Systems 46, 47–64 (2004)
7. Purwin, O., D'Andrea, R.: Trajectory generation and control for four wheeled omnidirectional vehicles. Robotics and Autonomous Systems 54(1), 13–22 (2006), doi:10.1016/j.robot.2005.10.002
8. Purwin, O., D'Andrea, R., Lee, J.W.: Theory and implementation of path planning by negotiation for decentralized agents. Robotics and Autonomous Systems (2007), doi:10.1016/j.robot.2007.09.020.
9. Raffard, R.L., Tomlin, C.J., Boyd, S.P.: Distributed optimization for cooperative agents: Application to Formation Flight. In: 43rd IEEE Conference on Decision and Control, Atlantis, Paradise Island, Bahamas, December 14-17 (2004)
10. Schouwenaars, T., How, J., Feron, E.: Decentralized cooperative trajectory planning of multiple aircraft with hard safety guarantees. In: AIAA Guidance, Navigation, and Control Conference and Exhibit, August 16-19. Providence, Rhode Island (2004)
11. Wurman, P.R., D'Andrea, R., Mountz, M.: Coordinating hundreds of cooperative, autonomous vehicles in warehouses. AI Magazine 29(1), 9–19 (2008)

Environment Modeling for Cooperative Aerial/Ground Robotic Systems

Teresa Vidal, Cyrille Berger, Joan Sola, and Simon Lacroix

Abstract. This paper addresses the cooperative localization and visual mapping problem for multiple aerial and ground robots. We propose the use of heterogeneous visual landmarks, points and line segments. A large-scale SLAM algorithm is generalized to manage multiple robots, in which a global graph maintains the topological relationships between a series of local sub-maps built by the different robots. Only single camera setups are considered: in order to achieve undelayed initialization, we present a novel parametrization for lines based on anchored Plücker coordinates, to which we add extensible endpoints to enhance their representativeness. The built maps combine such lines with 3D points parametrized in inverse-depth. The overall approach is evaluated with real-data taken with a helicopter and a ground rover in an abandoned village.

1 Introduction

In a aerial / ground multi-robot context, as in any multi-robot context, the ability to build and share environment models among the robots is an essential pre-requisite to the development of cooperation schemes. Be it for exploration, surveillance or intervention missions, environment models are indeed necessary to plan and coordinate paths, but also to determine the utility of vantage points, to assess whether robots will be able to communicate or not, and to localize the robots in a common frame. In particular, 3D information on the environment is required: not only the robots evolve in the three dimensions, but the determination of vantage points calls for visibility computations in the 3D space. Also, vision is here the primary

Teresa Vidal · Cyrille Berger · Joan Sola · Simon Lacroix
LAAS/CNRS, University of Toulouse 7, Ave du Colonel Roche,
31077 Toulouse Cedex 4, France
e-mail: Firstname.Name@laas.fr

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 681–696.
springerlink.com

environment sensor to build environment representations: besides the fact that images carry a lot of information on the environment, vision is passive, it has the main advantage to perceive features that are arbitrarily far away, and it is the only sensor that can be exploited on-board micro-drones – that have undoubtedly a promising future.

Approach. Our aerial / ground context requires the resolution of the two following issues: on the one hand the mapping approach must be distributed so as to cope with the communications constraints, and on the other hand the map structure must allow data association and fusion, coping with the fact that the sensors or viewpoints can be very different among the different robots. The contribution of this paper is therefore twofold.

We first propose a distributed mapping approach in which robots build series of sub-maps using a classical EKF-based SLAM paradigm. The overall spatial consistency of the maps among the robots is ensured by an optimization process, that takes into account various inter-robot and absolute localization estimates. This work stems from the work on hierarchical SLAM proposed by Estrada *et al* in [1], which relies on a hierarchical representation of the map: the global level is an adjacency graph, where nodes are local maps (or "sub-maps"), and edges are relative locations between local maps. In a multi-robot context, various events can trigger loop closures and later map merging, namely rendezvous between robots, landmark correspondences ("map matching") and absolute localizations provided by GPS fixes or by matches with an a priori existing map. These events exhibit a cycle on the graph of the map poses, and thus define constraints that allows the system to refine the estimates of the sub-maps origins.

Among the three possibilities to close loops in the overall graph of maps, map matching is the most difficult to achieve when maps have been built by heterogeneous robots, *i.e.* with different kinds of sensors or vantage points. To be able to match maps, we need to represent in maps characteristics of the environment that are invariant with respect to large viewpoint changes. For this purpose, we exploit line segments detected in the images in order to build 3D wireframe maps that can be matched among the robots. This calls for a dedicated implementation of the EKF-based SLAM approach, which relies on the undelayed initialization of anchored Plücker 3D lines.

Outline. Sections 2 and 3 present the localization and mapping approach to deal with multiple robots. The approach is based on the scalable SLAM approaches, known as hybrid or hierarchical, that consider sub-maps and graph levels. Section 4 is devoted to the building of a 3D wireframe model on the basis of line segments detected in monocular imagery. It introduces the anchored Plücker line parametrization to use segments as landmarks in a monocular EKF SLAM approach. Finally, Section 5 present results obtained with data gathered by a helicopter and a ground rover.

2 Hierarchical SLAM

The multiple robot localization and mapping problem is formulated using sub-maps in a similar manner to hierarchical SLAM in [1] or hybrid metric-topological SLAM in [2], where there are two levels: metric (local sub-maps), and topological (global graph).

The topological (global) level represents the relationships $\mathbf{s}_i^j$ between local maps i and j. The metric level contains the local maps, composed of the set of landmarks $\mathbf{m}_i$ and the current robot pose $\mathbf{x}_i$. Each local map stores information in its own *lrf*. At a certain point a new local map is generated with the robot pose acting as the new local reference frame (*lrf*). Thus the robot pose $\mathbf{x}_i$ truly represents the relation between the previous map and the new one, and one can set $\mathbf{s}_i^{i+1} = \mathbf{x}_i$. Other non-correlative relations $\mathbf{s}_i^j$ may be established between maps as we will see. Based on simple frame compositions, information in the world reference frame (*wrf*) is also available for the origins $\mathbf{S}_i$ of each map, and for the map itself $\mathbf{M}_i$ if it is required.

Metric Level. The local or metric level contains the feature-based locally referred stochastic maps, built with the standard EKF-SLAM. The i-th local map is defined by

$$\mathbf{x}_\mathbf{m}^i = \begin{bmatrix} \mathbf{x}_i \\ \mathbf{m}_i \end{bmatrix}, \tag{1}$$

where $\mathbf{x}_i$ is the current pose of the robot, and $\mathbf{m}_i = [\mathbf{l}_1^i \dots \hat{\mathbf{l}}_m^i]^\top$ is the set of m mapped landmarks, both with respect to the i-th *lrf*. EKF-SLAM keeps a Gaussian estimate $\mathbf{x}_\mathbf{m}^i \sim \mathcal{N}\{\hat{\mathbf{x}}_\mathbf{m}^i, \mathbf{P}_\mathbf{m}^i\}$ of this map, namely

$$\hat{\mathbf{x}}_\mathbf{m}^i = \begin{bmatrix} \hat{\mathbf{x}}_i \\ \hat{\mathbf{m}}_i \end{bmatrix}, \qquad \mathbf{P}_\mathbf{m}^i = \begin{bmatrix} \mathbf{P}_{\mathbf{x}_i\mathbf{x}_i} & \mathbf{P}_{\mathbf{x}_i\mathbf{m}_i} \\ (\mathbf{P}_{\mathbf{x}_i\mathbf{m}_i})^\top & \mathbf{P}_{\mathbf{m}_i\mathbf{m}_i} \end{bmatrix}. \tag{2}$$

The maps are built sequentially as mentioned above. Once a threshold is passed, either in number of landmarks or in robot's uncertainty, a new map is created. Let us consider for now that no information is shared between these sub-maps, thus the new map starts in a *lrf* with robot's pose and error covariance equal to zero.

Topological Level. The global or topological level is represented as an adjacency graph in which origins of local maps $\mathbf{S}_i$ in *wrf* are nodes, and the links between them are the relative transformations $\mathbf{s}_i^{i+1}$. Let us define the global level as the Gaussian state $\mathbf{s} \sim \mathcal{N}\{\hat{\mathbf{s}}; \mathbf{P}_s\}$ of relative transformations between local maps, namely:

$$\hat{\mathbf{s}} = \begin{bmatrix} \hat{\mathbf{s}}_0^1 \\ \vdots \\ \hat{\mathbf{s}}_i^{i+1} \end{bmatrix}, \qquad \mathbf{P}_\mathbf{s} = \begin{bmatrix} \mathbf{P}_{\mathbf{s}_0^1} & 0 & 0 \\ 0 & . & 0 \\ 0 & 0 & \mathbf{P}_{\mathbf{s}_i^{i+1}} \end{bmatrix}. \tag{3}$$

The global origins of the maps in the *wrf* are computed as the compounding of the previous global origin with the relative transformation between sub-maps, $\mathbf{S}_{i+1} = \mathbf{S}_i \oplus \mathbf{s}_i^{i+1}$. The current position of the robot in *wrf* is computed as $\mathbf{X}_i = \mathbf{S}_i \oplus \mathbf{x}_i$. Also, the global map can be obtained through,

$$\mathbf{M}_i = \mathbf{S}_i \oplus \mathbf{m}_i \,. \tag{4}$$

Mean and covariances of the Gaussian estimates are obtained by regular linear-Gaussian propagation using the Jacobians of $\oplus$ and $\ominus$.

Considering the relative transformations between local maps as past robot poses, we note that the global level can be viewed as a sparse delayed-state pose-SLAM [3], where local maps are like landmarks hanging from robot poses in *wrf*. The main difference is associated to the fact that the state-space in our case contains relative poses $\mathbf{x}_i$, instead of absolute poses $\mathbf{S}_i$.

Loop Closure. At the global level, a loop closure corresponds to a cycle in the graph, that appears for instance when a relative position estimate between non-consecutive sub-maps is established by a map matching process. Such a cycle defines a constraint between a series of relative transformations:

$$\begin{aligned} \mathbf{h}(\mathbf{s}) &= \mathbf{s}_0^1 \oplus \mathbf{s}_1^2 \cdots \oplus \mathbf{s}_i^0 = 0 \qquad (5)\\ &= \mathbf{S}_i \oplus \mathbf{s}_i^0 = 0 \,. \qquad (6) \end{aligned}$$

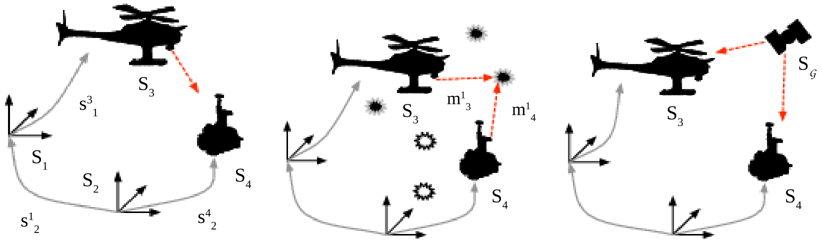
Given that $\mathbf{h}(\mathbf{s})$ is not linear due to the angular terms, the enforcement of this constraint can be formulated as a nonlinear constrained optimization problem. A solution for instance could be based on the Iterative EKF [1]: the part of the state involved in the loop closure at global level then becomes correlated, resulting in a non-sparse covariance matrix $\mathbf{P}_\mathbf{s}$.

Note that for a single robot, local maps are obtained sequentially, hence the relative transformation between the local maps is given by the last robot pose in the current *lrf*, $\mathbf{s}_i^{i+1} = \mathbf{x}_i$.

3 Multiple Robots

A hierarchical/hybrid SLAM approach in the multi-robot case seems a priori straightforward: each robot manages a set of sub-maps and a global graph of poses. But the interests of multi-robot mapping arise of course when the robots exchange mapping or position information, which allows to enhance the spatial consistency and to build up a *multi-robot global graph* of map poses (origins of local sub-maps).

In general when multiple robots are exploring the same area, they meet sooner or later or their maps partially overlap: these events will allow the system to establish connections between robot locations [4]. The events we have identified are: *(i)* robots rendezvous (Figure 1(a)), *(ii)* common information match within sub-maps (Figure 1(b)), and *(iii)* receiving external



(a) Rendezvous event (b) Common landmark event (c) GPS fix

Fig. 1 Loop-closing events for multiple robots.

information that provide absolute localizations (*e.g.* a GPS fix (Figure 1(c)), or feature matches with an existing environment model. The latter is not exactly a multi-robot loop closure, it provides a link between a *lrf* and a global geo-referenced frame: this yields the possibility to establish a link with another robot that has already been absolutely localized once.

All these events create a link between the robots' global levels. Whereas in a single robot case a loop closure only occurs when the robot revisits a previously mapped place, in a multi-robot case these events trigger loop closures: any cycle that appears in the overall graph defined by the concatenation of each robot graph (the *multi-robot graph*) is a loop closure. The compounding of all relative transformations that define a cycle is equal to zero as in Eq. 5, and a batch optimization over the transformations can be performed. Note that to obtain a cycle in the graph defined by the concatenation of two robots global levels, at least two events between these robots are required.

The main advantage to exploit a hierarchical map structure in multi-robot mapping is the low communication bandwidth required among the robots: only the individual graphs need to be exchanged to update the multi-robot graph. We will however see that in the *map-matching* case, the sub-maps that match must also be communicated: but they have naturally already been exchanged to establish the matches. Most importantly is that in the general case, only marginal distributions of each node has to be communicated, as opposed to the full joint distributions of the graph. As we will see later, each robot performs its own optimization based on the minimal cycle.

Robot Rendezvous. The event occurs when a robot observes another robot (partial rendezvous) or when both robots observe each other (full rendezvous). We will focus on the case when the relative transformation between two robots it is fully recovered, from the information obtained through a partial or full rendezvous.

New local maps are created in the instant of the rendezvous, then the current robot poses are promoted to the global level. In this way, the observed transformation $\mathbf{z}$ naturally produces a link between the maps' origins $\mathbf{S}_i$ and $\mathbf{S}_j$ on the global level, thus $\mathbf{z} = \mathbf{s}^i_j$.

Matching common information. There exist at least two different ways to match common information. For example, using information completely independent to the SLAM, *e.g.* image's descriptors matching (SIFT, SURF), image indexing techniques or scan matching (ICP). A common way to produce a map of poses [3] is to find the rotation and translation between two robot poses using one of these techniques, as opposed to tracking features. A second way to match common information is using the available information of the maps (position and uncertainty in global level). This is usually done using the current position of the robot or robots in the *wrf*.

- The "image to image" association produces a link directly between images, that are associated to certain poses $\mathbf{S}_i$ and $\mathbf{S}_j$. Note, the robot poses have to be part of the global graph when this event occurs.
  This is a simple, but effective manner to obtain the relationship between two robots, or even to close a loop with a single robot. Note that the observations are independent from the previous mapped information. As in the rendezvous case, this event produces the missing link $\mathbf{z} = \mathbf{s}_j^i$. In practical terms, image to image matching is equivalently to rendezvous.
- The "map to map" kind of data association requires both local maps to be transformed to a common frame, *e.g.* promoted to the global level as $\mathbf{M}_i$ and $\mathbf{M}_j$ using Eq. 4. In other words, the matching process happens in a common frame, being the *wrf* or one of the *lrf*. The disadvantage of this method is that in the worst case the absolute position of the two maps must be computed. Moreover, once the maps are matched, they have to be fused into a single one, otherwise it could lead into inconsistencies when merging all the maps.

Absolute Localization. In an aerial/ground context, it is reasonable to assume that both kind of robots receive GPS fixes from time to time. The relative transformation provided by a GPS fix for vehicle i is simply $\mathbf{s}_i^{\mathcal{G}}$, where $\mathcal{G}$ is the geo-referenced frame. Such information provides a link between a *lrf* and a global geo-referenced frame, and generates a loop at the graph level for an individual robot.

Impact on the sub-maps. From the point of view of a hierarchical SLAM formulation, the hierarchical nature of this model manifests itself in the ability to apply a loop-consistency constraint to a subset of local transformations (*e.g.* a single loop) without taking into account the local sub-maps. Particularly, when no information is shared between sub-maps, which is the case between sub-maps built by different robots, but an approximation for the sub-maps that are built by the same robot, the origin of the local sub-map is the only state that changes after the constraint is applied. It can be easily shown that $\mathbf{m}_{i-1} \perp \mathbf{m}_i \mid \mathbf{x}_{i-1}$, *i.e.*, given the relative transformations, the consecutive local sub-maps are independent. Notice that the global poses $\mathbf{S}$ are d-separated of all possible paths between any pair of sub-maps $\mathbf{m}$ or even $\mathbf{M}$. Note that the approximation can be palliated using conditionally independent local maps as proposed in [5].

Also, in the multi-robot case it can happen that two different events create a link on the same node, *i.e.* if a *map-matching* is established after a rendezvous. To avoid this problem, new sub-maps are started *after an event occurs*. In the case of receiving GPS fixes once in a while, the fact of starting a new local map at the time t when the fix is received, removes the dynamics aspects of the internal local maps setting a fix pose at global level.

Similarly, to avoid counting information twice if one eventually wants to merge all the sub-maps, after a map-matching event both sub-maps should be fused into a single sub map. This has the disadvantage that the sub-maps must be shared among the two robots, but on the one hand this is a prerequisite for at least one robot to establish the matches, and on the other hand such events will occur when the robots are within communication range.

4 Line Segments for Monocular EKF-SLAM

We explore the possibility of using linear landmarks or *line segments*, which provide an improved semantic over points: lines inherently contain the notions of connectivity and boundary, which open the door to potential automatic interpretations of the environment, both at the metrical and topological levels. A 3D model based on lines, akin to a wireframe model, can further allow the possibility to build higher level entities (planes, closed regions, objects).

Plucker lines (PL). Plücker lines have been used in major vision works with straight 3D lines [6, 7]. These works, and other ones referenced therein, are based on Structure From Motion approaches. We showed in [8], drawing on previous work in [9], the way to employ Plücker lines to achieve undelayed initialization of lines in monocular SLAM.

The Plücker coordinates for a line consist of a homogeneous 6-vector in projective space, $\mathcal{L} \in \mathbb{P}^5$. In this vector, one can identify two sub-vectors[1], $\mathcal{L} = (\mathbf{n} : \mathbf{v})$, with $\{\mathbf{n}, \mathbf{v}\} \in \mathbb{R}^3$, with which an intuitive geometrical interpretation of the line in 3D Euclidean space is possible (Fig. 2):

- The sub-vector $\mathbf{n}$ is a vector normal to the plane π supporting the line $\mathcal{L}$ and the origin $\mathcal{O}$.
- The sub-vector $\mathbf{v}$ is a director vector of the line.
- The distance from the line to the origin is given by $d = \|\mathbf{n}\|/\|\mathbf{v}\|$.

The two most remarkable properties of the Plücker line are its linear transformation and projection equations and the inverse-depth behavior of the sub-vector $\mathbf{v}$, something that will allow us to design appropriate initialization methods for EKF. For details on the use of Plücker lines in monocular EKF-SLAM see [8].

[1] We use a colon (:) to separate the non-homogeneous and homogeneous parts in projective space.

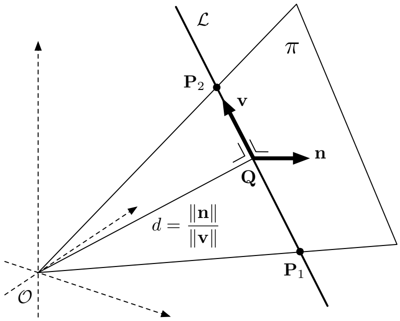


Fig. 2 Geometrical representation of the Plücker coordinates and sub-vectors $\mathbf{n}$ and $\mathbf{v}$. The line $\mathcal{L}$ and the origin $\mathcal{O}$ define the support plane π. The line's sub-vector $\mathbf{n} \in \mathbb{R}^3$ is orthogonal to π. The sub-vector $\mathbf{v} \in \mathbb{R}^3$ is a director vector of the line, and lies on π. This implies $\mathbf{n} \perp \mathbf{v}$. The closest point to $\mathcal{O}$ is $\mathbf{Q} = (\mathbf{v} \times \mathbf{n} : \mathbf{v}^\top \mathbf{v}) \in \mathbb{P}^3$. The distance from $\mathcal{L}$ to $\mathcal{O}$ is $d = \|\mathbf{n}\|/\|\mathbf{v}\|$, showing that $\mathbf{v}$ acts as the homogeneous part of $\mathcal{L}$, thus exhibiting inverse-depth properties.

Anchored Plücker lines (APL). Now, we add an anchor to the parametrization to improve linearity, as it is done for points in the inverse-depth parametrization [10]. Anchoring the Plücker line means referring it to a point $\mathbf{p}_0$ in 3D space different from the origin (Fig. 3). The anchor point $\mathbf{p}_0$ is chosen to be the optical center at initialization time. The effect of such anchoring is that, on subsequent EKF updates, only the accumulated errors from the anchor $\mathbf{p}_0$ to the current camera position T are considered, in contrast with regular Plücker lines where the error accounts for the absolute motion of the sensor from the origin of coordinates.

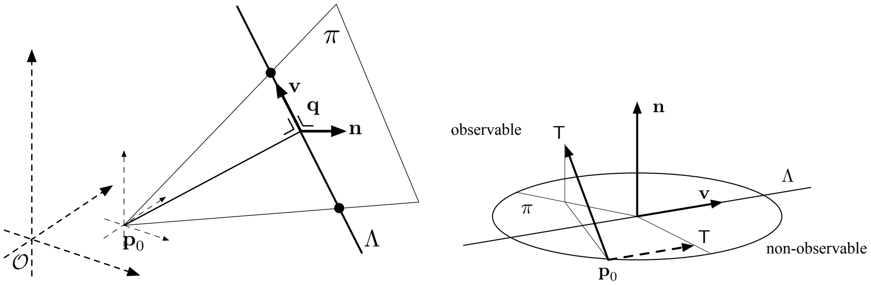
The *anchored Plücker line* (APL, Fig. 3) is then the 9-vector:

$$\Lambda = \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{n} \\ \mathbf{v} \end{bmatrix} \in \mathbb{R}^9 \tag{7}$$

Transformation and projection expressions are as follows:

Frame transformation: Given a (camera) reference frame $\mathcal{C}$ specified by a rotation R and a translation T, an anchored Plücker line Λ in global frame is obtained from a line $\Lambda^{\mathcal{C}}$ in frame $\mathcal{C}$ with the affine transformation

$$\Lambda = \begin{bmatrix} \mathsf{R} & 0 & 0 \\ 0 & \mathsf{R} & 0 \\ 0 & 0 & \mathsf{R} \end{bmatrix} \cdot \Lambda^{\mathcal{C}} + \begin{bmatrix} \mathsf{T} \\ 0 \\ 0 \end{bmatrix} . \tag{8}$$



(a) Geometrical interpretation of anchor $\mathbf{p}_0$ and sub-vectors $\mathbf{n}$ and $\mathbf{v}$. The closest point to the anchor is $\mathbf{q} = \mathbf{p}_0 + (\mathbf{v} \times \mathbf{n})/(\mathbf{v}^\top \mathbf{v})$

(b) Observability conditions. The line is 3D-observable if and only if the motion vector $(\mathsf{T} - \mathbf{p}_0)$ does not lie on the line's support plane π.

Fig. 3 The anchored Plücker line.

The inverse transformation is performed with

$$\Lambda^{\mathcal{C}} = \begin{bmatrix} \mathsf{R} & 0 & 0 \\ 0 & \mathsf{R} & 0 \\ 0 & 0 & \mathsf{R} \end{bmatrix}^\top \cdot \left(\Lambda - \begin{bmatrix} \mathsf{T} \\ 0 \\ 0 \end{bmatrix} \right). \tag{9}$$

Un-anchoring: given an anchored Plücker line $\Lambda = (\mathbf{p}_0, \mathbf{n} : \mathbf{v})$, its corresponding (un-anchored) Plücker line $\mathcal{L}$ is computed with

$$\mathcal{L} = \begin{bmatrix} \mathbf{n} + \mathbf{p}_0 \times \mathbf{v} \\ \mathbf{v} \end{bmatrix} \tag{10}$$

Pin-hole projection: Projection is better expressed for regular Plücker lines. Given a perspective camera defined by the intrinsic parameters $\mathbf{k} = (u_0, v_0, \alpha_u, \alpha_v)$, a Plücker line $\mathcal{L}^{\mathcal{C}} = (\mathbf{n}^{\mathcal{C}} : \mathbf{v}^{\mathcal{C}})$, expressed in the camera's coordinate system, projects into a homogeneous line $\mathbf{l} \in \mathbb{P}^2$ in the image plane with the linear expression [6, 11]

$$\mathbf{l} = \mathcal{K} \cdot \mathbf{n}^{\mathcal{C}} \triangleq \begin{bmatrix} \alpha_v & 0 & 0 \\ 0 & \alpha_u & 0 \\ -\alpha_v u_0 & -\alpha_u v_0 & \alpha_u \alpha_v \end{bmatrix} \cdot \mathbf{n}^{\mathcal{C}} \in \mathbb{P}^2. \tag{11}$$

where $\mathcal{K}$ is called the *Plücker intrinsic matrix.*

Transformation and projection: Transformation and projection are accomplished with a transformation to the camera frame (9), un-anchoring (10), and projection (11). This can be composed in one single expression with:

$$\mathbf{l} = \mathcal{K} \cdot \mathsf{R}^\top \cdot (\mathbf{n} - (\mathsf{T} - \mathbf{p}_0) \times \mathbf{v}) \in \mathbb{P}^2, \tag{12}$$

in which we will notice:

- The linear behavior with respect to $\mathbf{n}$.
- For accurate estimates of the camera motion $(\mathsf{T} - \mathbf{p}_0)$, which is true for observations shortly after initialization, the linear behavior with respect to $\mathbf{v}$, which additionally exhibits inverse-depth behavior.

Expression (12) gives us the means to analyze line observability as a function of the camera motion (Fig. 3(b)). Let $(\mathsf{T} - \mathbf{p}_0)$ be the camera motion since initialization time. Because the projected line $\mathbf{l}$ is expressed in projective space, any vector $\mathbf{l}' = \alpha\mathbf{l}$, with $\alpha \in \mathbb{R}$, represents the same line and the line sub-vector $\mathbf{v}$ is only observable if the vector $(\mathsf{T} - \mathbf{p}_0) \times \mathbf{v}$ is not proportional to $\mathbf{n}$. If we remind the Plücker constraint stating that $\mathbf{v} \perp \mathbf{n}$, this resumes to a motion $(\mathsf{T} - \mathbf{p}_0)$ that does not belong to the plane π. As the anchor $\mathbf{p}_0$ belongs to this plane, we conclude that the new camera position T must escape the plane π in order to fully observe the line.

Segment endpoints. The line's endpoints in 3D space are maintained out of the filter via two abscissas defined in the local 1D reference frame of the line, whose origin is at the point $\mathbf{q} = \mathbf{p}_0 + \frac{\mathbf{v}\times\mathbf{n}}{\mathbf{v}^\top\mathbf{v}}$, the closest point to the anchor (see Fig. 3(a)). Given the line $\Lambda = (\mathbf{p}_0, \mathbf{n} : \mathbf{v})$ and abscissas $\{t_1, t_2\}$, the 3D Euclidean endpoints are obtained with

$$\mathbf{p}_i = \mathbf{q} + t_i \cdot \frac{\mathbf{v}}{\|\mathbf{v}\|} \in \mathbb{E}^3, \ i \in \{1, 2\}. \tag{13}$$

Back-projection of an APL. APL back-projection consists in defining a Plücker line $\mathcal{L}$ from a segment observation $\mathbf{l}$, and anchoring it at the camera position T to obtain an APL Λ. These operations are detailed below.

Back-projection of a Plücker line: In the camera frame, the Plücker sub-vector $\mathbf{n}^{\mathcal{C}}$ resulting from the observation $\mathbf{l}$ is simply

$$\mathbf{n}^{\mathcal{C}} = \mathcal{K}^{-1}\mathbf{l}. \tag{14}$$

The sub-vector $\mathbf{v}^{\mathcal{C}}$ is not measured and must be obtained by injecting prior information. We give here the formulation and refer the reader to [8] where full explanations and justifications are provided. The sub-vector $\mathbf{v}^{\mathcal{C}}$ is obtained with

$$\mathbf{v}^{\mathcal{C}} = \mathbf{E}\beta, \tag{15}$$

where $\mathbf{E} \in \mathbb{R}^{3\times2}$ is a matrix transforming vectors $\beta \in \mathbb{R}^2$ in the Cartesian plane into the plane in $\mathbb{R}^3$ supporting the line, defined by the optical center and $\mathbf{n}^{\mathcal{C}}$. It is constructed as a base spanning the plane orthogonal to $\mathbf{n}^{\mathcal{C}}$, *i.e.*, the two non-measured DOFs,

$$\mathbf{E} = \begin{bmatrix}\mathbf{e}_1 \ \mathbf{e}_2\end{bmatrix}, \quad \mathbf{n}^{\mathcal{C}} \perp \mathbf{e}_1 \perp \mathbf{e}_2 \perp \mathbf{n}^{\mathcal{C}}, \tag{16}$$

the base vectors $\mathbf{e}_i$ being chosen so that $\mathbf{e}_1$ is parallel to the image plane,

$$\mathbf{e}_1 = \frac{\left[n_2^{\mathcal{C}} \; -n_1^{\mathcal{C}} \; 0\right]^\top}{\sqrt{(n_1^{\mathcal{C}})^2 + (n_2^{\mathcal{C}})^2}} \quad \text{and} \quad \mathbf{e}_2 = \frac{\mathbf{n}^{\mathcal{C}}}{\|\mathbf{n}^{\mathcal{C}}\|} \times \mathbf{e}_1. \tag{17}$$

The vector $\beta = (\beta_1, \beta_2)$ must be provided as prior. To help selecting an appropriate one, we give here some intuitive notions about β:

- A vector $\beta = (1, 0)$ is a line parallel to the image plane at a distance $d = \|\mathbf{n}^{\mathcal{C}}\|$ from the optical center.
- A vector $\beta = (0, 1)$ is a line perpendicular to the detected segment $\mathbf{l}$ in the image.
- The distance from the optical center to the line is given by $d = \|\mathbf{n}^{\mathcal{C}}\| / \|\beta\|$.

Anchoring: This step is trivial as we have interest in making the anchor $\mathbf{p}_0$ coincide with the current camera position, which is the origin when we are in camera frame,

$$\varLambda^{\mathcal{C}} = \begin{bmatrix} \mathbf{0} \\ \mathbf{n}^{\mathcal{C}} \\ \mathbf{v}^{\mathcal{C}} \end{bmatrix}. \tag{18}$$

Back-projection and transformation: The operations above plus the transformation to the global frame ([8]) can be composed and written as a single-step function of R, T, $\mathbf{l}$ and β,

$$\varLambda = \begin{bmatrix} \mathbf{p}_0 \\ \mathbf{n} \\ \mathbf{v} \end{bmatrix} = \begin{bmatrix} \mathsf{T} \\ \mathsf{R}\mathcal{K}^{-1}\mathbf{l} \\ \mathsf{R}\mathbf{E}\beta \end{bmatrix}. \tag{19}$$

ALP initialization and update in EKF-SLAM. We now have expressed all the basic relations to deal with APL, that are useful to exploit them in an EKF framework. The line initialization and update equations are not depicted here, details can be found in [8].

5 Experimental Results

The environment is "semi-structured", in the sense that it does not contain as many buildings as an urban area – and the building themselves do not contain many straight lines or perfect planar areas, see Figures 4.

The ground robot Dala is an iRobot's ATRV platform, equipped with a calibrated stereo-vision bench made of two 1024 × 768 cameras with a baseline of 0.35m. The helicopter Ressac is controlled by algorithms developed at Onera [12], and is also equipped with a calibrated stereo vision bench made of two 1024 × 768 cameras, with a 0.9m baseline (Figure 5).

Involved processes. Unfortunately, because of engineering issues encountered during the data collection, no inertial or odometric motion estimates

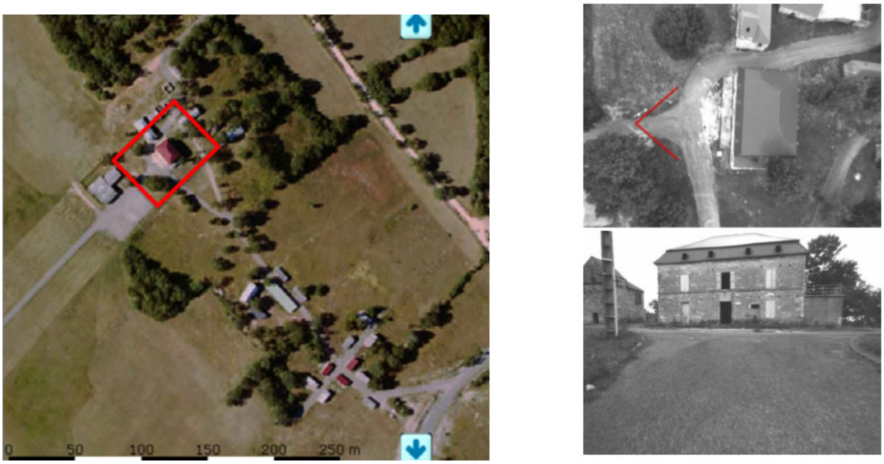
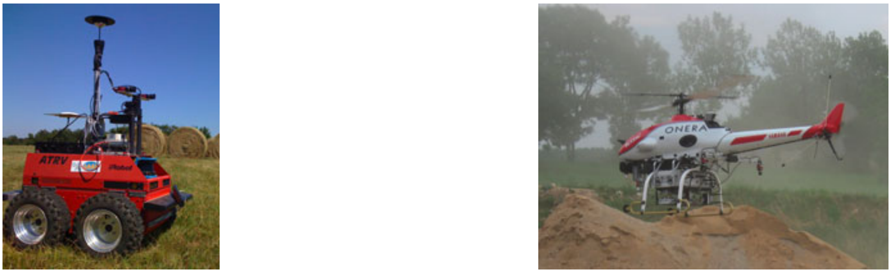


Fig. 4 Aerial view of the experiment area (obtained on www.geoportail.fr). The robot Dala evolves in the north-west group of buildings, while the helicopter Ressac is flying along a swathing pattern oriented diagonally between the north-west and south-east groups of buildings, at an elevation of about 40m. The red rectangle approximately represents the field of view of the image acquired by Ressac shown on top-right of the figure. On this latter image, the red angular sector shows approximately the field of view of Dala when taking the bottom-right image.



(a) Dala

(b) Ressac

Fig. 5 Ground and aerial robots used for the experimental validation.

are available[2]. As a consequence, we use a visual odometry approach based on stereo vision for the motion prediction steps of the EKF SLAM algorithms – all the results presented hereafter have therefore been obtained using *exclusively* visual data.

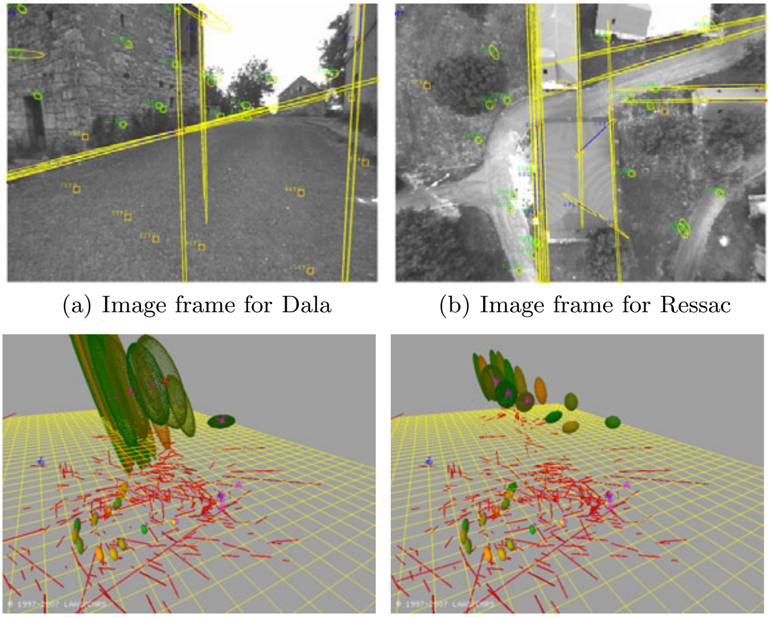
The SLAM algorithms integrate two types of observations from only one camera; image points (parametrized as inverse-depth points) and image line segments (parametrized as anchored Plücker line segments). At each image

[2] GPS ground truth could neither be recorded.

acquisition, point observations are firstly processed: the resulting updated motion estimate is exploited by the line segment tracker, and line landmarks observations are then processed. A heuristic is used to select the points that will be used as landmarks: the image is regularly partitioned in 3×3 regions, in which one ensures that at least 2 landmarks are being tracked. As for the lines, all the ones whose length is grater than 60 pixels are retained as landmarks.

Point landmarks are Harris interest points, that are matched from one view to the other with the group based matching procedure described in [13]. We use different initialization parameters for inverse depth point parametrization with Dala and Ressac. Dala's parameters are $\rho_{init} = 0.1$ m^{-1} and $\sigma_\rho = 0.2$ m^{-1}, while Ressac's parameters are $\rho_{init} = 0.025$ m^{-1} and $\sigma_\rho = 0.0125$ m^{-1} (the points are initialized at 40m, which is its the helicopter average elevation over the terrain.

We use the algorithms presented in [14] to detect and track segments. For the estimation part, the a priori parameters used in the experiment for the APL are $\beta = (0.025, 0)$, $\sigma_{\beta_1} = 0.025$ and $\sigma_{\beta_2} = 0.0375$ for both robots. The



(a) Image frame for Dala

(b) Image frame for Ressac

(c) Before an event

(d) After an event

Fig. 6 Top: Image frames from both robots before the event. Green squares represent interest point currently considered as landmarks, yellow squares represent interest points just initialized as landmarks. The line segments are in blue, with endpoints in red. Yellow ellipses are the uncertainty in the image view. Bottom: Event effect in the global map, the sub maps origins expressed in the *wrf* are the large ellipsoids – only 3D line segments landmarks are shown here.

prediction of the line segment position in the image, required for the segment tracker, is done using the projection of the 3D line segment into the image frame.

New local maps are created when 100 landmarks (combining points and line segments) are in the map. Immediately after, the current robot's pose is the new relative transformation in the global graph.

Enforcing loop closures. In the experiments, we have two types of events: rendezvous and image to image matching.

- The rendezvous is emulated using matches of interest points perceived by the two robots using their current image frames. The 3D coordinates of the points in each robot frame are obtained by stereo vision: as a result, the point matches yields an estimate of the relative robot position.
- The image matching event recovers the relative transformation between the current robot's pose and a past pose from a different robot (origin of a local sub-map), using also matches of interest points between their respective frames.

Figure 7 shows results obtained by the integration of events between Dala and Ressac. Dala starts at the entry of the north-west group of buildings, with no uncertainty in its local map, but also in the *wrf*: the first Dala sub-map is

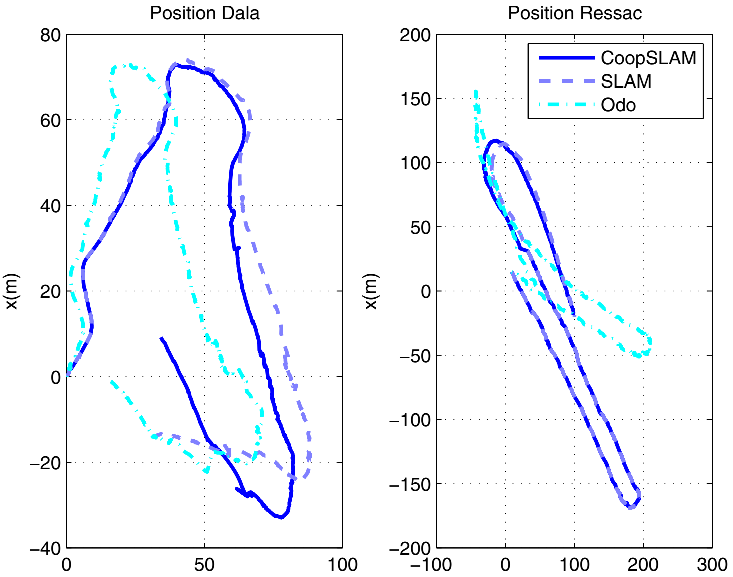


Fig. 7 Comparative trajectory plots: visual odometry in dash-dot line, open loop run in dashed line and cooperative run for Dala (left) and Ressac (right).

the origin of the world. Ressac starts above Dala, and heads towards south-east. A first rendezvous event occurs immediately after start, and Ressac is localized in Dala's reference frame.

A second event occurs after Ressac comes back from the south-east village, passing above a place previously visited by Dala. The effects of the image to image matching event are shown in Figure 6. The figure also shows the image frames that were evaluated for the matching: new local maps are initiated afterward for both robots. Note that Ressac's uncertainty in height is pretty large, especially before the second event: the visual motion estimates are indeed not very precise in the vertical direction, because Ressac stereo baseline is small with respect to the depth of the perceived points – and the integration of points and lines in the sub-maps does not reduces much the elevation estimates. However, after the image to image matching event, the elevation of Ressac and the origins of all the built maps are strongly corrected.

6 Conclusion

We have explored the use of a multiple local maps technique for multi-robots according to a hierarchical SLAM approach, in which loop closures are integrated through an optimization process (an Iterative EKF). Loop closures are triggered using multi- or single-robot events such as finding information correspondences between unconnected local maps, rendezvous between two robots or GPS fixes. Therefore, the mapping problem is relegated within the local sub-maps, and is decorrelated from the global localization problem, making our approach akin to a cooperative localization approach. The approach is distributed, and the graph level is the sole information that must be exchanged between the robots. It is well suited to a multi-robot context, and it can in particular handle *all* the possible localization means, from odometry to absolute localization with respect to an initial model.

In order to build more meaningful landmark maps and to be able to match data acquired from very different vantage points (or even different sensors), we have proposed to use line segments to build a wireframe model. An important contribution of this paper is the new line segment parametrization for undelayed initialization. We add an anchor to our previously proposed parametrization [8] to improve the linearity, exactly as it is done with inverse-depth points. Thanks to the cross-correlations stored in the EKF, the anchor allows the filter to account for accumulated errors only from the anchor to the current position, not from the origin of coordinates. Ongoing work is a more detailed analysis of different line parametrization in view of their linearity behavior.

Heterogeneous visual landmarks are proposed to map semi-structured environments. We combine inverse-depth points and anchored Plücker line segments. Our experiments show that inverse-depth points, that are numerous in the scene but not very robust to large viewpoint variations, play a crucial role for robot localization, while Plücker line segments, that are seldom in semi-structured environments but allow detection and matching from disparate viewpoints, are well suited to build a 3D model that exhibit some of the environment structure.

Acknowledgments. This work has been partially founded by the Action project (action.onera.fr). Cyrille Berger is founded by a Cifre contract with Thales Optronics. We would like to especially thank the Onera "Ressac team" for all their help for the experiments.

References

1. Estrada, C., Neira, J., Tardós, J.D.: Hierarchical SLAM: Real-time accurate mapping of large environments. IEEE Trans. Robot. 21(4), 588–596 (2005)
2. Blanco, J.-L., González, J., Fernández-Madrigal, J.-A.: Subjective local maps for hybrid metric-topological SLAM. Robotics and Autonomous Systems 57(1), 64–74 (2009)
3. Eustice, R.M., Singh, H., Leonard, J.J.: Exactly sparse delayed-state filters for view-based SLAM. IEEE Trans. Robot. 22(6), 1100–1114 (2006)
4. Vidal-Calleja, T., Berger, C., Lacroix, S.: Even-driven loop closure in multi-robot mapping. In: IEEE Int. Conf. on Intelligent Robots and Systems, Saint Louis, USA (2009) (to appear)
5. Piniés, P., Tardós, J.D.: Scalable slam building conditionally independent local maps. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, San Diego, CA, October 29-November 2 (2007)
6. Bartoli, A., Sturm, P.: The 3D line motion matrix and alilgnment of line reconstructions. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 287–292 (2001)
7. Kamgar-Parsi, B., Kamgar-Parsi, B.: Algorithms for matching 3D line sets. IEEE Transactions on Pattern Analysis and Machine Intelligence 26(5), 582–593 (2004)
8. Solà, J., Vidal-Calleja, T., Devy, M.: Undelayed initialization of line segments in monocular SLAM. In: IEEE Int. Conf. on Intelligent Robots and Systems, Saint Louis, USA (2009) (to appear)
9. Lemaire, T., Lacroix, S.: Monocular-vision based SLAM using line segments. In: IEEE International Conference on Robotics and Automation, Roma (Italy) (April 2007)
10. Civera, J., Davison, A.J., Montiel, J.M.M.: Inverse depth parametrization for monocular SLAM. IEEE Trans. on Robotics 24(5) (2008)
11. Solà, J., Monin, A., Devy, M., Vidal-Calleja, T.: Fusing monocular information in multi-camera SLAM. IEEE Trans. on Robotics 24(5), 958–968 (2008)
12. Fabiani, P., Fuertes, V., Piquereau, A., Mampey, R., Teichteil-Königsbuch, F.: Autonomous flight and navigation of vtol uavs: from autonomy demonstrations to out-of-sight flights. Aerospace Science and Technology 11(2-3), 183–193 (2007)
13. Lemaire, T., Berger, C., Jung, I.-K., Lacroix, S.: Vision-based slam: Stereo and monocular approaches. IJCV - IJRR (2006)
14. Berger, C., Lacroix, S.: DSeg: Direct line segments detection. Technical report, LAAS/CNRS (2009)

Energy-Efficient Data Collection from Wireless Nodes Using Mobile Robots

Onur Tekdas, Nikhil Karnad, and Volkan Isler

Abstract. This work focuses on systems where mobile robots periodically collect data from (static) wireless sensor network nodes. Suppose we are given approximate locations of the static nodes, and an order with which the robot will visit these nodes. We present solutions to the following problems. (i) From the static node's perspective: given the stochastic nature of the robot's arrival, what is an energy-efficient strategy to send beacon messages? Such a strategy must simultaneously minimize the robot's waiting time and the number of beacon messages. (ii) From the robot's perspective: given the stochastic nature of the wireless link quality, what is an energy-efficient motion strategy to find a good pose (location and orientation) from where the data can be downloaded efficiently? The robot must be able to find such a location quickly but without taking too many measurements so as to conserve the static node's energy.

For the first problem, we present an optimal algorithm based on dynamic programming. For the second problem, we present an efficient, data-driven heuristic based on experiments. Finally, we present a system implementation for an indoor data collection application, and validate our results on this system.

1 Introduction

Wireless sensor networks (WSNs) are finding increasing use in crucial applications such as environmental monitoring, factory automation and security. However, deploying such sensor systems and gathering the data collected by the sensors still remains a challenge. This is especially true when the target application requires collecting data over a large area such as a farm, a forest or a large warehouse.

In cases where the application requires a dense sampling of the environment, the data can be gathered by forming a wireless network where sensor nodes also act as

Onur Tekdas · Nikhil Karnad · Volkan Isler
Department of Computer Science and Engineering, University of Minnesota,
Minneapolis, MN - 55414, USA

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 697–712.
springerlink.com

relays. In certain applications, the underlying environment is very large and sampling locations are apart from each other. For example, in some habitat monitoring applications, sensors are deployed to collect humidity and temperature data across the entire habitat of species [11]. In building automation, a WSN can be rapidly deployed to collect data (temperature, light) in a few key locations in a large warehouse.

In these applications, a deployment that is dense enough to form a connected network can be costly and difficult to maintain. It is also very expensive to install a wired network just for data acquisition. Often, instead of deploying a dense, connected network, data is manually downloaded from the motes.

An alternative to collecting data manually is to use mobile robots. The last decade witnessed significant developments in mobile robot navigation. It is now feasible to develop systems where robots periodically visit sensor nodes and gather the data collected by them. Typically mobile robots are capable of carrying large batteries and can recharge themselves autonomously. Therefore, the life-time of the static nodes is usually the most crucial factor in determining the overall life-time of the system. In this work, we focus on such systems and address two problems that affect the life time of the automation system.

Beacon Scheduling. In most applications, a robot's arrival to a sensor's vicinity will be a stochastic process due to uncertainties in the navigation times of robots and changes in their trajectories. Therefore, sensor nodes must send beacon messages and listen to the channel for presence of robots. If this is done very frequently, energy consumed in beaconing can reduce the life-time of the network. In Section 3, we address the problem of scheduling beacon messages and present optimal beaconing algorithms.

Data Download. The quality of the communication link between a robot and a sensor can affect the time to download the data (and hence the energy consumption) drastically. If the robot can utilize its mobility and find a "good" location to download data, this can yield significant energy savings. This statement is further justified in Section 4 where we address this search problem and present a data-driven strategy to find a good download location. In indoor environments where the behavior of the signal is unpredictable due to multipath effects and the dynamic nature of the environment, it is easy to see that there is no online algorithm with provable performance guarantees[1]. Therefore, we present a heuristic strategy based on extensive experiments we performed to understand the effect of robot's location and orientation on the signal quality.

In Section 5, we present the details of a system implementation that utilizes robots for gathering data, and demonstrate the utility of our algorithms with experiments run on this system. We start with an overview of related work.

[1] For any given deterministic strategy, an adversary can pick the "good" location to be the last location visited by the strategy.

2 Related Work

Mobility in collecting sensor data is extensively studied. For example, Shah et al presented an architecture that uses mobile entities in the environment for data delivery [13]. In most of the related literature, mobility is treated as an uncontrolled process. More recently, researchers proposed architectures that exploit controlled mobility [14, 6, 3, 18, 15]. A recent review on the state of the art in exploiting sink mobility can be found in [9].

In existing approaches, the robot's trajectory is either given or computed in advance. This constraint is relaxed in [6] where the robot's velocity is modified (along a fixed path) to improve transmission quality. In more recent work, a system where robots efficiently collect data from static sensors have been presented [17]. This work demonstrates the energy savings attained by using mobile robots over using static relay nodes. In the present work, we take a further step and present a strategy that fully utilizes the controllability of the robot's mobility (position and orientation) to find good download locations in an online fashion. The strategy does not make strong assumptions about the wireless signal. We demonstrate its utility with real implementations.

We also study the interactions between the robot and static nodes during the discovery phase. This is related to sleep scheduling which is usually studied as a topology management problem [2, 12]. In this work, we focus on a special case where the arrival of the robot is given as a probability distribution, and present an algorithm to compute the optimal sleep schedule which simultaneously minimizes the number of beacon messages and the robot's wait time. In a real system implementation, we show how such distributions can be learned over time.

The interactions between robots and a static sensor network have also been addressed in the robotics literature for network repair [4], connectivity [1] and navigation [7] problems. In this work, we model the interaction between the robot and the nodes at a lower level and address signal-strength and node scheduling issues. Such approaches have recently started appearing in the robotics literature, mostly in tracking [8, 10] and connectivity maintenance [5, 16] applications.

3 Optimal Beacon Scheduling

There are many sources of uncertainty that causes variability in the robot's arrival time at each sensor. For instance, the robot may spend uncertain amounts of time to locate the motes and download data from them. It may take alternate routes due to obstacles, and spend time to compensate for uncertainties in its own actuators and sensors. Therefore, we model the robot's arrival as a stochastic process. Since the robot's arrival time is uncertain, each mote must periodically send beacon messages and execute a receiver check to establish connection with the robot. By adaptively scheduling beacons it is possible to keep the duty cycle of the mote as low as possible and conserve energy.

3.1 Formulation

Consider a system in which m sensor motes have been statically deployed. The approximate location of each mote is known with respect to a fixed origin: the home base from where the robot starts its journey. An ordered sequence of labels $S = \{s_1, s_2, \ldots, s_m\}$ is assigned to the sensor motes a priori.

If a mote knows the exact time (or the interval) the robot will be within its communication range, then it could establish communication with a single beacon. In the presence of uncertainties, one alternative is to send beacon messages infrequently and have the robot wait in the mote's vicinity until it hears a beacon. However, this has the undesired effect of making the overall data collection time large which, in turn, decreases the overall system performance. Therefore an efficient beaconing strategy is desirable.

Let us call the time interval between consecutive robot arrivals at sensor mote s as the *interarrival time*. Due to the stochastic nature, we represent the robot's interarrival time at sensor s as a probability distribution. This distribution can be either guessed (e.g. by using a Gaussian which represents the expected arrival time and uncertainty), or can be adaptively learned by keeping the history. In this work, we assume that the distribution is given and omit the details of how it can be learned.

We assume that the time between consecutive robot arrivals at a mote is bounded from above. In general, there can also be a lower bound that is non-zero, because the robot's velocity and the length of the complete path ensure that the robot takes a non-zero amount of time to revisit the same sensor.

We now solve the following problem: Given a robot interarrival probability distribution at a mote, find a beacon schedule for that mote, using the least number of beacons possible, such that the expected waiting time of the robot between its arrival at that mote, and receiving a beacon message is bounded from above by a predetermined value T_w. In the following section, we show how such a beacon schedule can be computed.

3.2 Optimal Solution

We now focus on the scenario where the mobile robot visits a sensor s in rounds. As explained in the formulation, we assume that for any round, the interarrival time is bounded: there is a time before which the robot cannot be present in communication range of the mote, and after which the robot is guaranteed to have visited the mote. Denote this interval as T. We model time as discrete by dividing the interval T into n time instants spaced equally by Δt units, $T = \{t_1, \ldots, t_n\}$. A beacon can be scheduled at any t_i $(1 \leq i \leq n)$.

The robot's arrival during the interval T is given as a probability distribution over the time instants in T. Let $p(t_i)$ denote the probability that the robot arrives at time t_i. In the case that the probability is continuous over time, one can interpret t_i to be the end point of a time interval $[t_{i-1}, t_i]$, with the beacon being placed at the end of that interval and $p(t_i)$ being the aggregate probability for that interval.

Let $B = \{b_1, \ldots, b_k = t_n\}$ be a beacon schedule. The values b_i denote the times in the interval T at which beacons are scheduled. Given the arrival distribution p, the robot's expected waiting time $ET(B,p)$ at sensor mote s is given by

$$ET(B,p) = \sum_{i=0}^{k-1} \sum_{t_j=b_i+\Delta t}^{b_{i+1}} p(t_j)(b_{i+1} - t_j) \tag{1}$$

In Equation 1, b_0 is the start time of T.

Consider beacon b_i. For any robot arrival at time $t_j > b_i$, the expression $p(t_j)(b_{i+1} - t_j)$ is the expected waiting time for the robot, until beacon b_{i+1} is heard. Thus, the inner summation in (1) gives the robot waiting times between beacons b_i and b_{i+1}. The outer summation accumulates the expected waiting times over the entire schedule.

We would like to simultaneously minimize the cardinality of B and the expected waiting time of the robot. Let the cardinality of B be denoted by $k = |B|$.

For a given value of k, we formulate the following decision problem: Given parameters k, T_w and an arrival distribution p, can we find a beacon schedule B such that $|B| = k$ and $ET(B,p) \leq T_w$? To minimize the number of beacons used to satisfy this T_w, we perform a search over the possible values of k by solving the decision problem for each value.

Typically, the time taken by the robot to traverse the whole round is much larger than the length of T. For such cases, we obtain the following insight about an optimal schedule.

Lemma 1. *During each round, there has to be a beacon at the last instant of time in T, to ensure that $ET(B,p) \leq T_w$.*

Proof. We prove this claim by contradiction. Suppose there is no beacon at the last time instant over which $p(t_i)$ is distributed. Let $t_j < t_n$ be the time at which the last beacon is scheduled. If the robot arrives after t_j, but before t_n, then it has to wait until the mote's next beacon, which, according to the distribution occurs only at the estimate of the next robot interarrival time. Since this can be of the order of the duration of a round, the waiting time can grow arbitrarily large, giving us a value greater than T_w: a contradiction.

Lemma 1 gives us a starting point to place a beacon: at time t_n (the end of T). Also, note that, in Equation 1 the robot's waiting time is determined by only the first beacon that it hears after arrival. This motivates the following dynamic programming solution.

Let the cost function be $C(i,t_j)$ which denotes the expected waiting time ("cost") for robot arrivals *after* interarrival time t_j, when beacon i is scheduled at time t_j.

$$C(i,t_j) = \min_{t_j < t_r < t_n} \left\{ \sum_{t_q=1+t_j}^{t_r} p(t_q) \cdot (t_r - t_q) + C(i+1,t_r) \right\} \tag{2}$$

Since beacon i is placed at time t_j and beacon $i+1$ at time t_r, the first term on the right-hand side in Equation 2 computes expected waiting time for a robot arrival between those two beacons. The second term computes the expected waiting time for arrivals after time t_r.

Since there must be a beacon at t_n (Lemma 1), $C(k,t_n) = 0$. Further, we do not allow beacon k to be scheduled anywhere except at time t_n, thus $C(k,t_y) = \infty$ for $t_1 \leq t_y < t_n$. We then use Equation 2 to compute the rest of the cost table, which in total is of size $k \times n$. To complete the computation, we need to increment each value $C(1,t)$ for all t by the expected waiting time for robot arrivals from t_0 to t. This accounts for robot arrivals *before* the first beacon.

The time at which the first beacon should be scheduled is the time step t_j such that the value of $C(1,t_j)$ is minimized, i.e. $b_1 = \arg\min_{t_j} C(1,t_j)$. Since the computation of $C(1,t_j)$ uses a minimum value for some $C(2,t_r)$, we backtrack to find the best possible scheduling times $b_2, b_3, \ldots, b_k$.

We want the value of k to be the least possible to satisfy the expected waiting time constraint. In order to do this minimization, we start with just $k = 1$ beacon i.e. a cost table of size $1 \times n$ and increase k if the expected waiting time for that k exceeds T_w. The entire computation can be performed in $O(kn^3)$ steps. Instead of this linear search, we could also use a binary search to find the best k, but that approach does not allow us to incrementally build the cost table. As a result, it ends up computing the whole table and then eliminating half of the values at each step.

3.2.1 Simulation Results

We demonstrate the utility of using the dynamic programming algorithm through simulated robot arrival times and beacon schedules.

We compare three different types of robot interarrival patterns: uniform, Gaussian and bimodal Gaussian (mixture of two Gaussian). In all three cases, the desired waiting time is 2.5 seconds and the robot interarrival times lie between 300 sec and 500 sec.

Figure 1 (left) models the robot's arrival pattern at a mote as a unimodal Gaussian. Uniform beaconing uses 34 beacons for an expected waiting time of 2.489 seconds. In contrast, our algorithm uses 10 beacons (≈ 70% better) with an expected waiting time of 2.233 seconds.

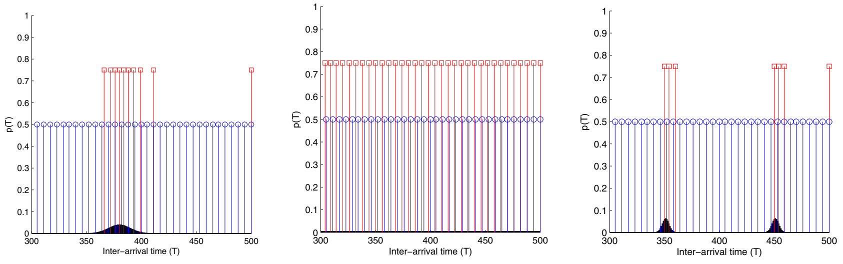


Fig. 1 A comparison of optimal beacon scheduling (red circles, tall) to uniform scheduling (blue circles, short) for different robot arrival patterns: unimodal Gaussian (left), uniform (middle), and bimodal Gaussian (right).

The algorithm is applicable to any type of arrival distribution. For instance, if the arrival pattern at a mote is a bimodal Gaussian (right of Fig. 1: $\mathcal{N}(350,10)$ and $\mathcal{N}(450,10)$ with equal weights), our solution uses 7 beacons ($\approx 79\%$ better) with an expected wait time of 2.115 seconds.

4 Local Search for Finding the Download Location

When the robot is downloading data from a node, the quality of the wireless communication link is a crucial factor in determining energy spent in communication: when the link quality is high, the same amount of data can be transferred using less energy. This in turn, drastically affects the lifetime of the node. In this section, we present a motion strategy for a robot to find a good location to download the data. The algorithm is based on insights from a series of experiments which we describe next.

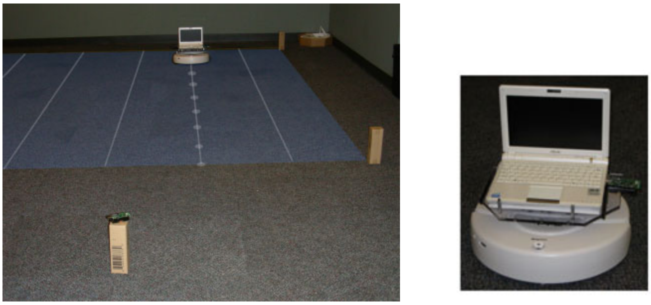


Fig. 2 Left: Experimental setup to measure the link quality of data transfer from a mobile robot to a base station, with the robot moving on a uniform grid.

We started our experiments by collecting data using the setup shown in Figure 2 where we placed an 11×11 grid on a $3m \times 3m$ indoor environment. We mounted a base station mote on our robot (iRobot Create with Asus Eee PC) and the robot autonomously visited grid points while pointing to a fixed direction.

Figure 3 shows signal strength measurements on the grid when the mote was placed at (3,10). Each measurement was taken by sending a 4 byte message during which Radio Signal Strength Indicator (RSSI) and Link Quality Indicator (LQI) values were recorded. From each location, robot samples 50 measurements. The left plots in Figure 3 show the mean (top) and median (bottom) values of the LQI measurements. The right plots show the RSSI values. All plots give a unified view of the link quality: although in general the LQI increases as we get closer to the mote, the surface is not smooth and contains many deep drops due to multi-path effects.

The next experiment illustrates the effects of the location and the link quality on the time to download the data. In Figure 4, the top figure shows the time to download 50 messages from each grid point. The peaks show a correlation with the deep fading effects in the previous experiment (Figure 3). Bottom left (resp. right) figure shows

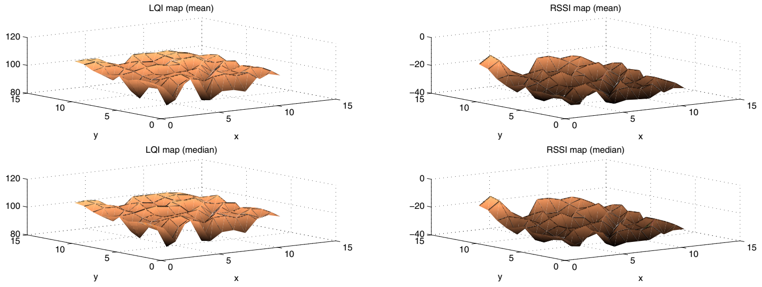


Fig. 3 The robot visited each location, and took 50 measurements. **Left:** mean (top) and median (bottom) values of the LQI measurements. **Right:** mean (top) and median (bottom) values of the RSSI measurements.

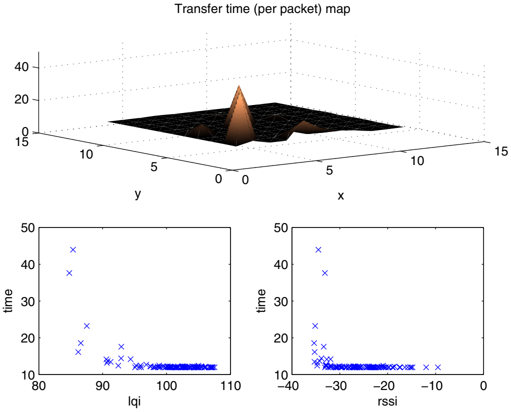


Fig. 4 Time to download 50 messages from each grid point as function of location (top), and as functions of LQI (bottom-left) and RSSI (bottom-right). With controlled mobility, the robot can decrease the download time significantly by moving slightly.

the relation between LQI (resp. RSSI) measurements and time transfer. As it can be seen in the left figure, the transfer time is very low for LQI measurements above 95. On the other hand, the transfer time increases drastically for values lower than 95. *This observation shows the potential utility of controlled mobility: robots can reduce the data download time (and increase the life of the sensor network) by finding a "good"*[2] *location to download the data.*

The next experiment sheds further light on path-loss and multi-path affects on link quality. To cover a wider range, we moved the robot on a line-segment in a corridor in our building and placed a mote on the mid-point of this line segment. In Figure 5-left, the mote is located at $x = 26$ and robot starts taking measurements at $x = 1$ and ends at $x = 51$. The discretization level is 1 foot, the robot takes 50 measurements from each location. Top figure shows mean values of 50 measurements,

[2] In this case, a good location is one where the LQI value is greater than 95.

bottom figure shows median of the measurements. As expected, the link quality increases when robots get closer to the mote, while it tends to decrease while getting further away from it. After performing similar experiments, we concluded that the following observation explains the link-quality behavior better: *Within a certain range (±8 ft, in this case), the link quality is consistently "good" and unpredictable (random) outside this range.*

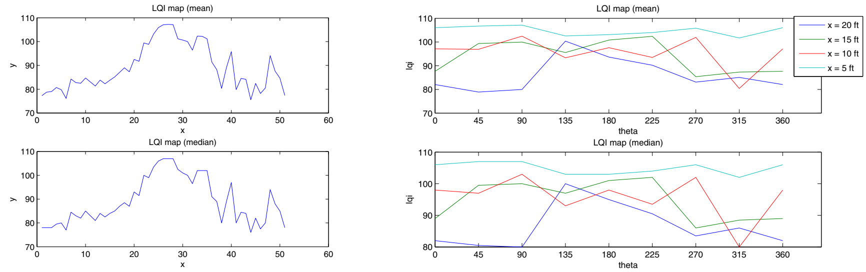


Fig. 5 **Left:** The mote is located at $x = 26$. The robot moves from $x = 1$ to $x = 51$. Within a range of ±8 ft, the link quality is consistently "good". It is unpredictable (random) outside this range. **Right:** The θ-values correspond to robot orientations. Each curve corresponds to a different location on a line. The mote is located at $x = 0$. The behavior of RSSI or LQI as a function of rotation is not easily predictable.

Most robotic systems have a rotational component which means that we can control the orientation of the robot. Therefore, the robot should search for not only a good location but a good orientation as well. The next experiment focuses on this aspect. Figure 5-right shows the change in link quality with the various orientations of the base station (on the robot) at fixed locations. The figure shows the results for 4 fixed points at distances 5,10,15 and 20 feet from the mote. The results show that when the base station is close to the mote, the orientation does not affect the link quality significantly. However, when the distance is large, small changes in orientation may result in drastic changes in link quality. Moreover, this change is not easily predictable. For example, when the robot is at the furthest point (the 20-feet curve), mote and base station point towards each other when the angle is 180°. In this orientation, the LQI is 95. If robot turns 45° in counter-clockwise direction, the LQI value increases to 100. If the robot turns 45° more on counter-clockwise direction, then the LQI value suddenly drops to 80. This example also shows that measurements from various orientations may not give a clear indication about the direction of the mote.

The results of these experiments can be summarized as follows:

- Within a certain distance (an environment dependent parameter), the signal quality is predictably good and the orientation of the robot does not make a significant difference.
- When the robot is outside this range, it is very difficult to use local information (such as gradient) to find the location or orientation of the mote.

In the next section, we present a search strategy based on these observations.

4.1 The Search Algorithm

In this section, we describe a search algorithm to find a good download location. In many applications, it is beneficial to search for this location in an online fashion because the location of the mote whose data will be downloaded can change locally, the signal properties may change over-time, or the robot may not have the localization capability to visit a location accurately.

The experiment in the previous section indicates that it is very difficult, if not impossible, to use local gradient information to seek a good location. A global approach is needed. The strategy we present uses two environment dependent parameters. The first parameter β is a lower-bound for an acceptable signal strength (LQI value). For example, an appropriate β value for the environment where the experiment shown in Figure 4 was performed, is 95. The second value α is mainly a grid resolution and it is set to the distance within which the link quality is predictably good. For the environment where the experiment shown in Figure 5-left was performed, an appropriate α value is 8 ft.

Upon hearing a beacon message, the robot searches for a good location by placing a grid on the environment where the dimension of each cell is determined by α. When the robot visits a grid cell, it rotates $0, 90, 180, 270$ degrees. This allows us to get rid of local multi-path effects and to simultaneously seek a good orientation. At each rotation, the robot takes five link quality measurements. The quality of each orientation is defined as the median of these five measurements. The weight of each cell is then set to the the highest of these four median values. In the algorithm below, $measure(c)$ subroutine performs these steps at cell location c. We also keep track of a table where we store the expected link quality values. For each unvisited cell, we set the *expected_weight* value to the average link quality value of its immediate neighbors. Next, robot visits the location with maximum expected weight. The algorithm is given below:

Algorithm 1 *LocalSearch*

```
expected_weight$(c \in C) \leftarrow 0$, $c \leftarrow (0,0)$ (initial location)
while there are unexplored cells do
    if $measure(c) \geq \beta$ then
        return
    end if
    Forall $c' \in Neighbor(c)$ if $c'$ is unvisited,
    $expected\_weight(c') = mean(\forall_{c'' \in Neighbor(c')} measure(c''))$
    $c \leftarrow \max_{c'} expected\_weight(c')$
end while
```

Remark 1. If the β value is not known, we can set it to a high value. In this case, the robot will visit all grid-cells. We can then pick the best location.

Remark 2. We can incorporate collision avoidance into the strategy by setting the weight of a cell to zero if there is an obstacle at that cell.

In the next section, we demonstrate the utility of this strategy with a series of experiments.

4.2 Search Experiments

We tested the search algorithm in a number of settings. In this section, we present two of these results.

The first experiment was performed in the indoor setting shown in Figure 6-left. The signal strength level in TelosB mote was set to 3. The other system parameters were $\alpha = 1.5m$ and $\beta = 100$.

When the robot started from the initial location shown in the figure, it quickly converged to a good location. Robot's steps are shown in Figure 6-left bottom where

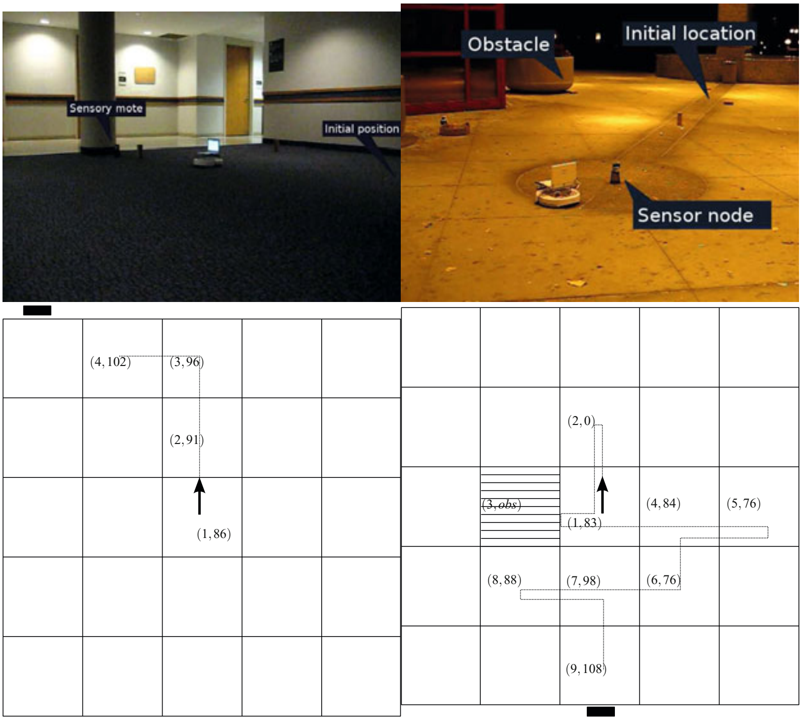


Fig. 6 Bottom figures show a virtual grid used by the search algorithm. The visited cells are labeled with the format (s,r): s is the order the cell was visited and r is the maximum median value sampled from four orientations. The black rectangles show the location of the mote. **Top Left:** The setup for an indoor experiment. The picture shows the best configuration found by the search algorithm. **Bottom Left:** Steps in finding a good location in the setup shown on top. **Top Right:** Search performed in an outdoor setting. The picture shows the best configuration found by the search algorithm. The shaded cell corresponds to the obstacle that robot avoided. **Bottom Right:** Steps taken during outdoor search.

the visited cells are labeled with the format (s,r): s is the order the cell was visited and r is the maximum median value sampled from four orientations.

The second experiment was performed outdoors with $\alpha = 3m$ and $\beta = 100$. As shown in Figure 6-right, the robot quickly converged to a good location.

In conclusion, the simple search strategy presented in this section was efficient in finding a good download location. Where most local search heuristics would get stuck with a single cell, the presented search strategy quickly converges to a robot pose from where the data can be downloaded efficiently.

5 System Design

In this section, we describe a system which incorporates the results presented in this paper. In Section 5.3 we present experimental results which demonstrate the utility of these components.

5.1 Hardware Components

Our system consists of three classes of devices. (i) The sensor motes are Crossbow Telos, (rev. B) 802.15.4 compliant. We deployed three static motes in the fourth floor lounge of the Digital Technology Center (DTC) at the University of Minnesota, Twin-Cities. The experimental setup is shown in Figure 7. (ii) The mobile robot is an iRobot Create without the command module. (iii) The control program for the robot runs on an Asus Eee PC, which interfaces with the Create directly through a USB-to-Serial cable. The system ran Linux (Ubuntu) and our Java and C++ programs used serial communication libraries to write motion commands to the robot, in accordance with the Create Open Interface (OI) specifications.

5.2 Adaptive Beacon Scheduling

The control program on the TelosB motes was written in the `nesC` language, then compiled and programmed onto the mote using `TinyOS 2.x`. In our design, sensing motes transmit beacon messages and the base station mote attached to the mobile robot listens for these messages.

To allow the TelosB motes to have an adaptive beacon schedule, we store a beacon time interval array on each mote. A one-shot timer cycles through the array, allowing the mote to keep its transmitter off for arbitrary intervals. We set the receiver sleep interval using the `LowPowerListening` interface. However, we believe that since we have packet acknowledgments enabled, the receiver of the sensing mote is turned on every time it sends a beacon. This design decision could be replaced with unacknowledged packets. In both cases, our optimal beacon schedule helps save power on the mote by reducing the amount of time during which the transmitter and/or the receiver are active.

5.3 Experiments

We performed four experiments to demonstrate the utility of incorporating both beacon scheduling, and local search. The experiments are: baseline (B), beacon scheduling (BS), local search (LS) and both local search and beacon scheduling (LSBS). Each experiment consisted of 8 rounds. In each round, robot visits a predefined location for each mote, and downloads the data from that mote. The locations that the robot starts downloading (shown as squares in Fig. 7) are fixed for comparison purposes: For example, if in experiment B the robot downloads from a fixed location then in experiment LS, the robot starts the local search from the same location.

We picked a range of download locations in a mote's vicinity to simulate the effects of localization uncertainty: If the robot does not have accurate means of localization, even if it targets a fixed location to download data, it may be off from that location by a distance given by the uncertainty range. After arriving at a predetermined location, the robot either directly downloads the data (experiment B and BS), or performs a local search to find a good location before downloading (LS and LSBS experiments). After download finishes, the robot either continues to the next mote directly (experiments B and LS), or computes an updated beacon schedule based on interarrival times, uploads it to that mote and proceeds to the next mote (experiments BS and LSBS).

In all experiments, the beacons are special messages whose payload consists of (i) the node id of the mote, and (ii) a sequence number of the triggered beacon. To compare the local search with base case, we needed a mechanism to compare the trade-off between energy gain in efficient download and energy spent in extra beacons sent during the search phase. Therefore, we used data packages which are the same as the beacon type messages. To download the data on the mote, the robot must successfully hear 100 additional beacons. This represents scenarios where the

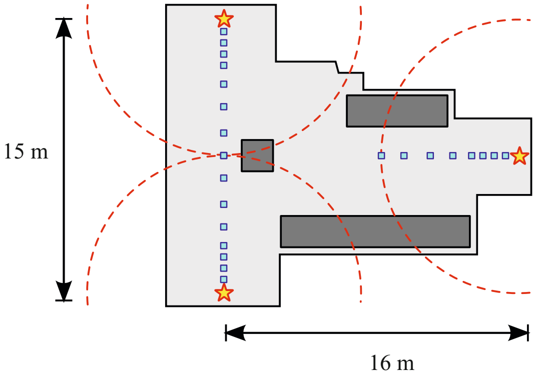


Fig. 7 A proof-of-concept deployment. The stars are approximate locations of the data nodes. The dashed lines show their communication range. The squares are locations where the robot starts either the download or the local search.

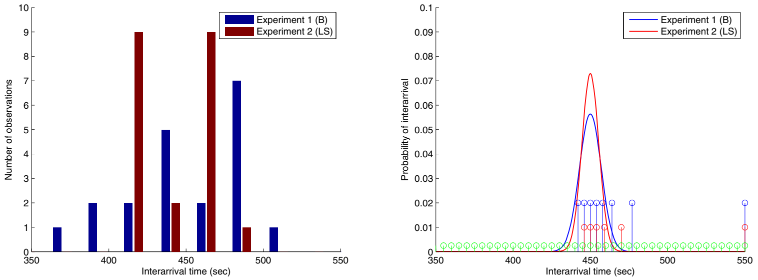


Fig. 8 The robot interarrival times from our experiments were modeled as normal distributions.

data stored on the mote corresponds to 100 messages and all of it must be successfully downloaded. This way, we can use the last received beacon sequence number for each mote to represent the total energy consumption metric spent in beaconing and download. Even with this modest amount of data, each experiment lasted about an hour.

In experiment B, we chose the beacon interval for discovery phase as 5 seconds. This guaranteed an expected robot waiting time of 2.5 sec. The optimal beacon scheduling algorithm of Section 3 and the robot inter-arrival distribution observed in experiment B are used to achieve 2.5 sec waiting time in experiment BS. In experiment LSBS, we used the interarrival times from the LS experiment to compute the optimal beacon schedule for this case. The recorded interarrival times and the robot's arrival model are shown in Figure 8. In experiment BS, the optimum beacon schedule uses 8 beacons.

In comparison to the number of beacons ($550/5 = 110$) in the base experiment B, the beaconing strategy yields significant energy savings (8 beacons) while satisfying the same expected waiting time constraint. Comparing the total number of beacons, we can see the effect in total performance. Beacon scheduling in experiment BS reduced the total number of beacons to 2791 compared to 4839 in experiment B, the baseline (first and second columns in right of Table 1).

The left one of the two tables shown in Table 1, shows the packet loss rates for various locations in each experiment. Clearly, local search provides a significant reduction in packet loss rate for the first two locations (compare B versus LS and BS versus LSBS) where the distance prevents a lossless communication between the mote and robot. For example, in the experiment B, the first mote has to sent 538 packets until the robot successfully downloads all of the 100 data packets, whereas after local search no packet is lost. We can see the efficiency of local search for the first two rounds of the examples in table on the right (Table 1). On the other hand, for the rest of the rounds, the local search does not provide significant gains. In fact, the energy consumption slightly increases in experiment LSBS compared to BS experiment due to the overhead (i.e. number of beacons sent during local search).

It is worth noting that in this indoor setting, the robot's total path is relatively short compared to the search distance. Thus, the search overhead becomes comparable to the number of discovery beacons. When the travel distances are large, (e.g. in outdoor settings), the search overhead will become negligible. In this case, local search will yield more significant energy savings.

Overall, the experiments clearly demonstrate that (i) adaptive beaconing strategies yield significant savings in the number of discovery beacons sent, and (ii) local search strategies can result in drastic improvements in the download time when the link quality is unpredictable.

Table 1 The **Left** table shows the package loss rates for each experiments (B:Base,LS:Local Search,BS: Beacon Schedule, LSBS: Local Search and Beacon Schedule together) with respect to the distance that robot starts to download or starts to the local search. For each download we calculate the number of packet loss until robot hears 100 beacons. **Right** figure shows the total number of beacons send from 3 motes during the experiment.

Dist.	B	BS	LS	LSBS
7.5m	173%	36%	0%	1%
6.25m	7%	21%	0%	1%
5m	11%	0%	0%	0%
3.75m	8%	0%	0%	0%
2.5m	2%	3%	0%	3%
1.7m	0%	0%	0%	0%
0.9m	0%	0%	0%	0%
0.1m	0%	0%	0%	0%

	B	BS	LS	LSBS
1-2	1642	897	997	828
3-8	3197	1894	3215	2152
Total	4839	2791	4212	2980

6 Conclusion

In this paper, we addressed two problems that arise in applications where robots collect data from static nodes. In the first problem, the goal is to minimize the energy spent by the static nodes for beaconing. For this problem, an optimal beaconing strategy based on dynamic programming was presented. In the second problem, the goal is to minimize the energy spent in communication. For this purpose, we present a strategy for the robot to adaptively discover a download location where the signal is strong. The strategy is based on insights gathered by the experiments. We report these in the paper as well. Finally, we present an indoor system for data collection which incorporates the algorithms presented in the paper. Experiments performed on the system demonstrate the utility of the two results in the paper.

There are additional factors (e.g. robot's interarrival times, the amount of data to be downloaded at each round) which effect the overall system performance. Currently, we are building an outdoor system for habitat monitoring including a new robotic platform. In the near future, we will demonstrate the use of these results within the context of a field application in environmental monitoring.

Acknowledgments

This work is supported in part by NSF Grants 0907658, 0917676 and 0936710.

References

1. Atay, N., Bayazit, B.: Mobile Wireless Sensor Network Connectivity Repair with k-Redundancy. In: Proceedings of the 2008 International Workshop on the Algorithmic Foundations of Robotics (WAFR) (2008)
2. Cerpa, A., Estrin, D.: ASCENT: Adaptive Self-Configuring sEnsor Networks Topologies. IEEE Transactions on Mobile Computing 3(3), 272–285 (2004)
3. Chatzigiannakis, I., Kinalis, A., Nikoletseas, S.: Efficient data propagation strategies in wireless sensor networks using a single mobile sink. Computer Communications 31(5), 896–914 (2008)
4. Corke, P., Hrabar, S., Peterson, R., Rus, D., Saripalli, S., Sukhatme, G.: Autonomous deployment and repair of a sensor network using an unmanned aerial vehicle. In: Proceedings of the 2004 IEEE International Conference on Robotics and Automation (ICRA), vol. 4, pp. 3602–3608 (2004)
5. Dixon, C., Frew, E.W.: Controlling the mobility of network nodes using decentralized extremum seeking. In: 45th IEEE Conference on Decision and Control (2006)
6. Kansal, A., Somasundara, A.A., Jea, D.D., Srivastava, M.B., Estrin, D.: Intelligent fluid infrastructure for embedded networks. In: MobiSys 2004, pp. 111–124. ACM Press, New York (2004)
7. Li, Q., Rus, D.: Navigation protocols in sensor networks. ACM Transactions on Sensor Networks 1(1), 3–35 (2005)
8. Lindhe, M., Johansson, K.H.: Communication-aware trajectory tracking. In: Robotics and Automation, ICRA 2008, pp. 1519–1524 (2008)
9. Ma, J., Chen, C., Salomaa, J.P.: mWSN for Large Scale Mobile Sensing. Journal on Signal Processing Systems 51(2), 195–206 (2008)
10. Mostofi, Y.: Communication-aware motion planning in fading environments. In: Robotics and Automation, ICRA 2008, pp. 3169–3174 (2008)
11. Musăloiu-E. R., Terzis, A., Szlavecz, K., Szalay, A., Cogan, J., Gray, J.: Life Under your Feet: A Wireless Sensor Network for Soil Ecology. In: EmNets Workshop (2006)
12. Schurgers, C., Tsiatsis, V., Ganeriwal, S., Srivastava, M.: Topology management for sensor networks: exploiting latency and density. In: MobiHoc, pp. 135–145 (2002)
13. Shah, R.C., Roy, S., Jain, S., Brunette, W.: Data mules: modeling and analysis of a three-tier architecture for sparse sensor networks. Ad Hoc Networks 1(2–3), 215–233 (2003)
14. Somasundara, A.A., Kansal, A., Jea, D.D., Estrin, D., Srivastava, M.B.: Controllably mobile infrastructure for low energy embedded networks. IEEE Transactions on Mobile Computing 5(8), 958–973 (2006)
15. Tariq, M.M.B., Ammar, M., Zegura, E.: Message ferry route design for sparse ad hoc networks with mobile nodes. In: MobiHoc 2006, pp. 37–48 (2006)
16. Tekdas, O., Isler, V.: Robotic routers. In: Proceedings of the 2008 IEEE International Robotics and Automation, ICRA 2008, pp. 1513–1518 (2008)
17. Tekdas, O., Lim, J.H., Terzis, A., Isler, V.: Using mobile robots to harvest data from sensor fields. IEEE Wireless Communications (2008)
18. Wang, W., Srinivasan, V., Chua, K.C.: Using mobile relays to prolong the lifetime of wireless sensor networks. In: MobiCom 2005, pp. 270–283. ACM Press, New York (2005)

Micro-Robots

Cédric Pradalier

This final chapter of the ISRR'09 proceedings report on a very particular field of robotic research, namely micro-robotic. While more classical robotic focuses on perception and control, micro-robotic is more concerned with the question of manufacturing and actuation at the millimetre or sub-millimetre scale.

- In “ Design and Fabrication of the Harvard Ambulatory Micro-Robot ”, Baisch and Wood report on the challenges related to the manufacturing of a 90 mg hexapod robot, using smart materials and innovative fabrication techniques. Actuation is implemented using shape memory alloy controlled from external electrical sources.
- In “Assembly and Disassembly of Magnetic Mobile Micro-Robots towards 2-D Reconfigurable Micro-Systems ”, Pawashe et al. address the problem of designing a modular reconfigurable system with modules smaller than 1mm. They consider, in particular, the question of assembly and disassembly of the modules, using only electrostatic and magnetic forces.

These chapters show that micro-robotic technology has reached a stage where controlled locomotion of a miniature system is possible. There is still, however, a considerable amount of research and engineering required to be able to one day use these systems for medical inspection as envisioned in the “Innerspace” film, directed by Joe Dante.

Design and Fabrication of the Harvard Ambulatory Micro-Robot

Andrew T. Baisch and Robert J. Wood

Abstract. Here we present the design and fabrication of a 90mg hexapedal microrobot with overall footprint dimensions of 17 mm long x 23 mm wide. Utilizing smart composite microstructure fabrication techniques, we combine composite materials and polymers to form articulated structures that assemble into an eight degree of freedom robot. Using thin foil shape memory alloy actuators and inspiration from biology we demonstrate the feasibility of manufacturing a robot with insect-like morphology and gait patterns. This work is a foundational step towards the creation of an insect-scale hexapod robot which will be robust both structurally and with respect to locomotion across a wide variety of terrains.

1 Introduction

Autonomous mobile robots are a desirable alternative to sending humans into hazardous environments such as natural disasters. Such robots equipped with embedded sensors could provide quick reconnaissance regarding survivor locations and chemical toxicity levels, or simply map out an area for rescue workers. In these situations, large numbers of insect-scale devices using swarm algorithms might be more efficient than larger, less agile, and more expensive robots.

There are multiple potential locomotion modes for microrobots, including crawling [10], rolling, flying [24], jumping [3], gliding [26], slithering, or rockets [22]. This work focuses on ambulation, which has been shown to be robust and efficient on diverse terrain by studies of insect locomotion. Some species of cockroach are capable of running at speeds up to 20 body lengths per second (*Blaberus discoidalis*)

Andrew T. Baisch
Harvard University School of Engineering and Applied Sciences, Cambridge, MA
e-mail: abaisch@seas.harvard.edu

Robert J. Wood
Harvard University School of Engineering and Applied Sciences, Cambridge, MA
e-mail: rjwood@eecs.harvard.edu

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 715–730.
springerlink.com

and even 40 body lengths per second (*Periplaneta Americana*) [8]. Cockroaches are known to maintain high speed ambulation across rough, uneven terrain [21] and provide further design inspiration with their ability to climb sheer vertical and even inverted surfaces [17]. Furthermore, they have efficient navigational aids such as antennae that allow obstacle avoidance even in low lighting [13]. These aspects of insect morphology enable robust locomotion, and we strive to translate the underlying principles into designs for an insect-scale ambulatory robot.

Roboticists have already proven the importance of using nature as a design guide, as evidenced by legged robots that are more agile than their wheeled counterparts [19],[15],[16]. There has been work to create climbing robots for rough [11] or smooth [12] surfaces, and even efforts to integrate insect-like proximity sensors for navigation along walls [14], [4]. Others have attempted to use MEMS fabrication to create silicon insect-scale ambulatory robots [6], [27], [5].

Our goal is to combine many of these functions into a single autonomous millimeter-scale robot. This work focuses on the design and fabrication of thoracic and leg mechanics, actuation, and electronics required to achieve such a goal. Furthermore we examine potential modes of attachment as a preliminary study on scaling vertical surfaces with a microrobot. We conclude with an overview of remaining research topics towards the creation of a fully autonomous ambulatory robotic insect.

2 Robot Design

The design of the Harvard Ambulatory Microrobot (HAMR), presented in Fig. 1, was motivated by several factors, including available fabrication techniques, and the necessary body mechanics to achieve gaits inspired by insects. Important constraints in designing a microrobot are brought about by the fabrication techniques and materials that may be used. A solution will be discussed in Section 3, however our designs will be based on standardized components such as rigid links, flexure joints, and linear actuators.

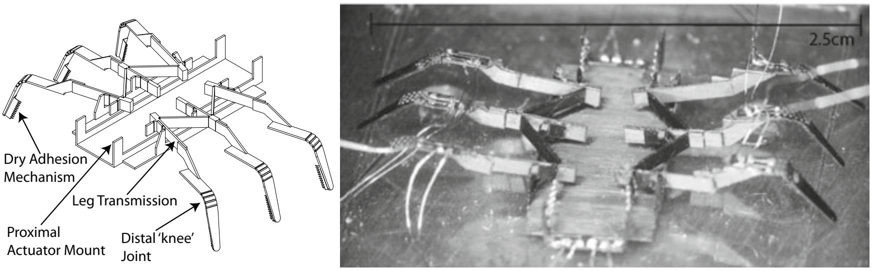
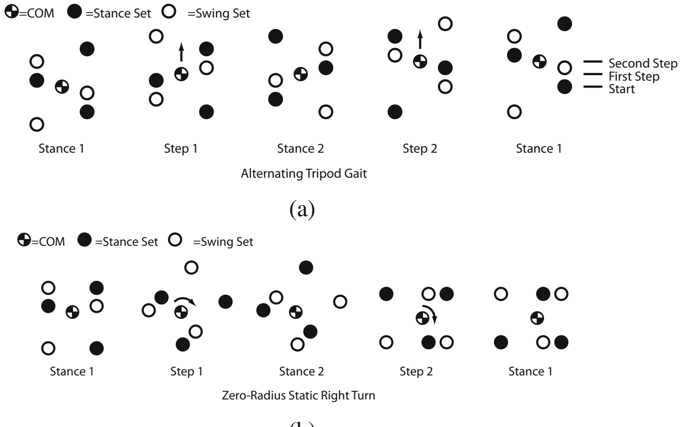


Fig. 1 Notional design of the thorax and legs of the Harvard Ambulatory Microrobot (left) and initial prototype (right)

Insect, and in particular cockroach, morphology demonstrates the utility of a hexapedal structure for static stability during locomotion. As opposed to bipedal or quadrupedal animals, hexapods typically exhibit static stability through the duration of their gait (for non-running gaits). We chose six legs as opposed to a greater number primarily for fabrication simplicity. Additionally, we believe a sprawled posture will be beneficial for future endeavors such as scaling vertical surfaces [9]. Our design adheres to insect-scale physical parameters: 100mg maximum body mass (excluding electronics), and overall footprint of approximately $2cm^2$.

To achieve maximum functionality in diverse environments HAMR must be capable of attaining and transitioning between numerous basis gaits. In forward locomotion, hexapods typically exhibit an alternating tripod gait (Fig. 2a) in which the anterior and posterior leg on one side of the body and the opposite center leg are in ground contact during each half period of a gait. The stance legs move towards the rear of the body, propelling the body center of mass (COM) forward. During the airborne, or swing phase, legs come into ground contact when stance legs reach their maximum displacement. Alternating tripod is not the sole required motion, however, since our robot must have maneuverability in diverse environments. We must achieve zero-radius turns (Fig. 2b) and/or gradual turns, which require phase adjustment between legs. As in animals, transitions between gaits must occur continually in response to locomotion speed and terrain. This dynamic gait modulation (DGM) must be achievable without unnecessarily increasing the robot complexity, power and control requirements, or exceeding physical constraints.



(a)

(b)

Fig. 2 Hexapods can achieve a variety of gaits. a) An alternating tripod gait propels the body COM forward during each step, as indicated by horizontal lines. b) Proposed gait pattern for zero-radius static turns that operate when one or more of HAMR's stance legs slips during rotation.

2.1 Thoracic Mechanics

To achieve the design requirements, HAMR's mechanics are separated into two distinct modes of actuation: proximal, which is controlled in the thorax, and distal, or lower-leg control. This feature allows simple control of transverse (parallel to ground) leg movement through two thoracic DOFs and gait definition by addition or removal of each of the six legs to the stance set.

HAMR's proximal mechanics consist of two leg transmissions, each controlling three legs. The transmissions are mirrored across the sagittal plane, and therefore legs are grouped on each side of the robot's thorax. An alternative grouping would couple the anterior and posterior leg with the center leg on the opposite side of the body. However, this method would require crossing linkages and dividing the single actuation plane into two. With the two leg transmissions coplanar, fabrication complexity is minimized and it ensures that the body center of mass lies in the sagittal plane. Any mass asymmetry could lead to difficulties in attaining a stable gait.

2.1.1 Kinematics

The kinematic coupling of a single transmission requires that the angles, θ_i of each outer leg $(i = 1,3)$ move in the same direction, while the center leg $(i = 2)$ moves with opposite sign. To determine the orientation, θ_i of each leg coupled by a single transmission, we can analyze the forward kinematic mapping of the two four-bar mechanisms that comprise each transmission given a single input angle the most anterior leg θ_{L1}. Analysis of the linkages labeled in Fig. 3 was done assuming each link is rigid and each joint is a pin joint. Using four-bar mechanism kinematic analysis software from [20], the lower four-bar mapping (θ_{L2} output) is

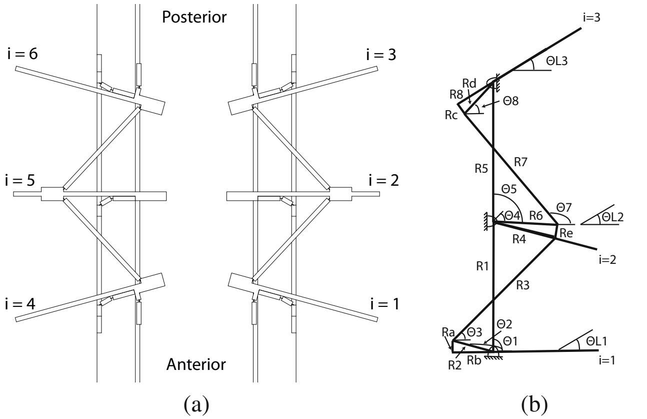


Fig. 3 a) HAMR's thoracic mechanics design with leg numbering convention that is used in this paper. b) Kinematics of HAMR's thoracic mechanics

$$\theta_{L2} = \theta_4 = atan2(y_1,x_1) - cos^{-1}((x_1^2 + y_1^2 + R_4^2 - R_3^2)/(2R_4\sqrt{x_1^2 + y_1^2})) \tag{1}$$

where $x_1 = R_1 \times \cos(\theta_1) - R_2 \times \cos(\theta_2)$, $y_1 = R_1 \times \sin(\theta_1) - R_2 \times \sin(\theta_2)$, R_{1-4} are known link lengths, and θ_1 is the known ground link orientation. We can again use eqn.(1) to calculate θ_{L3} using θ_{L2} as input. This yields

$$\theta_{L3} = atan2(y_2,x_2) - cos^{-1}((x_2^2 + y_2^2 + R_8^2 - R_7^2)/(2R_8\sqrt{x^2 + y^2})) \\ - \tan^{-1}(R_c/R_d) \tag{2}$$

Where $x_2 = R_5 \times \cos(\theta_5) - R_2 \times \cos(\theta_6)$, $y_2 = R_5 \times \sin(\theta_5) - R_6 \times \sin(\theta_6)$, $R_{5,7,8}$ and θ_5 are known, $R_6 = \sqrt{R_4^2 - R_e^2}$, $\theta_6 = \theta_4 + asin(R_e/R_4)$. Eqns.(1), (2) map input angle to output angles for one half of a leg-stroke. The thoracic mechanics are driven by two sets of proximal actuators (driving θ_{L1}). The stance is determined by distal knee actuators described in sec. 2.2.1.

2.1.2 Proximal Actuation

Shape memory alloy (SMA) and piezoelectric actuators were considered for proximal actuation. Piezoelectric actuators provide benefits such as greater efficiency and bandwidth, and will be considered for future iterations to achieve dynamic gaits. However, for this version of HAMR we chose SMA for ease of powering, control, and integrability. SMA materials can be actuated simply by Joule heating and therefore require only a switched current source as detailed in sec. 2.3. SMA actuators, as opposed to piezoelectric cantilever or stack actuators, require minimal additional transmission to convert small oscillatory motions into the desired leg swing.

Discrete SMA actuators are unidirectional linear actuators, and therefore a single DOF requires an antagonist pair (Fig. 4). Each of HAMR's four thorax actuators control a unipolar motion of one leg transmission. A two-dimensional spring design (Fig. 5) was chosen since it provides linear actuation when pre-strained in tension, and receives a larger displacement in a smaller actuator length than a wire. Since we are using a linear actuator to move along a radial path, HAMR's proximal actuators are oriented parallel to the sagittal plane to prevent singularities. A flexible attachment is provided to prevent the actuator from bending across its thickness, and is therefore constrained to motion in a single plane. To characterize our spring design, we tested output displacement for a single SMA spring acting against an antagonist. The resulting displacement was approximately 300 μm.

2.2 *Leg Mechanics*

HAMR's legs, each of which has an articulated knee joint, allows addition or subtraction of a leg from the stance set. Including thorax and legs, there are a total 8 DOFs, each driven with binary actuation and therefore we may define 2^8 or 256

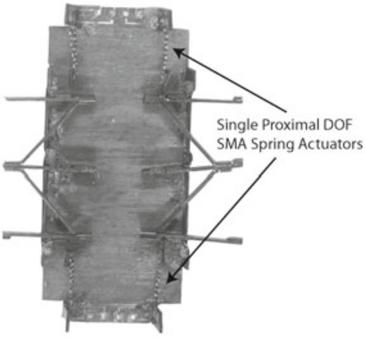


Fig. 4 HAMR's thorax and leg transmission. Each transmission DOF requires two SMA spring actuators. A flexure joint between actuator and transmission prevents singularities and restricts actuator movement to a single plane.

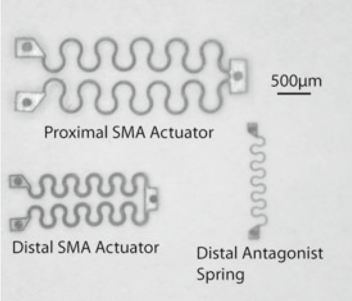


Fig. 5 HAMR's Proximal and distal actuators.

stances. Many of these stance patterns are useless to locomotion, but by transitioning between various stances we may prescribe numerous gaits including alternating tripod, zero-radius turns, and pre-loading attachment mechanisms for wall climbing.

2.2.1 Distal Actuation

HAMR's distal motion is provided by SMA actuators, however they function as cantilevers rather than the planar spring in Section 2.1.2. To conserve power during immobile stance phases, HAMR's knee joints are held down with a passive antagonist 2D spring (Fig. 5) and actuated to a raised position. Because of the simplicity and short time constant of thin foil spring fabrication (See Section 3.2), we used an iterative design process to find an appropriate force/displacement balance for leg actuation. This balance included a large displacement with a short full-cycle actuation time. Fig. 6 shows the maximum attainable actuated displacement of our design of 9.1 degrees.

To further inform our design we determined optimal actuation parameters. First, we experimentally determined the current limits for each actuator, the lower limit being necessary for Martensite-Austenite transition and upper limit causing the

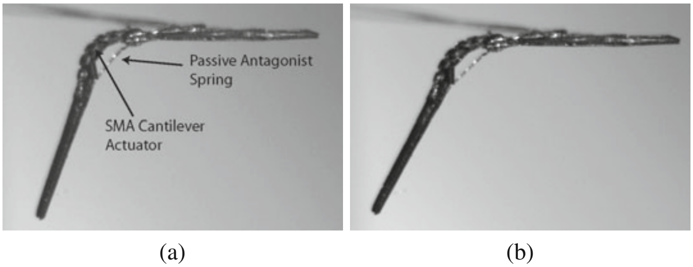


Fig. 6 HAMR's leg a) unactuated and b) actuated to maximum displacement. The maximum measured angle was 9.1 degrees.

bonding solder to melt. These limits are 100-150mA and 200-300mA for proximal and distal actuators, respectively. Using ProAnalyst 2D motion tracking software, we characterized the optimal current amplitude and pulse width for distal actuation. Here, optimal is defined as minimum overall actuation period (upstroke and downstroke) while maintaining at least 90 percent of maximum displacement. However in future iterations with on-board power we will also consider energy efficiency as a metric. For these tests the series resistance (See Fig. 7) was tuned to 5Ω and voltage was varied to obtain a current. The optimal leg actuation parameters are 280mA for 0.24s. This corresponds to 274mW power dissipation across the 3.5Ω SMA, and a total of 66mJ of energy per leg, per cycle.

2.3 Power and Control Electronics

HAMR's power electronics must be capable of driving four proximal SMA actuators each with 5.5Ω resistance and six distal actuators each 3.5Ω . We have experimentally determined that each thorax actuator requires 100-150mA and each leg actuator requires 200-300mA for actuation. We simply use ten *n*-type metal oxide semiconductor (NMOS) switches for binary control of the actuators with current tuned by 10Ω potentiometers in series with each actuator. The input signals are generated using Simulink and an xPc target system, which mimics the functionality of an integrated microcontroller, a feature of future iterations. Fig. 7 shows a schematic of the drive circuitry.

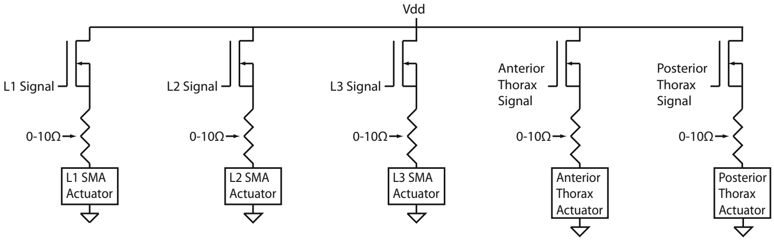


Fig. 7 Circuit drive for one of two contralateral actuator sets.

3 Fabrication Processes

When considering available fabrication techniques, millimeter-scale structures lie in an unconventional size regime. The overall system is too large to use microelectromechanical system (MEMS) fabrication exclusively, however too small for macro-scale machining. Additionally, when creating articulated structures at the millimeter-scale we are unable to consider classic revolute and telescopic joints or motors since they become inefficient as surface area effects become more relevant than Newtonian forces [23]. Therefore HAMR's physical structure was created utilizing the smart composite microstructure (SCM) manufacturing paradigm [25].

3.1 Smart Composite Microstructures

SCM fabrication utilizes laser-micromachining to pattern two structural composites and a flexible polymer layer, which assemble to form complex 3D articulated microstructures. Using the fabrication process detailed in Fig. 8, we created three leg transmission components (Fig. 9) from M60J carbon fiber laminates and 7.5μm polymide film that assemble into a complete eight DOF mechanism. Articulation is afforded through flexure joints, which exist where cuts in the composite face sheets allow the polymer to bend. This technique of creating flexures also enables the creation of sacrificial flexure joints, thereby creating 3D structures.

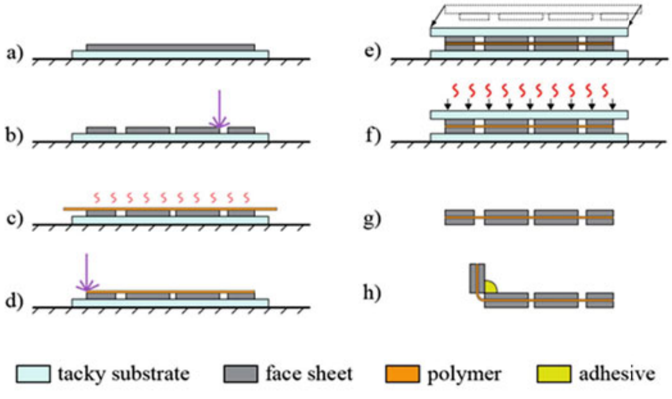
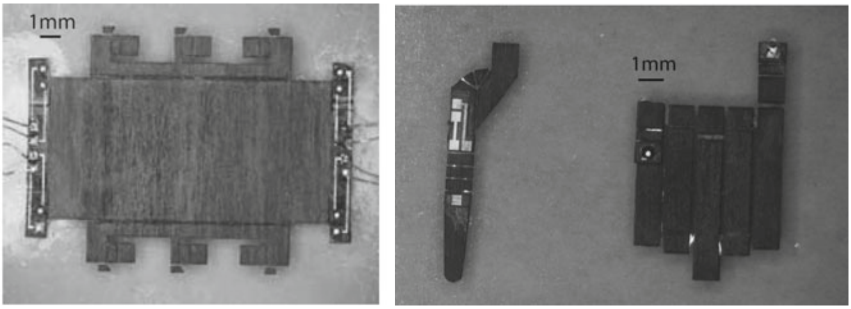


Fig. 8 SCM Fabrication process flow for creating flexure-based articulated structures. Carbon fiber prepreg (a) is laser micromachined to create a face sheet (b) and a polymer layer is debulked onto the face sheet (c). The polymer is patterned using different laser settings (d) and aligned to a second face sheet using kinematic mounts or optical alignment techniques (e). This laminate is vacuum bagged and cured (f). The quasi-planar stucture is released (g) and subsequently folded into 3D shapes (h).



(a) (b)

Fig. 9 HAMR's leg and transmission parts before folding to a 3D structure: a) Thorax b) Leg and leg transmission.

3.2 Actuators, Attachment Mechanisms, and Electronics

3.2.1 Actuators

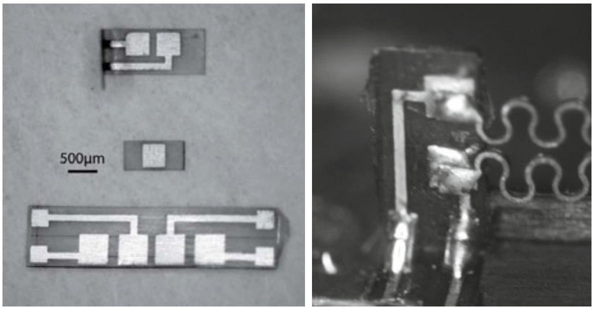
HAMR's two dimensional shape memory alloy (SMA) actuators were laser-cut from 50μm Nitinol sheets. As detailed in Sec. 2.2.1, passive springs were used as an antagonist for leg actuation. These springs were laser-cut from 25μm stainless steel. Similarly, ribbed attachment mechanisms for rough vertical terrain were cut from 50μm stainless steel. Fabrication of all three thin-foil metal structures required approximately one minute per batch.

3.2.2 Flexible Circuits

Flexible circuits are used to create an electrical path between HAMR's power input lines and each actuator. Using conventional photolithographic methods, circuits are fabricated from a 12.5 μm copper layer deposited on a 12.5 μm flexible polyimide layer. The resulting compliant circuits (Fig. 10a) are layered on the microstructures and cured along with the entire laminate. The entire structure may be fold-assembled without fracturing the copper traces. Using this process we created 500μm square solder terminals connected by 125μm thick wire traces. Future iterations will use flex circuits to house on-board power and control electronics. We also implemented flex circuits to increase joint rigidity by soldering across two terminals placed on orthogonal surfaces (Fig. 10b).

4 Results

HAMR is capable of displaying various leg gaits such as an alternating tripod and static zero-radius turns. To demonstrate gaits we actuated using our empirically-derived optimal parameters, defined here as minimizing simulated actuation period



(a) (b)

Fig. 10 a) Flex circuits trace power to HAMR's actuators. The copper traces may be folded without fracturing (top circuit). b) Solder is used structurally to increase joint strength once folded into the desired configuration.

while obtaining maximum leg deflection. Using ProAnalyst 2D motion tracking software to plot leg deflection for various energy inputs, we determined the maximum actuated leg angle to be 9.1 degrees. The parameters to obtain maximum angle at minimum speed are 280mA with a 0.24s pulse width. In the future we must consider energy usage in our description of optimality to conserve the limited lifetime of on-board power.

4.1 Alternating Tripod Gait

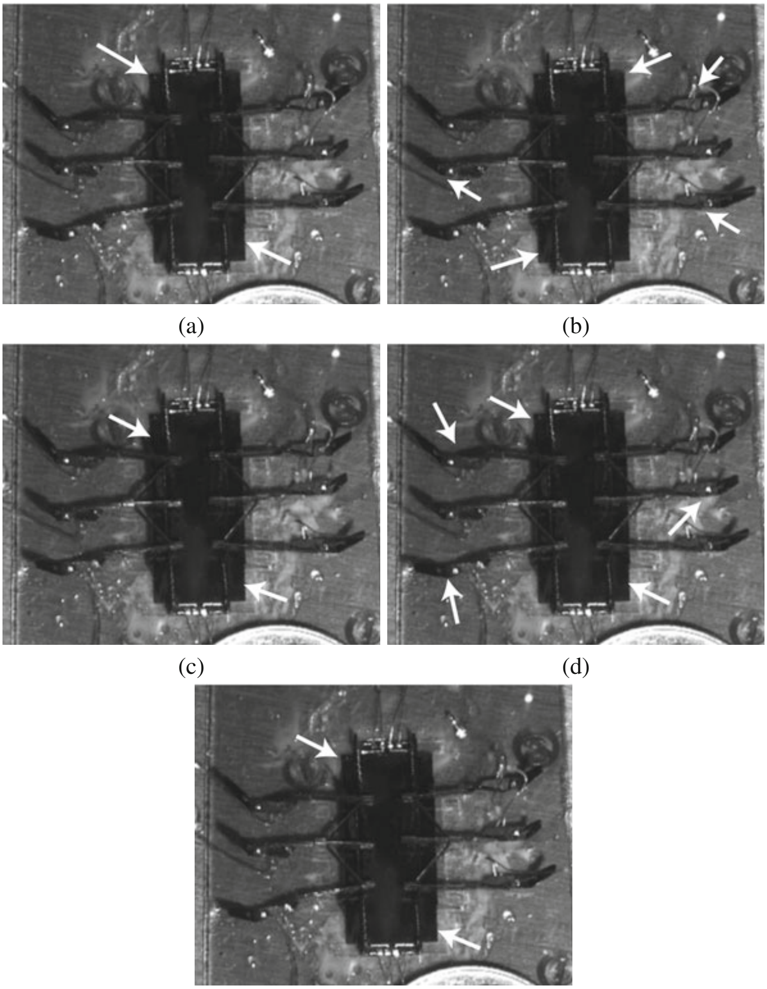
We were able to display the most fundamental gait to hexapod locomotion, the alternating tripod at 1/3Hz and 1Hz (See Fig. 11).

We were able to increase actuation speed up to 1 Hz, however this decreased leg displacements. Using ProAnalyst motion capture software we were able to track joint angles during a 1Hz alternating tripod gait. Fig. 13 plots transmission and leg joint angles over three periods, and tab. 1 gives the total measured sweep angle of each joint. By inspection, the 1Hz actuation cycle produces smaller proximal actuation displacements.

Table 1 Total measured sweep angle of each joint at 1Hz operation. All angles measured in degrees

Leg	1	2	3	4	5	6
Maximum Transverse Angle	2.4187	2.5458	3.2078	2.9191	3.8016	3.1618
Maximum Leg Angle	6.3407	5.1809	8.4887	8.4985	5.9961	5.8315

We were also able to demonstrate locomotion on a flat surface using the alternating tripod gait (Fig. 12) at approximately 1cm/min.



(a)

(b)

(c)

(d)

(e)

Fig. 11 The alternating tripod gait at 1/3Hz while suspended. Anterior is towards the bottom of the image, and arrows indicate powered actuators. a) All six legs are in stance while leg groupings change. b) Legs 1,3,5 are in swing. c) All six are again ground during a transition. d) Legs 2,4,6 are in swing. e) All six are again in stance at the end of the period.

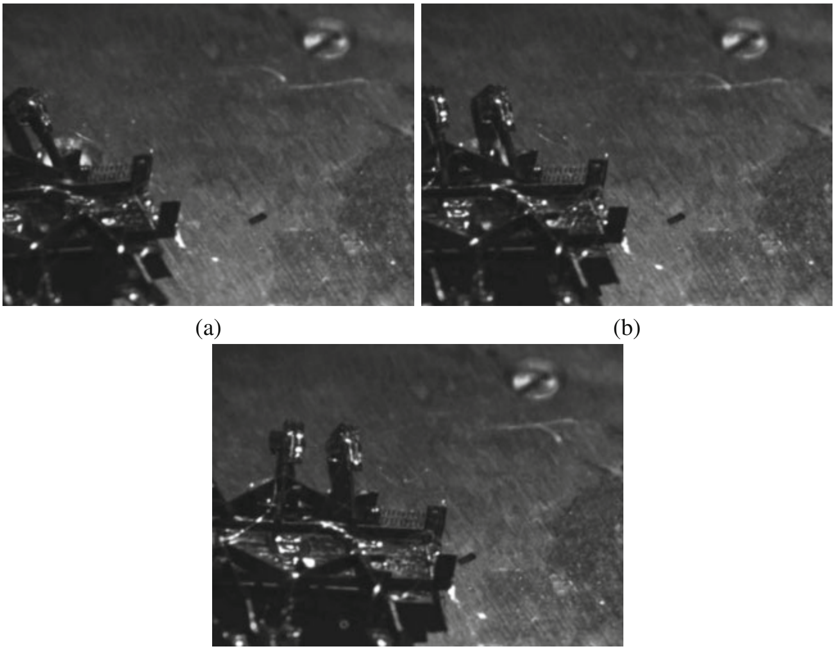
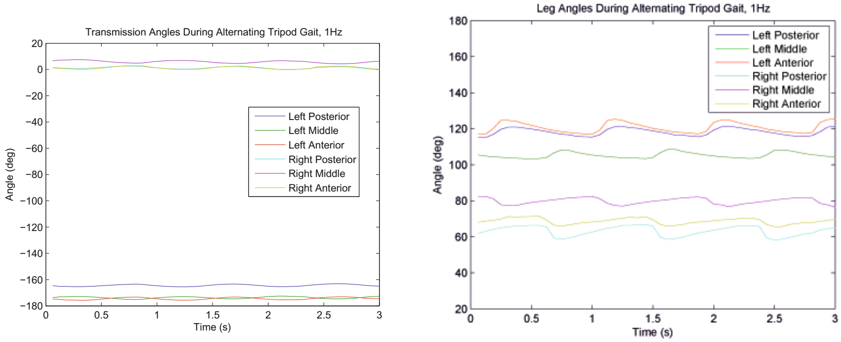

(a)
(b)
(c)

Fig. 12 Locomotion on a flat surface at 1/3Hz using the alternating tripod gait. a) at 0 seconds, b) at 15 seconds, c) at 30 seconds.



(a)
(b)

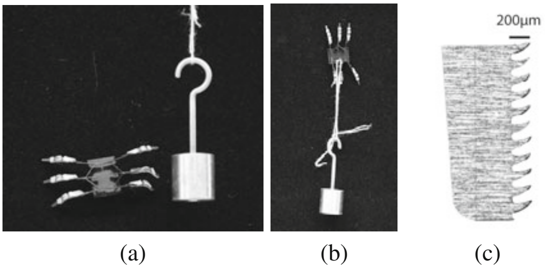
Fig. 13 a) Motion capture software is used to track HAMR's a) transmission angles and b) leg angles over three periods of 1Hz motion.

4.2 Turning

With a similar actuation scheme to alternating tripod, we are able to demonstrate the mechanics for a zero-radius turn. By maintaining identical tripod stance and swing groupings but actuating the transmissions in phase we obtain these mechanics. This turn requires one or more stance legs to slip. An alternative static turn mechanism is to pivot the body about one fixed ground leg, which is also achievable.

4.3 Attachment

This paper covers the numerous difficulties faced when designing a robotic insect, however it is important to note the benefits of small-scale robots with regards to wall attachment. Insects are capable of adhering to many vertical and inverted surfaces with attachment mechanisms such as tarsal claws [18], capillary adhesion [7] and Van Der Waals adhesion [2]. An un-actuated clawed version of HAMR illustrates the possibility of rough vertical surface attachment (Fig. 14a). Each leg includes an attachment mechanism containing eleven 200μm claws (Fig. 14b). They allow HAMR to support greater than 100 times its body weight while hanging from three legs on a rough felt surface (Fig. 14c). This experiment does not prove vertical locomotion but provides the foundation of research on one form of dry adhesion. Further research will explore the claw geometry [1] and body mechanics necessary to climb vertical surfaces of varying roughness.



(a) (b) (c)

Fig. 14 100 mg, unactuated, clawed version of HAMR. Outfitting the legs with attachment claws facilitates hanging a) in the direction of a vertical climb, b) while supporting a 10 g weight. c) Close-up of HAMR's 200μm claws.

5 Conclusions / Future Work

This paper has proven the feasibility of creating a hexapedal ambulatory microrobot with dynamic gait modulation. We have demonstrated the alternating tripod gait and zero-radius turns at 1/3Hz and 1Hz, and a simple mechanism for attachment to moderately rough surfaces. There is significant work yet to complete to create an autonomous, untethered, millimeter scale hexapod mobile robot capable of traversing any terrain, however this paper presents the initial steps.

5.1 Actuation and Control

Future work will include a thorough study of thin foil SMA actuators as well as exploration of other actuation alternatives such as piezoelectric ceramics and active fiber composites. As stated in Section 4.1, by increasing actuation speed from 1/3 to 1Hz we reduce thoracic actuator displacement, presumably because at the higher speed we do not allow actuator temperature to reach the required level. Therefore, we require a comprehensive model of thermodynamic and mechanical properties of our SMA spring to inform future designs. We must also model our distal actuators to achieve an appropriate force/displacement balance to clear a reasonable height during swing phases, yet produce enough force to overcome friction when lifting off the walking terrain.

As previously stated, HAMR will be untethered in future iterations by integrating an on-board power supply and microcontroller. Not only is this important for autonomy, but it is imperative to achieve forward locomotion since the tethering wires greatly outweigh the thorax and leg masses. Integrated power and control will also allow us to modulate the current to each of the 10 actuators, giving us more control over actuation timing. An onboard controller will also allow us to actuate with non-constant current. Actuator efficiency will increase with more complex control signals such as a short spike to quickly heat the SMA, followed by a lower, constant current to keep the actuator above its transition temperature [10]. This will have several effects such as reducing the thermal time constant of cooldown and prolonging the actuator lifetime.

5.2 Vertical Locomotion

Further research will explore the mechanics of wall-climbing, including attachment mechanisms for rough and smooth surfaces, transitions between orthogonal surfaces, and animal climbing mechanics.

Acknowledgements

The authors gratefully acknowledge The Charles Stark Draper Laboratory and the National Science Foundation (award number 0811571) for support of this work. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

1. Asbeck, A.T., Kim, S., Cutkosky, M.R., Provancher, W.R., Lanzetta, M.: Scaling hard vertical surfaces with compliant microspine arrays. The International Journal of Robotics Research 25(12), 1165 (2006)
2. Autumn, K., Sitti, M., Liang, Y.A., Peattie, A.M., Hansen, W.R., Sponberg, S., Kenny, T.W., Fearing, R., Israelachvili, J.N., Full, R.J.: Evidence for van der Waals adhesion in gecko setae. Proceedings of the National Academy of Sciences 99(19), 12252–12256 (2002)

3. Bergbreiter, S., Pister, K.S.J.: Design of an autonomous jumping microrobot. In: IEEE Int. Conf. on Robotics and Automation, Roma, Italy (April 2007)
4. Cowan, N.J., Ma, E.J., Cutkosky, M., Full, R.J.: A biologically inspired passive antenna for steering control of a running robot. Springer Tracts in Advanced Robotics 15, 541–550 (2005)
5. Donald, B.R., Levy, C.G., McGray, C.D., Rus, D.: An untethered, electrostatic, globally controllable mems micro-robot. J. of Microelectrical Mechanical Systems 15, 1–15 (2006)
6. Ebefors, T., Mattsson, J.U., Kälvesten, E., Stemme, G.: A walking silicon micro-robot. In: The 10th Int. Conf. on Solid-State Sensors and Actuators, Transducers 1999, Sendai, Japan, pp. 1202–1205 (June 1999)
7. Edwards, J.S., Tarkanian, M.: The adhesive pads of Heteroptera: a re-examination. In: Proceedings of the Royal Entomological Society of London. Series A, General Entomology, vol. 45, pp. 1–5. Blackwell Publishing Ltd., Malden (1970)
8. Full, R.J., Tu, M.S.: Mechanics of a rapid running insect: two-, four- and six-legged locomotion. J. of Experimental Biology 156, 215–231 (1991)
9. Goldman, D.I., Chen, T.S., Dudek, D.M.: Dynamics of rapid vertical climbing in cockroaches reveals a template. Journal of Experimental Biology 209(15), 2990 (2006)
10. Hoover, A.M., Steltz, E., Fearing, R.S.: Roach: An autonomous 2.4g crawling hexapod robot. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Nice, France (September 2008)
11. Kim, S., Asbeck, A.T., Cutkosky, M.R., Provancher, W.R.: Spinybotii: Climbing hard walls with compliant microspines. In: IEEE Int. Conf. on Advanced Robotics, Seattle, WA (July 2005)
12. Kim, S., Spenko, M., Trujillo, S., Santos, D., Cutkosky, M.R.: Smooth vertical surface climbing with directional adhesion. IEEE Transactions on Robotics 24, 65–74 (2008)
13. Lee, J., Sponberg, S.N., Loh, O.Y., Lamperson, A.G., Full, R.J., Cowan, N.J.: Templates and anchors for antenna-based wall following in cockroaches and robots. IEEE Transactions on Robotics 24(1) (February 2008)
14. Lewinger, W.A., Harley, C.M., Ritzman, R.E., Branicky, M.S., Quinn, R.D.: Insect-like antennal sensing for climbing and tunneling behavior in a biologically-inspired mobile robot. In: IEEE Int. Conf. on Robotics and Automation, Seattle, WA (July 2005)
15. Morrey, J.M., Lambrecht, B., Horchler, A.D., Ritzmann, R.E., Quinn, R.D.: Highly mobile and robust small quadruped robots. In: IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Las Vegas, NV (October 2003)
16. Raibert, M., Blankespoor, K., Nelson, G., Playter, R.: Bigdog, the rough-terrain quadruped robot. In: Proc. of the 17th World Congress, Seoul, Korea (July 2008)
17. Roth, L.M., Willis, E.R.: Tarsal structure and climbing ability of cockroaches. J. of Exp. Zool. 119(3), 483–517 (1952)
18. Roth, L.M., Willis, E.R.: Tarsal structure and climbing ability of cockroaches. Journal of Experimental Zoology 119(3) (1952)
19. Saranli, U., Buehler, M., Koditschek, D.E.: RHex - a simple and highly mobile hexapod robot. Int. J. of Robotics Research 20, 616–631 (2001)
20. SoftIntegration. Interactive four-bar linkage position analysis
21. Sponberg, S., Full, R.J.: Neuromechanical response of musculo-skeletal structures in cockroaches during rapid running on rough terrain. J. of Experimental Biology 211, 433–446 (2008)

22. Teasdale, D., Milanovic, V., Chang, P., Pister, K.S.J.: Microrockets for smart dust. Smart Materials and Structures 6, 1145–1155 (2001)
23. Trimmer, W.S.N.: Microrobots and micromechanical systems. J. of Sensors and Actuators 19, 267–287 (1989)
24. Wood, R.J.: The first flight of a biologically-inspired at-scale robotic insect. IEEE Transactions on Robotics 24(2) (April 2008)
25. Wood, R.J., Avadhanula, S., Sahai, R., Steltz, E., Fearing, R.S.: Microrobot design using fiber reinforced composites. J. of Mech. Design 130(5) (May 2008)
26. Wood, R.J., Avadhanula, S., Steltz, E., Seeman, M., Entwistle, J., Bacharach, A., Barrows, G., Sanders, S., Fearing, R.S.: Design, fabrication and initial results of a 2g autonomous glider. In: Conf. on IEEE Industrial Electronics, Raleigh, NC (November 2005)
27. Yeh, R., Hollar, S., Pister, K.S.J.: Design of low-power silicon articulated microrobots. J. of Micromechatronics 1(3), 191–203 (2002)

Assembly and Disassembly of Magnetic Mobile Micro-Robots towards 2-D Reconfigurable Micro-Systems

Chytra Pawashe*, Steven Floyd⋆, and Metin Sitti

Abstract. A primary challenge in the field of reconfigurable robotics is scaling down the size of individual robotic modules. We present a novel set of permanent magnet modules that are 900 μm × 900 μm × 270 μm in size, called Mag-μMods, for use in a reconfigurable micro-system. The module is actuated by oscillating external magnetic fields less than 5 mT in strength, and is capable of locomoting on a 2-D surface. Multiple modules can be controlled by using an electrostatic anchoring surface, which can selectively prevent specific modules from being driven by the external field while allowing others to move freely. We address the challenges of both assembling and disassembling two modules. Assembly is performed by bringing two modules sufficiently close that their magnetic attraction causes them to combine. Disassembly is performed by electrostatically anchoring one module to the surface, and applying magnetic forces or torques from external sources to separate the unanchored module.

1 Introduction

The field of reconfigurable robotics proposes versatile robots that can reconfigure into various configurations depending on the task at hand [1]. These types of robotic systems consist of many independent and often identical modules, each capable of motion, and capable of combining with other modules to create assemblies. These modules can then be disassembled and reassembled into alternate configurations. For example, Shen et al. [2] demonstrate SuperBot; this robot consists of 20 modules that can combine to form a mobile mechanism that can roll across the ground for 1 km and then reconfigure into one that can climb obstacles.

Chytra Pawashe · Steven Floyd · Metin Sitti
Carnegie Mellon University, Dept. Mechanical Engineering
5000 Forbes Ave, Pittsburgh, PA 15213, USA
e-mail: {csp,srfloyd,msitti}@andrew.cmu.edu

⋆ Equally contributing co-authors.

C. Pradalier, R. Siegwart, and G. Hirzinger (Eds.): Robotics Research, STAR 70, pp. 731–747.
springerlink.com

Another concept in the field of reconfigurable robotics is *programmable smart matter*, which is matter that can assemble and reconfigure into arbitrary three-dimensional (3-D) shapes, giving rise to *synthetic reality* [3]. This is similar to virtual or augmented reality, where a computer can generate and modify an arbitrary object. However, in *synthetic reality*, this object has physical realization. A primary goal for programmable matter is scaling down the size of each individual module, with the aim of increasing spatial resolution of the final assembled product. Currently, the smallest actuated module in a reconfigurable robotic system fits inside a 2 cm cube [4], which is a self-contained module that is actuated using shape memory alloy. Scaling down further into the sub-millimeter scale brings new issues, including module fabrication, control, and communication. Micro-robotics technologies of the past few years have been progressing, with the introduction of untethered mobile micro-robots under 1 mm in size; these robots can potentially be used as micron-scale modules. The micro-robots that operate on two-dimensional (2-D) surfaces in the literature can be controlled either electrostatically [5], electromagnetically [6, 7], or using laser thermal excitation [8]. 3-D swimming micro-robots are also possible, and are often electromagnetically controlled [9, 10], and can even be powered by bacteria [10, 11].

For the purposes of micron-scale assembly using micro-robots, Donald et al. [5] demonstrate the assembly of four MEMS-fabricated silicon micro-robots, each under 300 μm in all dimensions, actuated by electric fields. Once assembled however, they cannot detach and reconfigure, because the electrostatic driving fields do not allow for disassembly. As a result, disassembling micron-scale modules is currently an unsolved problem.

2 Concept: Reconfigurable Micro-Robotics

In this work, we propose using sub-millimeter untethered permanent magnet micro-robots (Mag-μBots) actuated by external magnetic fields [6] as components of magnetic micro-modules (Mag-μMods), for creating deterministic reconfigurable 2-D micro-assemblies; this implies that the Mag-μMods will be able to both assemble and disassemble. Permanent magnet modules will attract each other with large magnetic forces, therefore it is necessary to reduce this inter-magnet force to facilitate disassembly. This is done by adding an outer shell to the Mag-μBot for the design of a module. The outer shell prevents two magnetic modules from coming into close contact, where magnetic forces will become restrictively high. A schematic of a Mag-μMod is illustrated in Fig. 1(a).

Motion of multiple Mag-μMods is achieved by employing a surface divided into a grid of cells, where each cell on the surface contains an addressable electrostatic trap capable of anchoring individual Mag-μMods to the surface and preventing them from moving. Unanchored Mag-μMods can move on the surface due to the imposed magnetic fields, and move in parallel. This technique is identical to controlling multiple Mag-μBots, explained in detail in [12]. Assembling two Mag-μMods is straightforward - by moving an unanchored Mag-μMod towards an

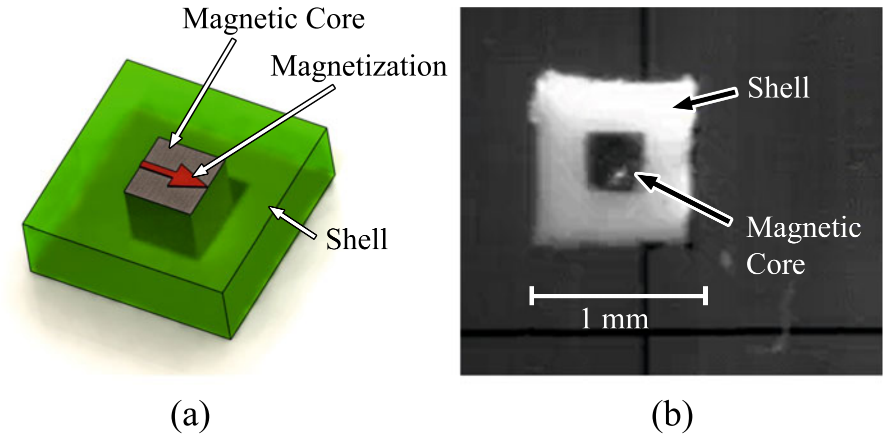


Fig. 1 (a) Schematic of a Mag-μMod. A permanent magnetic core with indicated magnetization is surrounded by a magnetically inactive shell. (b) Photograph of a Mag-μMod. A 300 × 300 × 170 μm^3 Mag-μBot is used as the magnetic core of the module, with a polyurethane shell 900 μm × 900 μm × 270 μm in outer dimensions surrounding the core.

anchored one, magnetic forces eventually dominate and cause the two Mag-μMods to self-assemble.

Disassembling two Mag-μMods is problematic. For this to occur, the magnetic force between two Mag-μMods must be overcome and the two separated without physically contacting either one. To do this, we use the electrostatic grid surface to anchor parts of assembled modules, and examine the effectiveness of both externally applied magnetic forces and torques to disassemble unanchored modules on the assembly.

Figure 2 displays the concept of multiple Mag-μMods assembling, disassembling, and reconfiguring into different configurations. Because the Mag-μMods are magnetic, they can only assemble into configurations that are magnetically stable.

3 Experimental Setup, Operation, and Fabrication

Mag-μBots are actuated by six independent electromagnetic coils, aligned to the faces of a cube approximately 11 cm on a side, with horizontal and vertical coils capable of producing maximum field strengths at the position of the Mag-μBot (see Fig. 3) of 3.0 mT and 2.3 mT, respectively. Control of the electromagnetic coils is performed by a PC with a data acquisition system at a control bandwidth of 1 kHz, and the coils are powered by custom-made electronic amplifiers.

Actuation of each Mag-μBot is accomplished by using two or three electromagnetic coils. One or more horizontal coils are first enabled (coil D in Fig. 3), causing the Mag-μBot to orient in the direction of the net magnetic field. The magnetic force exerted by the coils on the Mag-μBot is insufficient to translate it, due to

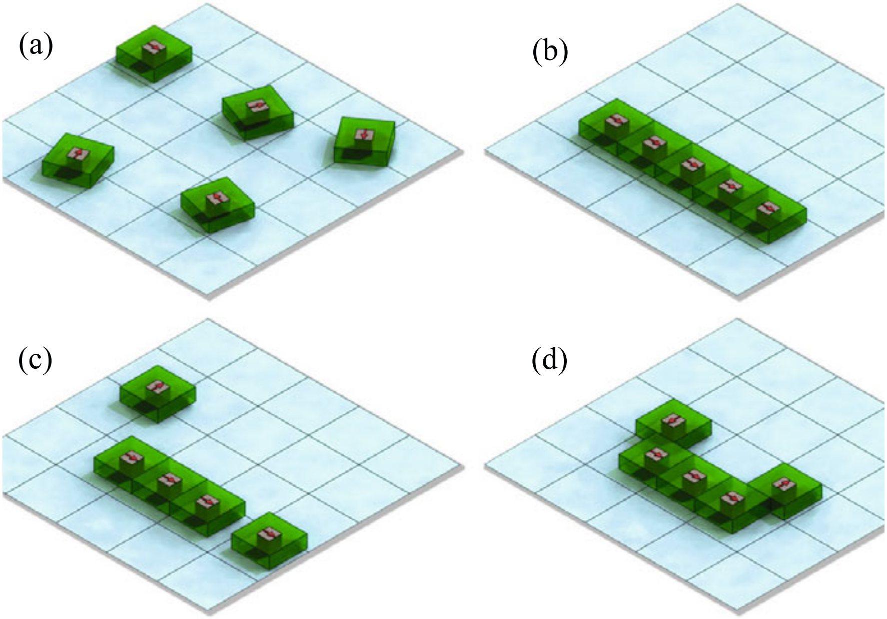


Fig. 2 Schematic of five Mag-μMods operating on an electrostatic grid surface, where each cell can be individually activated to anchor-down individual Mag-μMods; unanchored Mag-μMods can be moved by the global magnetic field. (a) The five Mag-μMods are separate, and (b) assemble into a magnetically-stable line. In (c), the two outer Mag-μMods disassemble from the line, and (d) reconfigure into a magnetically stable 'U' shape.

friction from the surface. Thus, vertical clamping coils (coils C and F in Fig. 3) are enabled and pulsed using a sawtooth waveform. This results in a non-uniform rocking motion of the Mag-μBot, which induces stick-slip motion across the surface. In general, the Mag-μBot's velocity increases with pulsing frequency, typically from 1-100 Hz, and can exceed velocities of 16 mm/s in air. The Mag-μBot is also capable of operating in fluids of viscosities less than about 50 cSt, and can operate on a variety of smooth and rough magnetically inactive surfaces, provided that the adhesion between the Mag-μBot and surface is low. Further explanation of this system is discussed in [6, 12, 13] and demonstration movies can be found online at [14].

3.1 Mag-μBot and Mag-μMod Fabrication

Mag-μBots can be produced in a batch process using soft-lithography techniques in a manner similar to [15]. The Mag-μBot used in this work is rectilinear, 300 × 300 × 170 μm^3, and is composed of a mixture of neodymium-iron-boron (NdFeB) particles (Magnequench MQP-15-7, refined in a ball mill to produce particles under

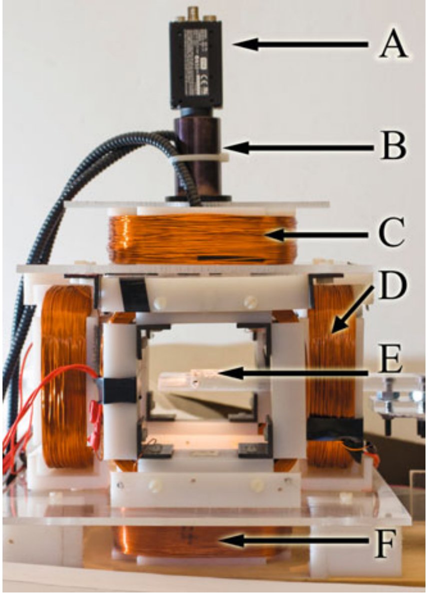


Fig. 3 Photograph of the electromagnetic coil setup, where A is the camera for visual feedback, B is the microscope lens, C is the top coil, D is one of four upright coils that orients the Mag-μBot within the plane on the surface, E is the surface on which the Mag-μBot locomotes, and F is the bottom coil. The top and bottom coils are clamping coils, which provide a clamping force and a torque that pushes and orients the Mag-μBot towards the surface, respectively.

2 μm in size) suspended in a polyurethane (TC-892, BJB enterprises) matrix. The fabrication process used is shown in Fig. 4.

A Mag-μMod is a polyurethane shell encasing a magnetic core, the Mag-μBot. The shells are fabricated in a manner similar to the Mag-μBot, omitting the addition of NdFeB powder into the mold mixture and the magnetization step. Assembly of the Mag-μBot into a shell is performed manually using tweezers under an optical microscope, and the two components are held together by a pressure-fit (an adhesive can also be used to bind the two components). Figure 1(b) displays an assembled Mag-μMod. In the presence of the global magnetic fields, these modules move similarly to individual Mag-μBots without shells, exhibiting stick-slip motion across the working surface, however at lower velocities of about 0.5 mm/s.

3.2 Electrostatic Grid Surface Fabrication

The electrostatic grid surface, described in [12], is necessary to enable the control of multiple Mag-μBots or Mag-μMods. It consists of an array of independently addressable pads, each pad containing a set of interdigitated electrodes to generate

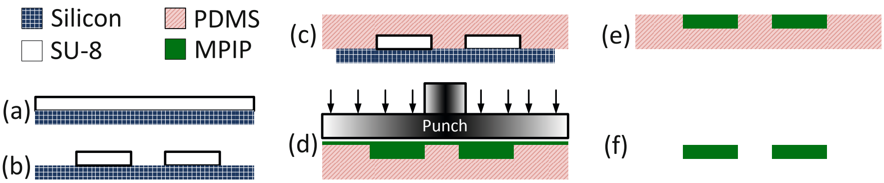


Fig. 4 The fabrication steps used to batch manufacture polymer Mag-μBots. (a) SU-8 is spin coated onto a silicon wafer to the desired thickness of the final micro-robot, and (b) is patterned and hardened to create the positive mold. (c) Polydimethylsiloxane (PDMS, Dow Corning HS II RTV) mold making material is poured onto the positive mold and allowed to cure. (d) The PDMS is removed from the positive mold, creating the negative mold, and a mixture of magnetic-powder-impregnated polyurethane (MPIP) is prepared by mixing 4 parts NdFeB powder to 1 part polyurethane, degassed in a vacuum, and then poured onto the PDMS mold. A large permanent magnet (not shown) is placed under the PDMS mold to ensure the NdFeB powder is densely packed in the mold. After a second degassing, a polypropylene flat punch is pressed and held against the mold, which pushes excess NdFeB-polyurethane out, leaving a thin backing layer. Next, the large magnet is moved to the front of the mold so that the NdFeB particles will orient along the lengths of the Mag-μBots, facilitating a higher net magnetization in the length-wise direction, as magnetic domains will be more favorably oriented. (e) After hardening and removing the punch, excess polyurethane is peeled off manually using tweezers. (f) Finished polymer Mag-μBots can be easily removed from the mold with tweezers or micro-probes, and are later magnetized in a 1 T magnetizing field along its length. This results in a magnetization of about 58 kA/m, estimated from measurements using a vibrating sample magnetometer (ADE Technologies Inc.).

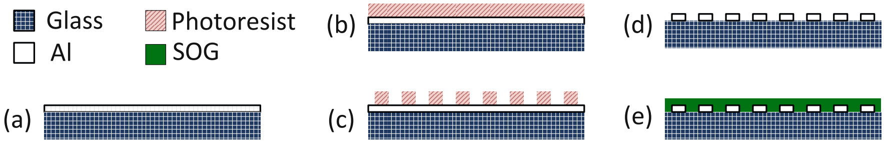


Fig. 5 Steps for fabricating an interdigitated electrostatic anchoring surface used for the control of multiple Mag-μMods. (a) 100 nm of aluminum is sputtered onto a glass substrate. (b) Photoresist is spun onto the metal surface, and patterned in (c) with a spacing of 10 μm. The aluminum layer is etched in (d) to create the interdigitaded electrode pattern. For this work, four 2 mm × 2 mm electrode pads are patterned in a 2 × 2 grid configuration. After creating interconnects from the electrodes to external electronics (not shown), a 400 nm spin-on glass (SOG) insulation layer is deposited in (e). Mag-μMods operate directly on this spin-on glass surface.

high electric fields. The surface is fabricated with the steps shown in Fig. 5. Mag-μMods are placed on this surface and are operated in a low-viscosity silicone oil (Dow Corning 200 fluid, 20 cSt) which supports the generation of the large electric fields required to anchor individual Mag-μMods. Anchoring occurs through a capacitive coupling force to the surface for conductive materials.

4 Modeling

Each of the Mag-μMods is subject to magnetic forces created by the driving magnetic field and other Mag-μMods, electrostatic anchoring forces created by the grid surface, and surface forces such as adhesion and friction. In this section, these different forces are briefly modeled to provide insight to their relative magnitude and effect on both assembly and disassembly of Mag-μMods.

4.1 Magnetic Influences

Each magnetized Mag-μMod and each electromagnet creates a magnetic field: $B_{mr}(x,y,z)$ and $B_{ec}(z)$, respectively. Within these magnetic fields, magnetized Mag-μMods experience both a torque and a force. This magnetic torque is proportional to the magnetic field strength, and acts to bring the internal magnetization of the Mag-μMod into alignment with the field. Magnetic force is proportional to the gradient of the magnetic field, and acts to move Mag-μMods to a local maximum. The relations that govern these interactions are:

$$\mathbf{T}_m = V_m \mathbf{M} \times \mathbf{B}(x,y,z) \tag{1}$$

$$\mathbf{F}_m = V_m (\mathbf{M} \bullet \nabla) \mathbf{B}(x,y,z) \tag{2}$$

where $\mathbf{T}_m$ is the torque the Mag-μMod experiences, V_m is the volume of the Mag-μMod's magnetic core, $\mathbf{M}$ is the magnetization of the Mag-μMod's magnetic core, and $\mathbf{F}_m$ is the force the Mag-μMod experiences.

To approximate the forces and torques Mag-μMods exert on each other, they are modeled as magnetic dipoles, located at their geometric center, with a value equal to their core's magnetic moment. Due to the symmetry inherent in this approximation, it is convenient to describe the field produced by the Mag-μMod's core in spherical coordinates:

$$\mathbf{B_{mr}}(r,\theta) = \frac{\mu_0}{4\pi} V_m \mathbf{M} \left(\frac{2cos(\theta)}{r^3} \mathbf{e}_r + \frac{sin(\theta)}{r^3} \mathbf{e}_\theta \right) \tag{3}$$

where $\mu_0 = 4\pi \times 10^{-7}$ H/m is the permeability of free space, $M = 58$ kA/m is the estimated magnetization of each Mag-μBot, $\mathbf{e}_r$ is a unit vector aligned with the magnetization of the Mag-μBot, $\mathbf{e}_\theta$ is a unit vector representing rotations about the y-axis, r is the distance from the center of the Mag-μMod, and θ is the azimuthal angle. Coordinate conventions are shown in Fig. 6. The attractive force that arises between two aligned Mag-μMods when separated by a distance r is:

$$F_{mr}(r) = \frac{\mu_0}{4\pi} (V_m M)^2 \frac{6}{r^4} \tag{4}$$

The magnetic field created by each of the electromagnets is directed towards the coil and is experienced by all Mag-μMods:

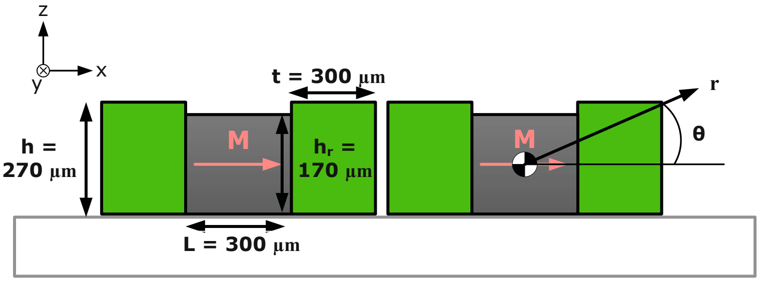


Fig. 6 Side view cross-section schematic of two Mag-μMods. Relevant geometry and coordinates used in equations are shown.

$$B_{ec}(z') \approx \frac{\mu_0 N I a^2}{2(z'^2 + a^2)^{3/2}} = 9.49 \times 10^{-4}(I) \left[\frac{\mathrm{T}}{\mathrm{A}}\right] \tag{5}$$

where, $z' = 0.095$ m is the approximate distance from the center of an electromagnet to the workspace, $N = 510$ is the number of turns, I is the current flowing through it, and $a = 0.0695$ m is its characteristic radius [16]. From this, the gradient produced by a single electromagnet can be derived:

$$\left|\Delta B_{ec}(z')\right| \approx \frac{3\mu_0 N I a^2 z'}{2(z'^2 + a^2)^{5/2}} = 1.95 \times 10^{-2}(I) \left[\frac{\mathrm{T}}{\mathrm{A \cdot m}}\right] \tag{6}$$

If two electromagnets are used in opposition, the combined field gradient is larger:

$$\left|\Delta B_{2ec}(z')\right| \approx \frac{\mu_0 N I}{2az'} = 4.85 \times 10^{-2}(I) \left[\frac{\mathrm{T}}{\mathrm{A \cdot m}}\right] \tag{7}$$

4.2 Electrostatic Influences

A conductive object with non-negligible surface roughness operating on an insulating surface covering a set of interdigitated electrodes at an applied voltage difference of V_{id} will assume a potential halfway between the two, or $\frac{1}{2}V_{id}$, if it overlaps equal areas of electrodes at both voltages; applicable when the electrode and gap widths (10 μm each) are much smaller than the length of the object (300 μm). For the conductive magnetic core of a Mag-μMod, its surface roughness traps a fluid layer with thickness comparable to the its maximum asperity height of about $b = 9$ μm beneath it, increasing the separation from the electrodes and causing the total capacitance C_{tot} between the Mag-μMod and the electrodes to be:

$$C_1 = \frac{\varepsilon_0 \varepsilon_{r1} A_{id}}{g} \approx 3.48 \times 10^{-12}\ [\mathrm{F}] \tag{8}$$

$$C_2 = \frac{\varepsilon_0 \varepsilon_{r2} A_{id}}{b} \approx 1.11 \times 10^{-13}\ [\mathrm{F}] \tag{9}$$

$$C_{tot} = \left(C_1^{-1} + C_2^{-1}\right)^{-1} \approx 1.07 \times 10^{-13} \text{ [F]} \tag{10}$$

where C_1 is the capacitance associated with the glass insulation layer, $\varepsilon_0 = 8.854 \times 10^{-12}$ F/m is the permittivity of free space, $\varepsilon_{r1} = 3.5$ is the relative permittivity of the glass surface, $A_{id} = 4.5 \times 10^{-8}$ m^2 is half the apparent area of the overlap between the Mag-μMod's magnetic core and the electrodes, C_2 is associated with the fluid gap, and $\varepsilon_{r2} = 2.5$ is the relative permittivity of the silicone oil environment. Using the principal of virtual work, the electrostatic anchoring force with a fluid gap, F_e, will be:

$$F_e = \frac{1}{16} V_{id}^2 C_{tot}^2 \left[\frac{1}{C_1 g} k + \frac{1}{C_2 a}(1-k)\right] \geq 5.16 \times 10^{-10} (V_{id}^2) \left[\frac{\text{N}}{\text{V}^2}\right] \tag{11}$$

where k is an empirical constant ($0 < k < 1$) used to bound the solution space.

To anchor a Mag-μMod to the surface, the electrostatic anchoring force must suppress three effects: (1) out-of-plane rotations about an axis in the $x-y$ plane of the Mag-μMod from magnetic fields in the z-direction, (2) rotations within the plane of motion about the z-axis due to magnetic fields in the $x-y$ direction, and (3) translation due to magnetic field gradients in a direction in the $x-y$ plane.

During stick-slip locomotion, each Mag-μMod can experience fields of up to 4.82 mT when using multiple coils. This creates out-of-plane magnetic torques up to 4.28×10^{-9} N$\cdot$m on the Mag-μMod, which can be approximated as a pair of 14.3 μN forces acting in opposite directions on opposite sides of the Mag-μMod. To prevent this rotation, the electrostatic surface must exert double this force, or approximately 28.6 μN, applied at the centroid of the Mag-μMod. Using Eq. (11), this force requires $V_{id} = 235$ V applied to the electrostatic surface.

4.3 Surface Forces

While immersed in silicone oil, Mag-μMods experience reduced adhesion forces to the surface [17, 13] when compared to air. Adhesion to the surface while immersed in silicone oil can be taken to be negligible. Thus, the maximum friction force, $F_{f,max}$, experienced by a Mag-μMod is determined using a Coulomb friction relation:

$$F_{f,max} = \mu_f (W + F_e + F_m) \tag{12}$$

where μ_f is the coefficient of friction, W is the Mag-μMod's buoyant weight, F_e is any electrostatic anchoring force, and F_m is any magnetic force that pushes the Mag-μMod toward the surface. Based upon the densities of polyurethane (1140 kg/m^3) and NdFeB (7400 kg/m^3), and assuming close packing of NdFeB spheres, the effective density of a Mag-μMod is 5770 kg/m^3. When immersed in silicone oil (density 950 kg/m^3), the buoyant weight $W = 1.09$ μN. From empirical results in Sec. 7, $F_{f,max} = 228$ nN for a Mag-μMod, when $F_e = F_m = 0$; thus using Eq. (12), $\mu_f = 0.21$ in the experiments.

5 Mag-μMod Assembly

As two Mag-μMods approach each other, the forces and torques between them increase due to the r^3 dependence of their fields, from Eq. (3), and they will eventually combine into the configuration shown in Fig. 6. Using the dipole approximation (Eq. (4)), the distance where two Mag-μMods jump-into-contact can be estimated by determining when the inter-magnetic force overcomes the friction force, based upon the Mag-μMod's buoyant weight. Conversely, given the jump-into-contact distance, the friction coefficient can be estimated.

When two Mag-μMods are assembled, they are separated by two shells, each one approximately 300 μm thick; thus their center-to-center distance is approximately 900 μm. From Eq. (4), the mutual attractive force is 720 nN. The assembled structure can continue to locomote, as shown in Fig. 10(d).

6 Mag-μMod Disassembly

To disassemble two Mag-μMods that are assembled, the application of external magnetic forces, torques, or both may be utilized. In this work, three methods for disassembly are described, and the necessary forces and torques for disassembly, as well as necessary anchoring forces, are derived.

Method 1 - Translation Disassembly: To separate two assembled Mag-μMods by translating one away from another, one Mag-μMod must first be anchored to the surface. Next, the global magnetic field gradient acting upon the unanchored Mag-μMod must overcome the local magnetic field gradient created by the anchored Mag-μMod, and the friction force, shown in Fig. 7. From Sec. 5, two Mag-μMods have an attraction force of 720 nN. Combined with the friction force of 228 nN, the total lateral force which must be overcome is 948 nN, this corresponds to a necessary field gradient of 1.07 T/m from Eq. (2). To produce this gradient from a single coil in Fig. 3, 55 amps would have to be passed through an electromagnet (Eq. (6)). Alternatively, using two coils separated by 19 cm, 22 amps per coil (or 44 amps total) would be required (Eq. (7)).

The anchored Mag-μMod experiences forces from both the electromagnet and the other Mag-μMod, totaling 1.67 μN. This force must be balanced by a friction force to the surface, which ensures that this Mag-μMod remains stationary. Using a friction coefficient of $\mu_f = 0.21$ and Eq. (12), with $F_m = W = 0$ to attain a conservative estimate, $F_e > 7.94$ μN is necessary to remain stationary. This corresponds to $V_{id} > 124$ V (Eq. (11)).

Method 2 - In-Plane Rotation Disassembly: Applied magnetic torques can be used to separate two assembled Mag-μMods. As in Method 1, one of the Mag-μMods must be electrostatically anchored to remain stationary. Then, the in-plane magnetic field is rotated about the z-axis, twisting the unanchored Mag-μMod about its point of contact with the anchored Mag-μMod. The unanchored Mag-μMod is

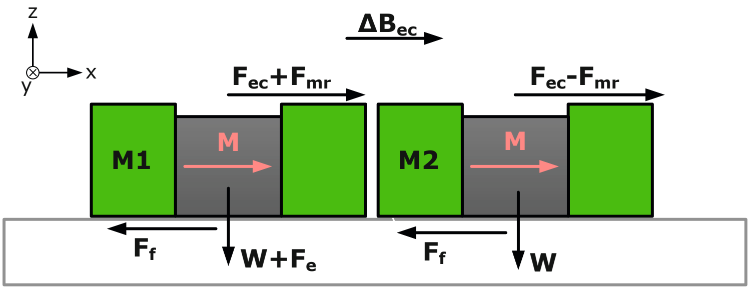


Fig. 7 Side view cross-section schematic of two Mag-μMods being disassembled via translation disassembly and the relevant forces from the electromagnetic coils (F_{ec}) and from other Mag-μMods (F_{mr}) acting upon them. M1 is anchored to the surface while M2 is pulled by the externally-induced field gradient ΔB_{ec}.

rotated until it is in a configuration with the anchored Mag-μMod that minimizes their attractive force. Once this orientation is achieved, the unanchored Mag-μMod can move away using its standard locomotion method using pulsed magnetic fields, leaving the anchored Mag-μMod in place. Fig. 8 displays a schematic of this disassembly technique.

The applied magnetic torque that rotates the unanchored Mag-μMod must overcome the torque created by the anchored Mag-μMod, which acts to align the two. This torque is calculated as a function of θ and ϕ as one Mag-μMod rotates about the other at a constant distance $r = 900$ μm, shown schematically in Fig. 8. Using Eqs. (1) and (3), the maximum torque is 2.16×10^{-10} N $\cdot$ m, and occurs when $\theta = 0$, $\phi = [\pi/2, 3\pi/2]$. A weight-based friction torque of 5.13×10^{-11} N $\cdot$ m, acting at the centroids of opposite sides of the Mag-μMod, is added to the magnetic torque, as it must too be overcome. To counteract this torque, using Eq. (5), a single electromagnet from Fig. 3 must impose a field of $B_{ec} = 301$ μT, which corresponds to $I = 0.32$ A.

To determine the necessary anchoring force, both the electromagnet's applied torque and the rotating Mag-μMod's torque are applied on the stationary Mag-μMod. To resist these moments, the electrostatic-based friction is treated as a force couple acting at the half body centroids of the Mag-μBot, as it is the only part of the Mag-μMod being pulled down. Each force in this force pair must exceed 3.22 μN, leading to a conservative total anchoring force of $F_e = 30.7$ μN that must be applied; from Eq. (11), $V_{id} = 244$ V.

Method 3 - Out-of-Plane Rotation Disassembly: Another approach for using applied torques to separate two Mag-μMods requires that one of the Mag-μMods be anchored to the surface, while the other Mag-μMod is unanchored and rotated about an axis in the $x-y$ plane, shown schematically in Fig. 9. The unanchored Mag-μMod rotates until the mutual attractive force between the two Mag-μMods is minimized, or becomes repulsive. When this orientation achieved, the unanchored Mag-μMods can move away using its standard locomotion method.

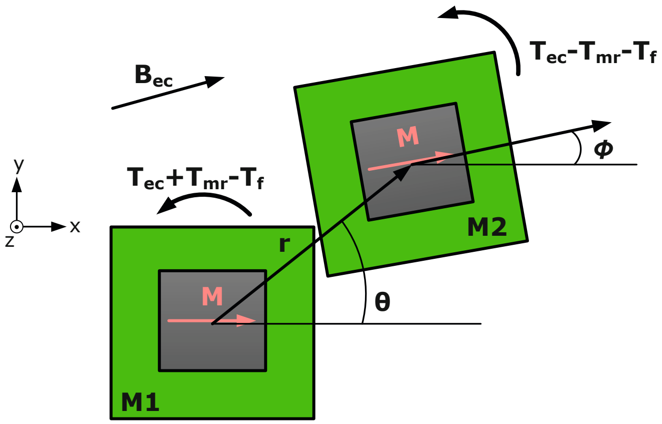


Fig. 8 Top view schematic of two Mag-μMods being disassembled via in-plane rotation disassembly, and the relevant torques from the electromagnetic coils (T_{ec}), from other Mag-μMods (T_{mr}), and from friction (T_f) acting upon them. M1 is anchored to the surface while M2 is disassembled by rotating it using the global electromagnet-induced field (B_{ec}).

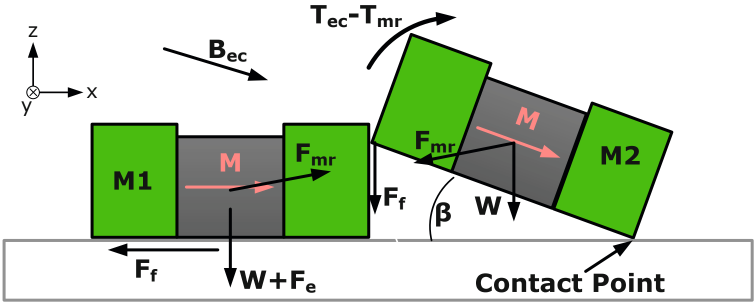


Fig. 9 Side view cross-section schematic of two Mag-μMods being disassembled via out-of-plane rotation disassembly, the relevant torques from the electromagnetic coils (T_{ec}), from other Mag-μMods (T_{mr}), and the friction forces (F_f) acting upon them. M1 is anchored to the surface while M2 is disassembled by rotating it using the global electromagnet-induced field (B_{ec}).

Assuming that the point of contact for the rotating Mag-μMod remains fixed (see Fig. 9), the maximum torque required to rotate it upward is 7.98×10^{-10} N $\cdot$ m, and occurs at an angle of $\beta = 11.5°$. This torque is a function of the magnetic force and torque exerted on the unanchored Mag-μMod by the anchored Mag-μMod (see Eqs. (1), (2) and (3)), the weight, and the contact friction between the two (see Eqs. (4) and (12)). After passing a critical angle, $\beta_{crit} = 69.3°$, the force between the two Mag-μMods becomes repulsive.

To overcome this maximum torque, a single electromagnet has to produce a field of $B_{ec} = 899\ \mu T$ using Eq. (5), implying $I = 0.95$ A. For electrostatic anchoring to effectively resist the torques from both the magnetic field and the moving Mag-μMod, a force of $F_e = 10.6\ \mu$N must be applied with $V_{id} = 144$ V from Eq. (11).

7 Experimental Results and Discussion

In the experiments, two Mag-μMods are placed on an electrostatic grid surface with four anchoring pads, in a 2×2 configuration, and operated in a silicone oil environment. Motion is achieved by pulsing the electromagnetic coils from 1-3 Hz using a sawtooth waveform. An anchoring voltage of $V_{id} = 400$ V is used, which is greater than any of the estimated requirements presented in Sec. 6, and ensures proper anchoring. Movies of assembly and disassembly can be found online [14].

Assembly: The process of Mag-μMods assembling is demonstrated in Fig. 10. In this experiment, the magnetic attraction between the Mag-μMods is sufficient to combine them when their center to center distance is 1.2 mm. Using Eq. (4), This corresponds to an attractive force of 228 nN. The assembled Mag-μMod structure is capable of motion, shown in Fig. 10(d).

Two Mag-μMods can also combine in 3-D when one slides over the other, resulting in a cuboid structure. In this configuration, the Mag-μMods are much closer than the case in Fig. 10, and as a result are held together with a greater force. Disassembly techniques for this 3-D configuration will be investigated in future works.

Method 1 - Translation Disassembly: Fig. 11 demonstrates two Mag-μMods being disassembled. Due to hardware limitations of the electromagnetic coil system, a permanent NdFeB magnet $6.3 \times 6.3 \times 31.8$ mm^3 (N42 grade, magnetized along its length) was placed approximately 1.5 cm from the workspace, and is used to generate the necessary large magnetic field gradient. When the permanent magnet is brought near the workspace, its field gradient exceeds that of the anchored Mag-μMod, forcing the unanchored Mag-μMod to separate.

Method 2 - In-Plane Rotation Disassembly: Disassembly of two Mag-μMods using torques exerted by the electromagnetic coils is demonstrated in Fig. 12. One Mag-μMod is electrostatically anchored while the other is rotated about the z-axis to minimize the attractive force. Once in a configuration with nearly zero attractive force between them ($\theta = \pi/2$, $\phi = \pi/2$ using the convention of Fig. 8), the unanchored Mag-μMod is moved away from the anchored Mag-μMod using a standard locomotion signal from the electromagnets.

Method 3 - Out-of-Plane Rotation Disassembly: In Fig. 13, a magnetic field was created by placing a 1.27 cm cube-shaped NdFeB (grade N42) permanent magnet approximately 9 cm beneath the surface, and the unanchored Mag-μMod rotated out-of-plane about its contact point with the surface. When the angle of rotation is sufficiently large, the force between the two two Mag-μMods becomes repulsive, and the unanchored Mag-μMod can be disassembled by moving it away with a standard locomotion signal from the electromagnets.

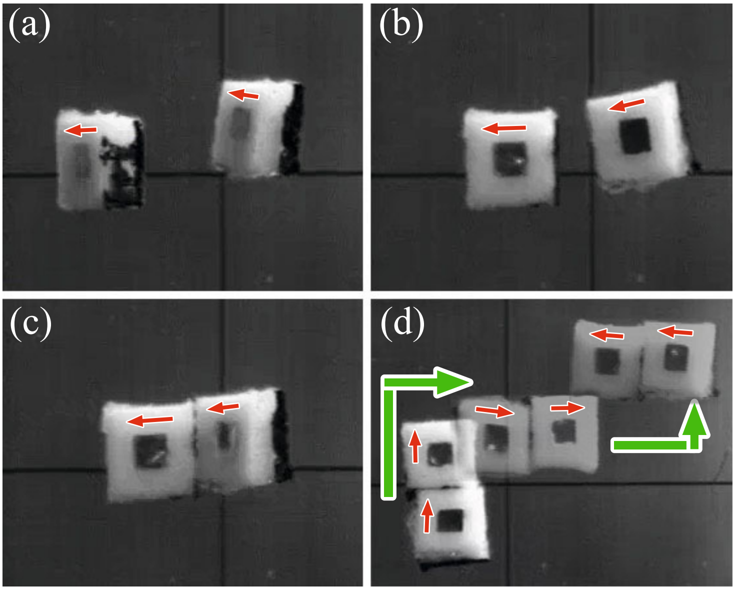


Fig. 10 Frames from a video of two Mag-μMods assembling; arrows on Mag-μMods indicate magnetization direction. In (a), the Mag-μMods are initially separated, and oriented towards the surface by the electromagnetic clamping field, which is turned off in (b). The Mag-μMods attract each other and assemble in (c). The assembled Mag-μMods move together in (d), shown as superimposed frames.

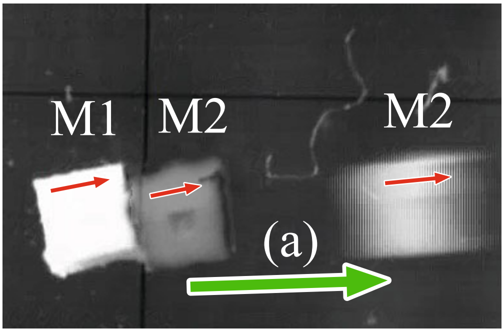


Fig. 11 Superimposed frames from a video of two Mag-μMods, M1 and M2, disassembling by manually bringing in a strong permanent magnet to the right of the image. M1 and M2 are initially assembled, and M1 is anchored. M2 disassembles in (a) when the permanent magnet is sufficiently close to the working area.

From the three methods of disassembly, *Method 2 - In-Plane Rotation Disassembly*, is advantageous because only small magnetic fields are required, which can be created by the electromagnetic coils used to locomote the Mag-μMods. *Method 1 - Translation Disassembly* requires a large field gradient, which must currently be

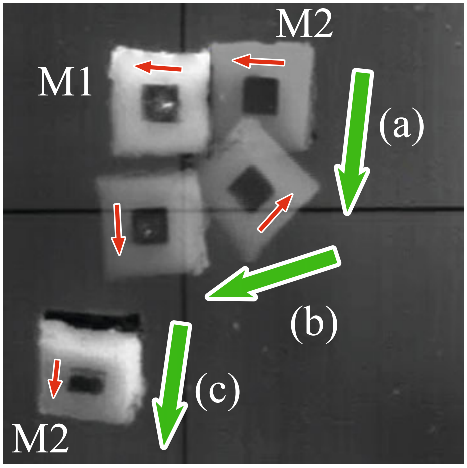


Fig. 12 Superimposed frames from a video of two Mag-μMods, M1 and M2, disassembling by using applied electromagnetic torques from the electromagnetic coils. M1 is anchored to the surface and initially combined with M2. The external fields are applied and cause M2 to orient downwards from (a) to (b) by rotating it clockwise. In this configuration, M2 can walk away from M1 in (c), as the two modules repel each other.

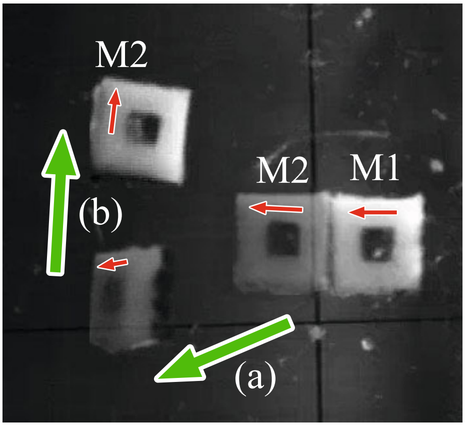


Fig. 13 Superimposed frames from a video of two Mag-μMods, M1 and M2, disassembling by manually bringing permanent magnet below the surface, with magnetization pointing into the page. M1 and M2 are initially assembled, and M1 is anchored. A leftward driving field is initially applied, and M2 disassembles in (a) when the permanent magnet is positioned. M2 then moves upwards in (b).

produced by an external permanent magnet. While *Method 3 - Out-of-Plane Rotation Disassembly* should be possible with the current electromagnetic system, successful disassembly occurred intermittently, implying that the necessary fields can be significantly larger than estimated. For assured success, a permanent magnet was used in Methods 1 and 3, though placed much further away for Method 3.

These permanent magnets can potentially be replaced by high-field electromagnetic coils to support future automation at the cost of increasing the complexity of the electromagnetic coil system. On the other hand, Methods 1 and 3 require fewer magnetic field direction changes, and can potentially be quicker and more reliable than Method 2 for disassembly. Method 3 in particular can be implemented by a single high-field coil placed underneath the working surface, only marginally increasing complexity of the system.

8 Conclusion

In this study, robotic magnetic micro-modules, under 1 mm in size, have been shown to assemble and disassemble by using externally applied electromagnetic fields and electrostatic anchoring techniques. Future work will include extending this concept to larger numbers of modules, demonstrating reconfigurable magnetically-stable configurations, implementing autonomous control to generate these configurations, and further developing the theory to describe forces between modules and assemblies of modules. In addition, a more powerful electromagnet system will be created so that all disassembly methods can be computer controlled.

Acknowledgement

The authors would like to thank Brian Knight for use of the vibrating sample magnetometer, Magnequench, Inc. for graciously supplying us with magnetic powder, and the NanoRobotics Laboratory members for all of their support and suggestions. This work is supported by the National Science Foundation CAREER award program (NSF IIS-0448042), and the NSF Graduate Research Fellowship.

References

1. Yim, M., Shen, W.M., Salemi, B., Rus, D., Moll, M., Lipson, H., Klavins, E., Chirikjian, G.S.: IEEE Robotics and Automation Magazine 14, 43 (2007)
2. Shen, W.M., Chiu, H., Rubenstein, M., Salemi, B.: Proceedings of the Space Technology International Forum, Albuquerque, New Mexico (2008)
3. Goldstein, S.C., Campbell, J.D., Mowry, T.C.: Computer 38, 99 (2005)
4. Yoshida, E., Kokaji, S., Murata, S., Tomita, K., Kurokawa, H.: Journal of Robotics and Mechatronics 13, 212 (2001)
5. Donald, B., Levey, C., Paprotny, I.: Journal of Microelectromechanical Systems 17, 789 (2008)

6. Pawashe, C., Floyd, S., Sitti, M.: International Journal of Robotics Research 28, 1077 (2009)
7. Vollmers, K., Frutiger, D., Kratochvil, B., Nelson, B.: Applied Physics Letters 92 (2008)
8. Sul, O., Falvo, M., Taylor, R., Washburn, S., Superfine, R.: Applied Physics Letters 89 (2006)
9. Ergeneman, O., Dogangil, G., Kummer, M.P., Abbott, J.J., Nazeeruddin, M.K., Nelson, B.J.: IEEE Sensors Journal 8, 20–22 (2008)
10. Martel, S., Felfoul, O., Mathieu, J.B., Chanu, A., Tamaz, S., Mohammadi, M., Mankiewicz, M., Tabatabaei, N.: International Journal of Robotics Research 28, 11–69 (2009)
11. Behkam, B., Sitti, M.: Applied Physics Letters 90 (2007)
12. Pawashe, C., Floyd, S., Sitti, M.: Applied Physics Letters 94 (2009)
13. Floyd, S., Pawashe, C., Sitti, M.: IEEE Transactions on Robotics (2009) (in press)
14. Nanorobotics laboratory, http://nanolab.me.cmu.edu/projects/MagneticMicroRobot/
15. Imbaby, M., Jiang, K., Chang, I.: Journal of Micromechanics and Microengineering 18 (2008)
16. Cheng, D.K.: Field and Wave Electromagnetics, 2nd edn. Addison-Wesley Publishing Company, Inc., Reading (1992)
17. Israelachvili, J.: Intermolecular and Surface Forces. Academic Press, London (1992)

Author Index

Springer Tracts in Advanced Robotics

Edited by B. Siciliano, O. Khatib and F. Groen

Further volumes of this series can be found on our homepage: springer.com

Vol. 69: Rocon E.; and Pons, J.L.
Exoskeletons in Rehabilitation Robotics
138 p. 2010 [978-3-642-17658-6]

Vol. 68: Hsu D.; Isler V.; Latombe J.-C.;
Ming C. Lin (Eds.)
Algorithmic Foundations of Robotics IX
424 p. 2010 [978-3-642-17451-3]

Vol. 67: Schütz, D.; Wahl, F.M. (Eds.)
Robotic Systems for Handling
and Assembly
460 p. 2010 [978-3-642-16784-3]

Vol. 66: Kaneko, M.; Nakamura, Y. (Eds.)
Robotics Research
450 p. 2010 [978-3-642-14742-5]

Vol. 65: Ribas, D.; Ridao, P.; Neira, J.
Underwater SLAM for Structured
Environments Using an
Imaging Sonar
142 p. 2010 [978-3-642-14039-6]

Vol. 64: Vasquez Govea, A.D.
Incremental Learning for Motion Prediction
of Pedestrians and Vehicles
153 p. 2010 [978-3-642-13641-2]

Vol. 63: Vanderborght, B.;
Dynamic Stabilisation of the
Biped Lucy Powered by Actuators
with Controllable Stiffness
281 p. 2010 [978-3-642-13416-6]

Vol. 62: Howard, A.; Iagnemma, K.;
Kelly, A. (Eds.):
Field and Service Robotics
511 p. 2010 [978-3-642-13407-4]

Vol. 61: Mozos, Ó.M.
Semantic Labeling of Places with
Mobile Robots
134 p. 2010 [978-3-642-11209-6]

Vol. 60: Zhu, W.-H.
Virtual Decomposition Control –
Toward Hyper Degrees of
Freedom Robots
443 p. 2010 [978-3-642-10723-8]

Vol. 59: Otake, M.
Electroactive Polymer Gel Robots –
Modelling and Control of
Artificial Muscles
238 p. 2010 [978-3-540-23955-0]

Vol. 58: Kröger, T.
On-Line Trajectory Generation in Robotic
Systems – Basic Concepts for Instantaneous
Reactions to Unforeseen (Sensor) Events
230 p. 2010 [978-3-642-05174-6]

Vol. 57: Chirikjian, G.S.; Choset, H.;
Morales, M., Murphey, T. (Eds.)
Algorithmic Foundations
of Robotics VIII – Selected Contributions
of the Eighth International Workshop on the
Algorithmic Foundations of Robotics
680 p. 2010 [978-3-642-00311-0]

Vol. 56: Buehler, M.; Iagnemma, K.;
Singh S. (Eds.)
The DARPA Urban Challenge – Autonomous
Vehicles in City Traffic
625 p. 2009 [978-3-642-03990-4]

Vol. 55: Stachniss, C.
Robotic Mapping and Exploration
196 p. 2009 [978-3-642-01096-5]

Vol. 54: Khatib, O.; Kumar, V.;
Pappas, G.J. (Eds.)
Experimental Robotics:
The Eleventh International Symposium
579 p. 2009 [978-3-642-00195-6]

Vol. 53: Duindam, V.; Stramigioli, S.
Modeling and Control for Efficient Bipedal
Walking Robots
211 p. 2009 [978-3-540-89917-4]

Vol. 52: Nüchter, A.
3D Robotic Mapping
201 p. 2009 [978-3-540-89883-2]

Vol. 51: Song, D.
Sharing a Vision
186 p. 2009 [978-3-540-88064-6]

Vol. 50: Alterovitz, R.; Goldberg, K.
Motion Planning in Medicine: Optimization
and Simulation Algorithms for
Image-Guided Procedures
153 p. 2008 [978-3-540-69257-7]

Vol. 49: Ott, C.
Cartesian Impedance Control of Redundant
and Flexible-Joint Robots
190 p. 2008 [978-3-540-69253-9]

Vol. 48: Wolter, D.
Spatial Representation and
Reasoning for Robot
Mapping
185 p. 2008 [978-3-540-69011-5]

Vol. 47: Akella, S.; Amato, N.;
Huang, W.; Mishra, B.; (Eds.)
Algorithmic Foundation of Robotics VII
524 p. 2008 [978-3-540-68404-6]

Vol. 46: Bessière, P.; Laugier, C.;
Siegwart R. (Eds.)
Probabilistic Reasoning and Decision
Making in Sensory-Motor Systems
375 p. 2008 [978-3-540-79006-8]

Vol. 45: Bicchi, A.; Buss, M.;
Ernst, M.O.; Peer A. (Eds.)
The Sense of Touch and Its Rendering
281 p. 2008 [978-3-540-79034-1]

Vol. 44: Bruyninckx, H.; Přeučil, L.;
Kulich, M. (Eds.)
European Robotics Symposium 2008
356 p. 2008 [978-3-540-78315-2]

Vol. 43: Lamon, P.
3D-Position Tracking and Control
for All-Terrain Robots
105 p. 2008 [978-3-540-78286-5]

Vol. 42: Laugier, C.; Siegwart, R. (Eds.)
Field and Service Robotics
597 p. 2008 [978-3-540-75403-9]

Vol. 41: Milford, M.J.
Robot Navigation from Nature
194 p. 2008 [978-3-540-77519-5]

Vol. 40: Birglen, L.; Laliberté, T.; Gosselin, C.
Underactuated Robotic Hands
241 p. 2008 [978-3-540-77458-7]

Vol. 39: Khatib, O.; Kumar, V.; Rus, D. (Eds.)
Experimental Robotics
563 p. 2008 [978-3-540-77456-3]

Vol. 38: Jefferies, M.E.; Yeap, W.-K. (Eds.)
Robotics and Cognitive Approaches to
Spatial Mapping
328 p. 2008 [978-3-540-75386-5]

Vol. 37: Ollero, A.; Maza, I. (Eds.)
Multiple Heterogeneous Unmanned Aerial
Vehicles
233 p. 2007 [978-3-540-73957-9]

Vol. 36: Buehler, M.; Iagnemma, K.;
Singh, S. (Eds.)
The 2005 DARPA Grand Challenge – The Great
Robot Race
520 p. 2007 [978-3-540-73428-4]

Vol. 35: Laugier, C.; Chatila, R. (Eds.)
Autonomous Navigation in Dynamic
Environments
169 p. 2007 [978-3-540-73421-5]

Vol. 33: Kong, X.; Gosselin, C.
Type Synthesis of Parallel
Mechanisms
272 p. 2007 [978-3-540-71989-2]

Vol. 34: Wisse, M.; van der Linde, R.Q.
Delft Pneumatic Bipeds
136 p. 2007 [978-3-540-72807-8]

Vol. 30: Brugali, D. (Ed.)
Software Engineering for Experimental Robotics
490 p. 2007 [978-3-540-68949-2]

Vol. 29: Secchi, C.; Stramigioli, S.; Fantuzzi, C.
Control of Interactive Robotic Interfaces – A
Port-Hamiltonian Approach
225 p. 2007 [978-3-540-49712-7]

Vol. 28: Thrun, S.; Brooks, R.;
Durrant-Whyte, H. (Eds.)
Robotics Research – Results of the 12th
International Symposium ISRR
602 p. 2007 [978-3-540-48110-2]

Vol. 27: Montemerlo, M.; Thrun, S.
FastSLAM – A Scalable Method for the
Simultaneous Localization and Mapping
Problem in Robotics
120 p. 2007 [978-3-540-46399-3]