

Sukhan Lee  
Kwang-Joon Yoon  
Jangmyung Lee (Eds.)

# Frontiers of Intelligent Autonomous Systems

**Editor-in-Chief**

Prof. Janusz Kacprzyk  
Systems Research Institute  
Polish Academy of Sciences  
ul. Newelska 6  
01-447 Warsaw  
Poland  
E-mail: kacprzyk@ibspan.waw.pl

Sukhan Lee, Kwang-Joon Yoon,  
and Jangmyung Lee (Eds.)

# Frontiers of Intelligent Autonomous Systems

 Springer

*Editors*

Sukhan Lee  
School of Information and  
Communication Engineering  
Sungkyunkwan University  
Kyunggi-Do  
Korea

Jangmyung Lee  
School of Electrical Engineering  
Pusan National University  
Busan  
Korea

Kwang-Joon Yoon  
Dept of Aerospace Information  
Engineering  
Konkuk University  
Seoul  
Korea

ISSN 1860-949X

e-ISSN 1860-9503

ISBN 978-3-642-35484-7

e-ISBN 978-3-642-35485-4

DOI 10.1007/978-3-642-35485-4

Springer Heidelberg New York Dordrecht London

Library of Congress Control Number: 2012953655

© Springer-Verlag Berlin Heidelberg 2013

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed. Exempted from this legal reservation are brief excerpts in connection with reviews or scholarly analysis or material supplied specifically for the purpose of being entered and executed on a computer system, for exclusive use by the purchaser of the work. Duplication of this publication or parts thereof is permitted only under the provisions of the Copyright Law of the Publisher's location, in its current version, and permission for use must always be obtained from Springer. Permissions for use may be obtained through RightsLink at the Copyright Clearance Center. Violations are liable to prosecution under the respective Copyright Law.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Printed on acid-free paper

Springer is part of Springer Science+Business Media ([www.springer.com](http://www.springer.com))

# Preface

Nowadays, a rapid yet full scale of convergence among computation, control and communication is underway, with humans and things being ever more in integration and interaction toward their maximum synergy. This on-going convergence is to result in various scales of distributed cyber physical systems implemented, driving a fundamental change in the way we live with a new generation of transportation, healthcare, education, manufacturing, business, security, energy/environment management as well as domestic/personal service, to name only several. The essence of technologies pursued by the aforementioned convergence is best represented by the following two keywords: intelligence and autonomy, as they are key enablers for systems, machines, robots and tools to deliver their ultimate services to human under cyber physical integration. This volume entitled, “Frontiers of Intelligent Autonomous Systems”, aims at providing readers with the most recent progress on intelligent autonomous systems, with its particular emphasis on intelligent autonomous ground, aerial and underwater vehicles as well as service robots for home and healthcare under the context of the aforementioned convergence. As an edition of the papers selected from the 12<sup>th</sup> International Conference on Intelligent Autonomous Systems (IAS-12), held in Jeju, Korea, June 26–29, 2012, we chose 35 papers out of the 202 papers presented at IAS-12 and requested them to be undergone thorough revision and extension to be included in this volume. The 35 contributions to this volume were then organized into three chapters: Chapter 1 is dedicated to autonomous navigation and mobile manipulation, Chapter 2 to unmanned aerial and underwater vehicles and Chapter 3 to service robots for home and healthcare. To help readers easily access this volume, each chapter starts with a chapter summary introduced by one of the editors: Chapter 1 by Sukhan Lee, Chapter 2 by Kwang-Joon Yoon and Chapter 3 by Jangmyung Lee. It is our wish as the editors of this volume that readers find this volume stimulating and resourceful. Finally, the editors of this volume would like to thank Springer for undertaking the timely publication of this volume.

Sukhan Lee  
Kwang-Joon Yoon  
Jangmyung Lee

# Contents

## Chapter I: Autonomous Navigation and Mobile Manipulation

<b>Fast 6D Odometry Based on Visual Features and Depth</b> . . . . .	5
<i>Salvador Domínguez, Eduardo Zalama, Jaime Gómez García-Bermejo, Rainer Worst, Sven Behnke</i>	
<b>Kinect Enabled Monte Carlo Localisation for a Robotic Wheelchair</b> . . . . .	17
<i>Theodoros Theodoridis, Huosheng Hu, Klaus McDonald-Maier, Dongbing Gu</i>	
<b>Topological Mapping with Image Sequence Partitioning</b> . . . . .	29
<i>Hemanth Korrapati, Jonathan Courbon, Youcef Mezouar</i>	
<b>Visual Memory Update for Life-Long Mobile Robot Navigation</b> . . . . .	43
<i>Jonathan Courbon, Hemanth Korrapati, Youcef Mezouar</i>	
<b>A Software Architecture for RGB-D People Tracking Based on ROS Framework for a Mobile Robot</b> . . . . .	53
<i>Matteo Munaro, Filippo Basso, Stefano Michieletto, Enrico Pagello, Emanuele Menegatti</i>	
<b>Drivable Road Modeling Based on Multilayered LiDAR and Vision</b> . . . . .	69
<i>Sukhan Lee, Yongjin Park</i>	
<b>Omnidirectional Vision for Indoor Spatial Layout Recovery</b> . . . . .	95
<i>Jason Omedes, G. López-Nicolás, J.J. Guerrero</i>	
<b>Development and Experiences of an Autonomous Vehicle for High-Speed Navigation and Obstacle Avoidance</b> . . . . .	105
<i>Jee-Hwan Ryu, Dmitry Ogay, Sergey Bulavintsev, Hyuk Kim, Jang-Sik Park</i>	
<b>RoboEarth Action Recipe Execution</b> . . . . .	117
<i>Daniel Di Marco, Moritz Tenorth, Kai Häussermann, Oliver Zweigle, Paul Levi</i>	

<b>Exchanging Action-Related Information among Autonomous Robots</b> . . . . .	127
<i>Moritz Tenorth, Michael Beetz</i>	
<b>New Motor Primitives through Graph Search, Interpolation and Generalization</b> . . . . .	137
<i>Miha Deniša, Aleš Ude</i>	
<b>Model Based Visual Servoing Tasks with an Autonomous Humanoid Robot</b> . . . . .	149
<i>Amine Abou Moughlba, Enric Cervera, Philippe Martinet</i>	
<b>Chapter II: Unmanned Aerial and Underwater Vehicles</b>	
<b>MAVwork: A Framework for Unified Interfacing between Micro Aerial Vehicles and Visual Controllers</b> . . . . .	165
<i>Ignacio Mellado-Bataller, Jesús Pestana, Miguel A. Olivares-Mendez, Pascual Campoy, Luis Mejias</i>	
<b>Smart Filter Design for the Localization of Robotic Fish Using MEMS Accelerometer</b> . . . . .	181
<i>Tae Suk Yoo, Sang Cheol Lee, Sung Kyung Hong, Young Sun Ryuh</i>	
<b>View Planning of a Multi-rotor Unmanned Air Vehicle for Tree Modeling Using Silhouette-Based Shape Estimation</b> . . . . .	193
<i>Dae-Yeon Won, Ali Haydar Göktoğan, Salah Sukkarieh, Min-Jea Tahk</i>	
<b>Design and Field Testing of Water Quality Sensor Modules Designed for Round-the-Clock Operations from Buoys and Biomimetic Underwater Robots</b> . . . . .	209
<i>Joongki Park, Juchan Sohn, Sunghoon Kim, Jonghyun Park</i>	
<b>Analysis on the Robotic Fish Propulsion for Various Caudal Fin Shapes</b> . . . . .	217
<i>Dongwon Yun, Jinho Kyung, Chanhum Park</i>	
<b>Development of a 3-DOF Fish Robot ‘ICHTHUS V5’</b> . . . . .	225
<i>Gi-Hun Yang, Hyunjin Lee, YoungSun Ryuh</i>	
<b>Polar Histogram Based Sampling Method for Autonomous Vehicle Motion Planning</b> . . . . .	235
<i>Dmitriy Ogay, Jee-Hwan Ryu, Eun-Gyung Kim</i>	
<b>Gaze Control-Based Navigation Architecture for Humanoid Robots in a Dynamic Environment</b> . . . . .	243
<i>Jeong-Ki Yoo, Jong-Hwan Kim</i>	
<b>Comparison between Photo Interrupter and Giant Magnetoresistive Sensor for Auto Focusing System in the Digital Camera</b> . . . . .	253
<i>Sakura Sikander, Han Nam Lee, Hee-Je Kim</i>	

<b>Natural Terrain Detection and SLAM Using LIDAR for an UGV</b> . . . . .	263
<i>Kuk Cho, SeungHo Baeg, Sangdeok Park</i>	
<b>Indoor Flight Testing and Controller Design of Bioinspired Ornithopter</b> . . .	277
<i>Jun-Seong Lee, Jae-Hung Han</i>	
<b>Effect of Passive Body Deformation of Hawkmoth on Flight Stability</b> . . . . .	287
<i>Ryusuke Noda, Masateru Maeda, Hao Liu</i>	
<b>Chapter III: Service Robots for Home and Healthcare</b>	
<b>Mechanism of a Learning Robot Manipulator for Laparoscopic Surgical Training</b> . . . . .	297
<i>Tao Yang, Jiang Liu, Weimin Huang, Liangjing Yang, Chee Kong Chui, Marcelo H. Ang Jr., Yi Su, Stephen K.Y. Chang</i>	
<b>Learning Probabilistic Decision Making by a Service Robot with Generalization of User Demonstrations and Interactive Refinement</b> . . . . .	309
<i>Sven R. Schmidt-Rohr, Fabian Romahn, Pascal Meissner, Rainer Jäkel, Rüdiger Dillmann</i>	
<b>A Case Study of Safety in the Design of Surgical Robots: The ARAKNES Platform</b> . . . . .	323
<i>L. Alonso Sanchez, M.Q. Le, K. Rabenoroosa, C. Liu, N. Zemiti, P. Poignet, E. Dombre, A. Menciassi, P. Dario</i>	
<b>Design of Master Console Haptic Handle for Robot Assisted Laparoscopy</b> . . . . .	333
<i>Chi Yen Kim, Byeong Ho Kang, Min Cheol Lee, Sung Min Yoon</i>	
<b>Fusion of Inertial Measurements and Vision Feedback for Microsurgery</b> . . .	341
<i>Yan Naing Aye, Su Zhao, Cheng Yap Shee, Wei Tech Ang</i>	
<b>A New Concept for a “Vaginal Hysterectomy” Robot</b> . . . . .	351
<i>Kovit Khampitak, Wathanyu Neadsanga, Sirivit Taechajedcadarung-sri, Thantakorn Pongpimon</i>	
<b>Classification of Modeling for Versatile Simulation Goals in Robotic Surgery</b> . . . . .	357
<i>Stefan Jörg, Rainer Konietzschke, Julian Klodmann</i>	
<b>Role of Holographic Displays and Stereovision Displays in Patient Safety and Robotic Surgery</b> . . . . .	369
<i>Ali Sengül, Attila Barsi, David Ribeiro, Hannes Bleuler</i>	
<b>A Methodological Framework for the Definition of Patient Safety Measures in Robotic Surgery: The Experience of SAFROS Project</b> . . . . .	381
<i>Angelica Morandi, Monica Verga, Elettra Oleari, Lorenza Gasperotti, Paolo Fiorini</i>	



<b>System Concept for Collision-Free Robot Assisted Surgery Using Real-Time Sensing</b> .....	391
<i>Jörg Raczkowski, Philip Nicolai, Björn Hein, Heinz Wörn</i>	
<b>Human-Robot Interaction-Based Intention Sharing of Assistant Robot for Elderly People</b> .....	401
<i>Jeong-Yean Yang, Oh-Hun Kwon, Chan-Soon Lim, Dong-Soo Kwon</i>	
<b>Author Index</b> .....	411

# Chapter I

## Autonomous Navigation and Mobile Manipulation

Introduction by Sukhan Lee

Chapter 1 introduces the recent advancement of autonomous navigation and mobile manipulation for mobile robots and intelligent vehicles. For autonomous navigation, the chapter features the application of an affordable RGB-D camera to 6D visual odometry and vehicle localization, the construction of an appearance based topological map for a fast and accurate localization, the update of a visual memory for life-long memory based visual navigation, a robust multi-people tracking using an RGB-D sensor, the recognition of a drivable road with LiDAR and vision, the recovery of spatial layout of a scene from a collection of lines obtained by an omnidirectional vision, and an autonomous vehicle designed for a high speed on/off-road navigation while avoiding diverse obstacles. For mobile manipulation, the chapter features a task execution engine for the RoboEarth as a methodology for storing and reusing task plans on a global accessible database, a knowledge representation and inference for the web-based exchange of information between robots via the Robot Earth, a methodology for discovering novel movement primitives in a database of example trajectories, and an approach of self-localization and manipulation by a humanoid robot based on a closed-loop integration of real-time visual servo. A more detailed introduction of the above listed features of Chapter 1 is followed:

There are 8 papers selected for inclusion for autonomous navigation: The paper entitled, “Fast 6D Odometry Based on Visual Features and Depth,” by Salvador Dominguez, Eduardo Zalama, Jaime Gomez Garcia-Bermejo, Rainer Worst and Sven Behnke presents an efficient and reliable 6D visual odometry based on an affordable RGB-D camera, MS Kinect Sensor, where a reliable set of, as well as a limited number of, 3D features obtained by combining 2D corner features with their measured depths and filtering out unreliable features are tracked to accurately keep track of the camera pose. While, the paper entitled, “Kinect Enabled Monte Carlo Localization for a Robotic Wheelchair,” by Theodoros Theodoridis, Huosheng Hu, Klaus McDonald-Maier and Dongbing Gu reports the efficiency of an RGB-D camera, MS Kinect Sensor, for carrying out the particle filter based Monte Carlo localization of a robotic wheel chair, where a  $6 \times 1$  region locator descriptor is proposed for the observation associated with the particle filter. The  $6 \times 1$  region locator descriptor is obtained from the depth map of the Kinect sensor first by locally approximating the depth map into a  $6 \times 6$  regional depth matrix, then by constructing a  $6 \times 1$  region locator descriptor taking the minimum from each column of the depth matrix. In the paper entitled, “Topological Mapping with Image Sequence Partitioning,” the authors, Hemanth Korrapati, Jonathan Courbon and Youcef Mezouar, address a fast and accurate robot localization

in a large scale of navigation environment. To this end, the authors propose a framework of building sparse topological maps based on the appearance based partitioning of a sequence of images into a group of nodes in similar appearances, such that it can facilitate a coarse loop closure at the node level and a finer loop closure at the image level. On the other hand, in the paper entitled, “Visual Memory Update For Life-Long Mobile Robot Navigation,” the authors, Jonathan Courbon, Hemanth Korrapati and Youcef Mezouar, deal with the problem of updating a visual memory, constructed for vehicle localization based on memory matching, in order to cope with the constant change in visual appearance in time, where they present a method of updating visual memory during navigation based on the concept of short-term and long-term memories. The paper entitled, “Fast and Robust Multi-People Tracking from RGB-D Data for a Mobile Robot,” by Filippo Basso, Matteo Munaro, Stefano Michieletto, Enrico Pagello and Emanuele Menegatti proposes a fast and robust multi-people tracking algorithm for mobile platforms equipped with an RGB-D camera, MS Kinect Sensor. Their approach features a combination of efficient point cloud depth-based clustering and HOG-like classification not only for initializing person tracking but also for carrying out person classification with the online learning of person ID, allowing multi-people tracking even after a full occlusion. The paper entitled, “Drivable Road Modeling Based on Multilayered LiDAR and Vision,” by Sukhan Lee and Yongjin Park, presents the real-time modeling of a drivable road with a 4-layer LiDAR and a vision camera. A drivable road is modeled in real-time along the vehicle movement as an articulated form of multiple rectangular plates linked serially by rotational joints. By fusing the road surfaces and curbs identified by the 4-layer LiDAR with the lanes and crossings recognized by the vision camera, the poses associated with individual plates, as well as the obstacles located on the drivable road, if any, are identified. The paper entitled, “Omnidirectional Vision for Indoor Spatial Layout Recovery,” by J. Omedes, G. Lopez-Nicolas and J. J. Guerrero deals with the problem of recovering the spatial layout of a scene from a collection of lines extracted from a single indoor image obtained by an omnidirectional vision. Based on the vanishing points and lines extracted from omnidirectional images, the authors present a novel approach for spatial layout recovery by taking a set of geometrical constraints into consideration, allowing to detect an arbitrary number of, possibly occluded, floor-wall boundaries. Finally, the paper entitled, “Development of an Autonomous Vehicle for High-Speed Navigation and Obstacle Avoidance,” by Jee-Hwan Ryu, Dmitriy Ogay, Sergey Bulavintsev, Hyuk Kim, and Jang-Sik Park introduces the autonomous vehicle, called Pharos, developed with the capability of up to 60 Km/h of high-speed on/off-road driving, while avoiding diverse patterns of obstacles.

For mobile manipulation, 4 papers are selected for inclusion: The paper entitled, “RoboEarth Action Recipe Execution,” by Daniel Di Marco, Moritz Tenorth, Kai Haussermann, Oliver Zweigle and Paul Levi presents a method of storing and reusing task plans on a global accessible database by robots, allowing them to be able to self-learn sophisticated services, independent of their platforms and application scenarios. The paper describes a task execution engine for RoboEarth and demonstrates its ability to execute tasks in a flexible and reliable way. The paper entitled, “Exchanging Action-Related Information among Autonomous Robots,” by Moritz Tenorth and

Michael Beetz introduces the knowledge representation and inference techniques that are used in the RoboEarth system for the web-based exchange of information between robots. In particular, the paper presents a novel method of representing an environment map by combining an expressive semantic environment model with the selection of a suitable map from the web-based RoboEarth knowledge base. The integrated experiments demonstrate the mobile manipulation task of a robot with the retrieval of a suitable environment map and the estimation and exchange of the information on object properties. The paper entitled, "Discovering New Motor Primitives in Transition Graphs," by Miha Denisa and Ales Ude presents a methodology for discovering novel movement primitives in a database of example trajectories. The proposed method first constructs transition graphs at different levels of granularity from the example trajectories and then discovers novel movements by exploiting the interdependencies between the movements encoded by the transition graph, where the example trajectories are acquired either from human demonstrations or by kinesthetic guiding of manipulators. The paper entitled, "Real-Time Model Based Visual Servoing Tasks on a Humanoid Robot," by Amine Abou Moughlbay, Enric Cervera and Philippe Martinet presents an approach of self-localization and manipulation by a humanoid robot, Nao, based on a closed-loop integration of real-time visual servo. The paper demonstrates the effectiveness by showing the self-localization while walking, the head servo for visibility, as well as the detection, tracking and manipulation of objects distributed in the environment.

# Fast 6D Odometry Based on Visual Features and Depth

Salvador Domínguez<sup>1</sup>, Eduardo Zalama<sup>2</sup>, Jaime Gómez García-Bermejo<sup>2</sup>,  
Rainer Worst<sup>3</sup>, and Sven Behnke<sup>3,4</sup>

<sup>1</sup> Cartif Foundation, Valladolid, Spain

<sup>2</sup> ITAP, University of Valladolid, Spain

<sup>3</sup> Fraunhofer IAIS, Sankt Augustin, Germany

<sup>4</sup> University of Bonn, Germany

**Abstract.** The availability of affordable RGB-D cameras which provide color and depth data at high data rates, such as Microsoft MS Kinect, poses a challenge to the limited resources of the computers onboard autonomous robots. Estimating the sensor trajectory, for example, is a key ingredient for robot localization and SLAM (Simultaneous Localization And Mapping), but current computers can hardly handle the stream of measurements. In this paper, we propose an efficient and reliable method to estimate the 6D movement of an RGB-D camera (3 linear translations and 3 rotation angles) of a moving RGB-D camera. Our approach is based on visual features that are mapped to the three Cartesian coordinates (3D) using measured depth. The features of consecutive frames are associated in 3D and the sensor pose increments are obtained by solving the resulting linear least square minimization system. The main contribution of our approach is the definition of a filter setup that produces the most reliable features that allows for keeping track of the sensor pose with a limited number of feature points. We systematically evaluate our approach using ground truth from an external measurement systems.

## 1 Introduction

Optimizing the use of the available resources in mobile robotics like energy or processing capabilities is important, because some robots do not have any external physical connection and have to work autonomously. One of the most important tasks a mobile robot must perform is self-localization with respect to a map. This task is, however, also one of the most resource-consuming when no global positioning system is used. When the robot pose is known, it can be used for navigation as well as other purposes such as mapping the environment. The measurements needed for localization are usually provided by expensive and heavy laser range sensors. The availability of affordable lightweight RGB-D cameras offers the possibility to acquire color images and depth maps at high frame rates which must be processed in real time in order to be used for robot control.

In this paper, we propose a reliable method to obtain 6D odometry from color and depth images. Our method can be applied to estimate the sensor pose in real time, requiring limited computational resources.

We use a Kinect RGBD sensor as the only capturing device. The sensor driver provides color and depth images in a standard format. At each frame, we extract a set of visual features from the RGB image. They are filtered and mapped to 3D space using the associated depth image. We associate the features of consecutive frames in 3D and finally obtain the sensor pose increment. Key to our method is a carefully designed filtering process, which takes the most reliable features to estimate odometry. Finally, we calculate the transformation that minimizes the mean square error between two sets of associated features. Because our method processes only a limited number of features, it can be easily applied for pose estimation on mobile robots with limited computational resources, like e.g. a 1.6 GHz CPU with 1 GB RAM that we used for our experiments. Even with such a small computer, we achieve odometry rates near 20 Hz in low resolution, which makes it possible to use our method on a robot that moves at full speed (1 m/sec linear velocity and /or 45°/sec angular velocity in our case). Our method does not rely on any assumption about the kinematics of the movement. The odometry obtained can be used later as an input data stream for 6D-SLAM.

Raw detected features are intrinsically noisy and would produce low-quality odometry estimates if they were applied directly. The filtering reduces the number of features and consequently the time consumption of the overall process significantly. Instead of trying to match the features extracted from consecutive frames, we first stabilize the consistency of the current feature set with respect to a reference feature set. As explained below, such stabilization is performed by dynamically adapting the current set of features to a reference set.

The remainder of the paper is structured as follows. In Section II, we discuss related work. In Section III, we present our approach to the estimation of 6D odometry based on visual features. Finally, we demonstrate the advantages of our method using systematic experiments in Section IV.

## 2 Related Work

Visual odometry is the process of calculating the increments of pose based on the analysis of images captured by a camera. One of the first works in visual odometry is attributed to Moravec [1]. In 1976 he managed to implement a visual odometry system on the Stanford Cart to perform corrections on the control of the vehicle autonomously. In that technical report, the basic concepts of tracking visual features and depth estimation were introduced.

We can identify two main types of visual odometry using cameras as the only capturing device: monocular and stereo. In monocular odometry depth information must be estimated from motion parallax as the camera moves through the environment. Interesting examples of monocular odometry implementation include Nister et al. [2] and Davison [3]. In stereo odometry, the camera does not have to move to calculate depth information. Some implementations of stereo

odometry are described by Moravec [1], Matties [4] and Nister et al. [2]. On the other hand, it is also possible to use additional devices for the acquisition of the depth like laser range scanners, time-of-flight cameras or infrared pattern projectors. The resulting input data stream for visual odometry estimation is in most cases a sequence of images with depth information

The tracking of the features plays an important role in visual odometry, because it is a process that consumes less time than extracting the features in every frame, and can be applied to several consecutive frames while as long as enough features are within the image boundaries. Barron [5] compiled a survey of optical flow algorithms for tracking, like the gradient-based technique proposed by Lucas and Kanade [6] (LK). LK is implemented in OpenCV and used in our approach. The selection of good features to track is also important in order to reduce resource consumption and increase accuracy. An interesting attention-driven feature selection is described by Einhorn [7].

### 3 Method

Figure 1 summarizes our approach for the estimation of the odometry. No prior knowledge about the environment is assumed. We can distinguish two main loops, the extraction loop and the tracking loop. The extraction loop is only executed when a new set of features is needed (generated features GF), i.e. when there are not enough features to obtain a good odometry estimate. The tracking loop is executed in every frame and performs, among other things, tracking, filtering, and matching of the features to get the transformation. The key steps that make this approach suitable for real time application and less sensitive to noise are the filtering of the features and the consistency adaptation (marked with stars in Figure 1) which will be explained in detail below.

#### 3.1 Visual Features Extraction

Visual features are interesting parts of the image with properties that make them suitable for detection. Different steps to detect robust, distinctive invariant features are explained in Lowe [8]. There are different kinds of such features, e.g. corners and blobs. Because their extraction is computationally less demanding, we focus in this paper on the so called corners. Although corner features are not completely robust and stable, they can be stabilized by the 3D information, which in our case is available.

Corners are points which have two dominant directions in the luminosity gradient, in a local neighborhood. Some of the most important techniques for extracting corners are presented by Harris and Stephens [9] as well as Shi and Tomasi [10].

An evaluation of three features detectors (Harris, SIFT (Scale Invariant Feature Transform) and LKT (Lucas-Kanade-Tomasi)) is described by Klippenstein and Zhang [11]. They found LKT to have a better performance in general. We test our approach with two feature extractors: Harris corner detector and LKT

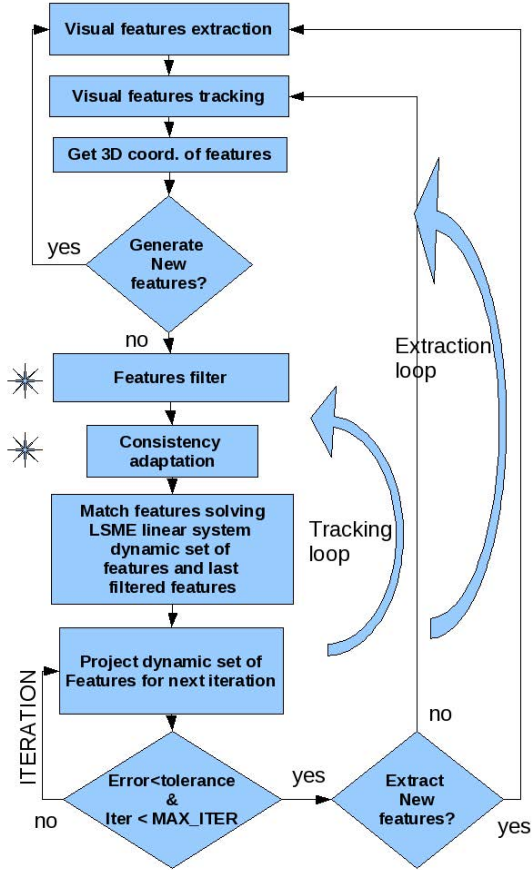


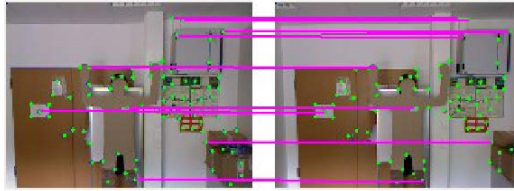
Fig. 1. Flow diagram of the overall process of odometry estimation

corner detector which are both included in the OpenCV library getting lightly better results with LKT.

### 3.2 Tracking of Visual Features

The tracking of features is a crucial step of visual odometry. Shi and Tomasi [10] propose a matching algorithm for tracking visual features based on a model of affine image changes. This technique has been adopted in our approach. For the tracking process, a pair of images is used: the most recently captured image and the previous one (Figure 2). The visual features corresponding to the previous image can be obtained either by feature extraction or as a result of the previous tracking iteration. We apply Shi-Tomasi's method to track the features in the first image, so we know where to expect them in the second image. If we cannot





**Fig. 2.** Features in one frame (previous) are tracked during the tracking process to the next frame (tracked features  $TF$ ). Some of the correspondences are shown in magenta.

establish correspondence for a feature in the second image, it is marked as “non-paired” feature.

### 3.3 Generation of New Features

Because we are looking for a disambiguous solution to the linear error minimization system (see paragraph III-E), a minimum of 4 features are needed so the number of equations is larger or equal to the number of unknowns. When no such solution can be found we seek to generate a new set. There are several reasons for this to happen:

- the scene has no interest points to be easily distinguished like flat colored surfaces with few corners.
- when moving the sensor, the features will likely change their positions in the image and some of them can disappear on the image borders.
- the distribution of the features can lead to clustering in some parts of the image, for example when the camera moves backwards and distance between tracked features reduces and some of the features fuse together.

Moreover, the grouping of features decreases the precision of the movement estimate. For better performance, the visual features should be distributed roughly uniformly in the image. In order to deal with this problem, we measure the distance traveled and the rotated angles of the sensor from the last feature extraction. When either the distance or the angle, exceeds a given threshold, we trigger the generation of a new set of features.

### 3.4 Projection of Visual Features in 3D

RGB-D sensors like MS Kinect provide the depth information at most pixels in the color image. However, sometimes the physical properties of the surface at some positions are not suitable for the measurement of depth. Many of the visual features lie on object boundaries where depth information is often missing. Moreover, visual features might jump from the front edge of an object to the background side.

If the entire neighborhood of the feature has similar depth values, then the depth of the feature point can be obtained by averaging these values. It could

happen that the feature does not have depth information or that there is a large difference among the neighbors depths, probably because the feature is lying at the jump edge caused by a border. In this case, we utilize border ownership, which means that we average the depth of the closest neighboring points.

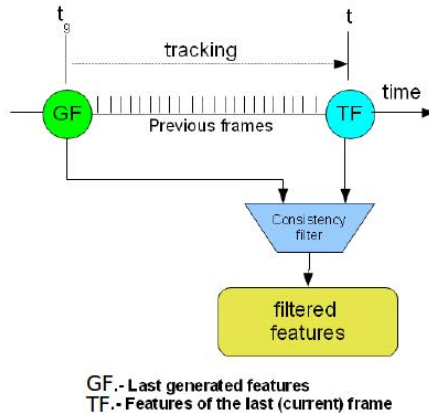
Averaging has also the effect of reducing the noise of the depth measurements. It can happen that no points in the neighborhood have depth information and it will not be possible to calculate the 3D depth of the feature. Those features will be discarded in the filtering process.

Preferring closer points is also advantageous for the measurement precision of triangulation sensors like Kinect because the error increase quadratically with the distance. The optimal radius of the neighborhood depends on the image resolution. In our experiments with a 640x480 resolution, a radius of 3 pixels has been proven to work suitably. Because our neighborhood is defined as a square, this implies a neighborhood of 49 pixels.

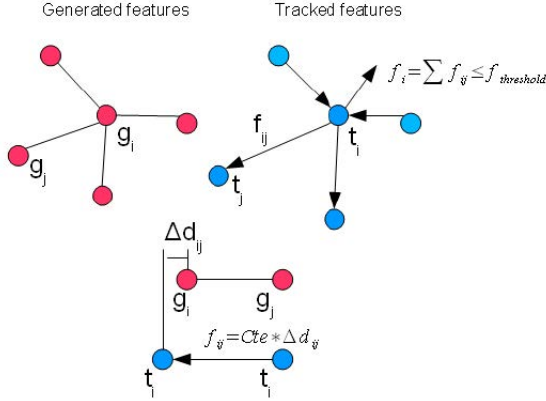
### 3.5 Filtering and Consistency Adaptation

In order to retain only the most reliable features for pose estimation, we designed a set of rules for discarding features. Firstly, we filter out the features with missing depth, then the consistency of the distribution is checked. We analyze the relative 3D positions of the last tracked features  $TF$  compared to the respective positions of the generated features  $GF$ , as is shown in Figure 3.

This filter is based on the idea that every feature has to correspond to a fixed point in the real world. This way, the position of the  $TF$  in global coordinates must not differ too much from the global position of  $GF$  during the tracking process. In our approach, we model the set of features as a cloud of 3D points where each point is connected to the rest by elastic links.



**Fig. 3.** Filtering process. The last extracted features ( $GF$ ) and the last tracked features ( $TF$ ) are used to filter the features.

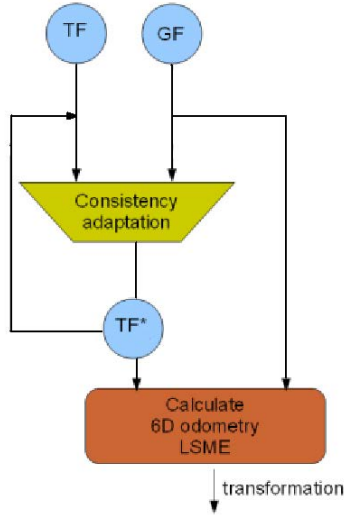


**Fig. 4.** The force in each feature is determined as a vectorial sum of the deformation forces in each link. The force in every link (i,j) is proportional to the difference of distance between that link in TF and GF.

To move a point from its equilibrium point in such elastic net, it is necessary to apply a force (see Figure 4). If we consider the *GF* as the balanced state where the deformation energy is zero, then the *TF* has an energy accumulated because the relative position of the features is not exactly the same due to non-perfect tracking. The force required at every node to move it from its equilibrium point to its current position, can be calculated through a vectorial sum of the forces in every link connecting that node to the rest. Every link can only apply a force in its longitudinal direction and can be calculated because we know the 3D coordinates of every feature in both sets of features. The condition to pass the filter is that the displacement force must stay below a threshold value.

The consistency filter discards features that moved more than a threshold distance from their equilibrium position. However, probably most of the rest of the features are still noisy in depth because the sensor accuracy decreases on the square of the distance to the point in the real world. Noisy depth values lead to decreased odometry accuracy, especially at distances above 2 meters using the MS Kinect sensor. To solve this problem we apply a filter to adapt and stabilize the consistency of the *TF*. This is what we call consistency adaptation in Figure 11. This filter works on so-called adaptive features *TF\**. This new set of features is the result of the gradual consistency adaptation from set *TF* to *GF*. It can be seen as a stable version of the *TF* that we use for calculating the odometry transformation.

Figure 5 represents the process of consistency adaptation of the features. *TF\** have the properties of having consistency similar to *GF* and a global pose similar to *TF*. Little changes in the relative average positions of the tracked features in 3D will be adapted progressively to the equilibrium position. *TF\** and *GF* are finally used to calculate the transformation of the odometry leading to stable odometry.



**Fig. 5.** Calculation and use of the adaptive features

The adaptation of the consistency has positive effects over the noise in the estimated position. Figure 6 shows the difference without and with adaptation, respectively. The ground truth in this experiment was composed by perpendicular straight lines following X, Y and Z axes. When no adaptation is applied, the noise increases with the distance to the features, on the contrary with adaptation the noise is less dependent on that distance.

### 3.6 Estimation of Pose Increments

Let us consider that we have tracked a set of  $N$  visual features  $p$  from the previous frame to the last one producing the set  $q$  whose features are paired one by one



**Fig. 6.** Effect in the adaptation of the consistency. The distance from the sensor to the features increases in the direction of the arrow. Notice that the dispersion of the position increases with the distance to the sensor when no adaptive consistency is used

with those in  $p$ . These features have passed the filtering process and consistency adaptation so they are stable.

We are looking for a rigid transformation  $(\mathfrak{R}, t)$  that minimizes the value of the following expression

$$J(\mathfrak{R}, t) = \sum_N \|p_i - \mathfrak{R}q_i - t\|^2 \quad (1)$$

where  $p_i$  and  $q_i$  represent the 3D position of the paired feature  $i$  in the previous frame and the current frame, respectively.

This is a typical problem of Linear Least Square Minimization. The linear system is represented by  $3 \times N$  equations ( $N$  features  $\times$  3 coordinates) with 12 unknowns (9 for the rotation matrix  $\mathfrak{R}$  and 3 for the translation vector  $t$ ). Therefore, to have a unique solution we need at least 4 features.

Equation (1) can be represented in a different way, being equivalent to minimize the following expression

$$\begin{aligned} \sum_N \|p_i - A(q_i) * \varphi\|^2 &\rightarrow p_i \approx A(q_i) * \varphi \\ A(q_i)^T * p_i &= A(q_i)^T * A(q_i) * \varphi \\ \varphi &= (A(q_i)^T * A(q_i))^{-1} * A(q_i)^T * p_i = R(q_i) * p_i \end{aligned} \quad (2)$$

In this expression,  $\varphi$  is the  $12 \times 1$  vector of unknowns which is composed by elements of the rigid transformation  $(\mathfrak{R}, t)$  and  $R(q_i)$  is a  $12 \times 3N$  matrix which depends on the coordinates of the features in the last frame  $q_i$ , and can be determined by classical methods such as QR or Cholesky factorization [12].

The error of the transformation is checked by projecting every feature in the last frame to the previous frame using the transformation  $(\mathfrak{R}, t)$ . If the error for the feature  $i$   $\epsilon_i$  is less or equal to a threshold  $\delta$  the feature will be used for the next iteration (3).

$$\epsilon_i = \|p_i - \mathfrak{R}q_i - t\| \leq \delta \quad (3)$$

The process of pose estimation stops when

$$\sum_N \|p_i - \mathfrak{R}q_i - t\| \leq \text{threshold} \quad (4)$$

## 4 Experiments and Results

To evaluate the quality of our method, we have used datasets with a known ground truth. The datasets were recorded at Gelsenkirchen University of Applied Sciences using an industrial Kuka KR60HA robot (Figure 7). One example of the scene seen by the sensor is shown in Figure 8.

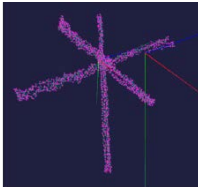
The sensor trajectories used were geometric paths such as lines and arcs (see Figure 9) traced at about 15 cm/sec. The number of visual features always was bigger than 10 and the distance to the features covered a range between 2 and 5 m. Every test starts and finishes in the same position so that the difference



**Fig. 7.** Robot used for ground truth experiments



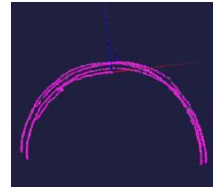
**Fig. 8.** Scene seen by the camera for experiments 1 and 2



(a) Experiment 1



(b) Experiment 2



(c) Experiment 3

**Fig. 9.** Trajectories obtained on the experiments

between the initial position and the final one will measure the total error. We also pay attention to the maximum distance of the estimated position to the theoretical trajectory.

*Experiment 1:* Linear translation along X, Y, and Z directions. The traveled distance along each direction was 1 m, and was repeated three times (see Figure 9a). This way, the total traveled length was 18 m. The final error found in the position was 11 cm in translation and  $1.5^\circ$  in inclination, which correspond to a 0.6% accumulated position error. The maximum deviation from the theoretical trajectory was only 6 cm.

*Experiment 2:* Circles traced on the XY, XZ, and YZ planes (see Figure 9b). Each circle was 50 cm in diameter. Every circle was repeated three times. The total length of the trajectory was 14.13 m and the final error was 4.5 cm and  $1.2^\circ$  in inclination, that is less than 0.3% of the total length of the trajectory. The maximum deviation from the theoretical curve was 7 cm.

*Experiment 3:* Translation and rotation along an arc of  $180^\circ$  and 1 m radius, with the camera pointing along the radial direction (see Figure 9c). Each movement was repeated three times, so the total traveled distance was 18.85 m and the total angle rotated was  $1080^\circ$ . The final pose of the camera had an error of 36 cm in position and  $10^\circ$  in orientation, which corresponds to a 1.9% linear error and 0.9% rotation error.

## 5 Conclusion

We have presented a method for estimating the trajectory of a moving RGB-D sensor using 6D odometry based on visual features obtained from the RGB images, which are mapped to 3D using their measured depth. Key to the success of our method is a filtering process that removes the features that are not reliable, thus making our solution robust. The consistency adaptation of the features improves the stability and precision of the result. Our algorithm is computationally very efficient, because only a limited number of points is processed every frame.

We evaluated our method by moving the camera with an industrial arm, which provides ground truth for the trajectories. The obtained results are satisfactory for visual odometry considering the limitations of the sensor with regards to depth noise.

Our 6D visual odometry is in use as one of the key parts of the 3D SLAM system for the ground robot used in NIFTi project which is equipped with a small computer with a processor of 1.6 GHz. Working at  $640 \times 480$  and 15-20 fps the CPU usage stays near 35% on that computer.

**Acknowledgments.** We would like to thank to both Cartif Foundation and Fraunhofer IAIS for having made this work possible. This work has been funded by the European Commission under contract number EU FP7 NIFTi / ICT-247870 and also supported by Junta de Castilla y León (CCTT/10/VA/0001) and by the Science and Innovation Spanish ministry (Pr.Nb.DPI2008-06738-C02-01/DPI). Thanks also to the NIFTi team of IAIS, especially to Thorsten Linder, Viatcheslav Tretyakov and Nenad Biresev for their support in the experiments and helpful suggestions.

## References

1. Moravec, H.: Obstacle avoidance and navigation in the real world by a seeing robot rover. Tech. report and doctoral dissertation, Robotics Institute, Carnegie Mellon University, Stanford University, CMU-RI-TR-80-03 (1980)
2. Nister, D., Naroditsky, O., Bergen, J.: Visual odometry for ground vehicle applications. *Journal of Field Robotics* 23(1) (2006)
3. Davison, A.J.: Real-time simultaneous localization and mapping with a single camera. In: *Proceedings of the International Conference on Computer Vision*, Nice (2003)
4. Matties, L.: Dynamic Stereo Vision. PhD thesis, Dept. of Computer Science, Carnegie Mellon University, CMU-CS-89-195 (1989)
5. Barron, J.L., Fleet, D.J., Beauchemin, S.S.: Performance of Optical Flow Techniques. *The International Journal of Computer Vision* 12(1), 43–77 (1994)
6. Lucas, B.D., Kanade, T.: An Iterative Image Registration Technique with an Application to Stereo Vision. In: *Proceedings of 7th International Joint Conference on Artificial Intelligence*, pp. 674–679 (1981)
7. Einhorn, E., Schrter, C., Gross-M, H.-M.: Can't Take my Eye on You: Attention-Driven Monocular Obstacle Detection and 3D Mapping. In: *IROS 2010* (2010)

8. Lowe, D.G.: Distinctive image features from scale invariant keypoints. *International Journal of Computer Vision* 60(2), 91110 (2004)
9. Harris, C., Stephens, M.: A Combined Corner and Edge Detector. In: *Proc. of The Fourth Alvey Vision Conference, Manchester*, pp. 147–151 (1988)
10. Shi, J., Tomasi, C.: Good features to track. In: *IEEE Conference on Computer Vision and Pattern Recognition, Seattle* (1993)
11. Klippenstein, J., Zhang, H.: Quantitative Evaluation of Feature Extractors for Visual SLAM. J Department of Computing Science University of Alberta Edmonton, AB, Canada T6G 2E8 (2007)
12. Golub, G.H., Van Loan, C.F.: *Matrix Computations*, 3rd edn. Johns Hopkins (1996) ISBN 978-0-8018-5414-9



# Kinect Enabled Monte Carlo Localisation for a Robotic Wheelchair

Theodoros Theodoridis, Huosheng Hu, Klaus McDonald-Maier, and Dongbing Gu

School of Computer Science and Electronic Engineering  
University of Essex, Wivenhoe Park, Colchester, CO4 3SQ, UK  
{ttheod, hhu, kdm, dgu}@essex.ac.uk

**Abstract.** Proximity sensors and 2D vision methods have shown to work robustly in particle filter-based Monte Carlo Localisation (MCL). It would be interesting however to examine whether modern 3D vision sensors would be equally efficient for localising a robotic wheelchair with MCL. In this work, we introduce a visual Region Locator Descriptor, acquired from a 3D map using the Kinect sensor to conduct localisation. The descriptor segments the Kinect's depth map into a grid of 36 regions, where the depth of each column-cell is being used as a distance range for the measurement model of a particle filter. The experimental work concentrated on a comparison of three different localization cases. (a) an odometry model without MCL, (b) with MCL and sonar sensors only, (c) with MCL and the Kinect sensor only. The comparative study demonstrated the efficiency of a modern 3D depth sensor, such as the Kinect, which can be used reliably for wheelchair localisation.

**Keywords:** Monte Carlo Localisation, Particle Filter Localisation, Region Locator Descriptors, Kinect sensor.

## 1 Introduction

The need for reliable wheelchair localisation has become a high priority need for the elderly and disabled. To provide adequate autonomous mobility, robotic wheelchairs can be used for transportation. This would require an efficient localisation method, and a depth sensor for coverage. In our effort to tackle the localisation problem with a robotic wheelchair, the Monte Carlo Localisation (MCL) method was used. The perception was undertaken by the Microsoft's Kinect depth sensor, for the coverage of a 640×480 depth pixels in ~60° wide lens angle. The sensor's resolution along with its low cost, compared to 1D laser range finders, makes it an attractive apparatus for localisation.

MCL is one of the most popular probabilistic methods for online robot localization, which is based on a particle filter algorithm. MCL was primarily introduced by [2], on their work in mobile robot localisation. MCL in its purest form tackles the global position estimation, and the local position tracking problem. To perform localisation, the robot's state space is represented by a sample (particle) density, which approximates the robot's position. The whole process is conducted in two phases; the

*prediction* and *update*. In the prediction phase a motion model predicts the robot's current pose with a Probability Density Function (PDF). Similarly, in the update phase a measurement (sensor) model is incorporated to evaluate each particle in the density, and sample with replacement those particles evaluated as good predictors. In [3], an alternative MCL method is proposed based on an adaptive particle density, which showed improved accuracy and less computation effort. A stereo vision based MCL was presented by [4], using a scale invariant feature transform descriptor, given a 3D map. The measurement model incorporated 3D features extracted from landmarks. The proposed method appeared to solve the 6 DOF motion, and perform accurate localisation with a motion model having sensor measurements only. Another MCL method that exploits visual features was introduced by [5], for the localisation of the Aibo dog-robots. Two MCL variants were proposed; a landmark, and a field line-based extraction. The MCL using the visual features performed fast and reactive localisation within a miniaturised football pitch. In [7] a landmark tree model is employed for self-localisation of an autonomous wheelchair. The method utilises an image retrieval technique and the Bayes rule to localise the wheelchair. In addition, a path planning algorithm is introduced, which exploits a treelike structure to locate landmarks and destination locations. Landmarks are recognised through an image by extracting the shape and structure, and localisation is conducted by traversing within the tree nodes to elicit an optimal path. A probabilistic odometry (motion) model was introduced by [8] for an autonomous wheelchair. The method constructs a set of frequency tables of the wheelchair's pose stored in bins. A particle filter advises these tables to make predictions for localisation.

The rest of the paper is organized as follows. Section 2 describes the classical Monte Carlo Localisation method, given a brief algorithm. Section 3 presents the Kinect sensor and a depth-based region locator descriptor used in MCL. The probabilistic models that implement the MCL's structure are given in Section 4. Section 5 demonstrates a tri-case experiment using a mobile robot, and Section 6 points out conclusions and future directions.

## 2 Monte Carlo Localisation

The general idea of MCL is the deployment of a particle density, designated to predict a robot's position. The particles can be seen as virtual copies of the robot's existence, incorporating 2D kinematics for locomotion (motion model), and a beam array for perception (measurement/sensor model). As stated, MCL is conducted in two main phases to perform localisation. In the *prediction* phase at every time step the particles' kinematics is updated when the robot transits from state  $X_{k-1}$  to  $X_k$ . Each particle is assigned with a weighting fitness value that assesses how well the actual position of the robot is approximated. This accounts the actual (the robot's) and the expected (the particle's) sensory perception. In the *update* phase a subset of particles is drawn probabilistically with replacement to represent the new particle population. The resampling is based on the Darwinian principle of natural selection. Hence, high fitness particles have higher probability to be selected multiple times. Progressively, the whole particle population converges near the actual position of the robot.

**Algorithm 1** Monte Carlo Localisation1: Generate Particles  $\mathbf{X}_x \sim N(\boldsymbol{\mu}_x, \boldsymbol{\sigma}_x^2)$ 2: Initialise Particle Weights  $w_k = 1/Q$ 3: Prediction  $\mathbf{x}_k \sim p(\mathbf{x}_k | \mathbf{x}_{k-1})$ 

4: Weight Estimation:

$$w_k = p(z_k | \mathbf{x}_k, u) w_{k-1}, \quad w_k = w_k / \sum_{q=1}^Q w_k^{[q]}$$

5: Update  $\mathbf{X}_k \sim \left\langle \mathbf{x}_k^{[q]}, w_k^{[q]} \right\rangle_{q=1}^Q$ 

Algorithm 1 describes briefly the MCL method in five steps. In the first, a particle distribution  $\mathbf{X}$  is created with randomly chosen state vectors  $\mathbf{x} \bullet \langle \langle x, y, \theta \rangle, w \rangle$  from a Gaussian distribution  $N$ . The randomly chosen parameters are the robot's pose at coordinates  $x$ ,  $y$ ,  $\theta$ , and the weight factor  $w$ . Step 2 initialises the particles weights with the factor  $1/Q$ , where  $Q$  is the total number of particles. In step 3, the prediction takes place of the state  $\mathbf{x}_k$  given the initial state  $\mathbf{x}_{k-1}$ . At this step, the robot's kinematics is applied at each particle. The weight estimation in step 4 is done by the product of the measurement probability shown as  $z_k$  given the state  $\mathbf{x}_k$  after applying a control command  $u$ , for particle  $q$ . Normalisation follows to scale the particle weights in the interval  $\{0, 1\}$ , so all add to 1. The update phase in step 5 samples with replacement a subset of particles based on their weight values  $w_k$ . The process is repeated recursively by looping back to step 2.

### 3 Kinect Depth-Based Perception

#### 3.1 Kinect Sensor

The main sensor apparatus utilised in this work is the Kinect sensor (see Fig. 1), produced by Microsoft. Kinect is a vision sensor providing a RGB and a depth image in a single device. It is being used by Microsoft's game industry for games that do not require physical control devices such as joysticks. Basically, the user becomes the controller by using gestures and body movements. For using the sensor without the X-Box, as it is the primary device to work with, several drivers have been released to access the device from a PC. For the current application we have used the CL-NUI-Platform driver for Windows 7 (version 1.0.0.1121).



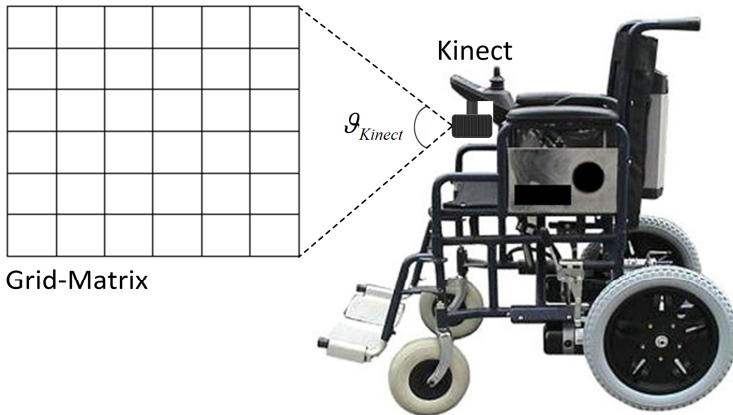
**Fig. 1.** Microsoft's Kinect sensor

The Kinect's video frame rate is up to 30 Hz, with an 8-bit VGA resolution of  $640 \times 480$  pixels. The device provides 11-bit depth with 2,048 levels of intensity. The sensor's approximate ranging limit is between 0.5 to 6 m distance, while the angular field of view is  $57^\circ$  horizontally, and  $43^\circ$  vertically. There is also an embedded quadruple microphone array with a 16-bit audio and sampling rate of 16 kHz [6].

### 3.2 Region Locator Descriptor

The sensor coverage of a significant angular range is a crucial factor for autonomous and safe transportation. Kinect's specifications allow us to cover a large field of view from which information can be acquired. Such information regards the extraction of depth in local regions in this field. The local depth regions can play a twofold role; initially to harness them for the MCL's sensor model to conduct localisation. Secondly, for obstacle avoidance and safe navigation.

A local depth region acquisition method is being proposed to work for localisation purposes. For this reason, we generate a  $6 \times 6$  depth matrix, which discretises the image in 36 depth regions as depicted in Fig. 2. The region matrix is given by matrix **A** in Eq. 1, with entry elements the regions represented by the matrix **B**. The elements of matrix **B** are the individual depth pixels indexing  $107 \times 80$ , as divided by 6 from the overall image resolution. For each region **B** the average depth is estimated, therefore matrix **A** now consists of the mean values for each region. The final region locator descriptor (*rl<sub>d</sub>*) is a vector with 6 elements reflecting to the minimum distance from each column  $n$  (0, ..., 5) of matrix **A**.



**Fig. 2.** The Kinect sensor attached on a robotic wheelchair. The grid-matrix is extracted by the visual descriptor from the Kinect's depth image.

$$rld = \begin{bmatrix} \underset{m}{\operatorname{argmin}} A_{m,0} \\ \underset{m}{\operatorname{argmin}} A_{m,1} \\ \underset{m}{\operatorname{argmin}} A_{m,2} \\ \underset{m}{\operatorname{argmin}} A_{m,3} \\ \underset{m}{\operatorname{argmin}} A_{m,4} \\ \underset{m}{\operatorname{argmin}} A_{m,5} \end{bmatrix} \leftarrow A = \begin{bmatrix} \bar{B}_{0,0} & \bar{B}_{0,1} & \cdots & \bar{B}_{0,5} \\ \bar{B}_{1,0} & \bar{B}_{1,1} & \cdots & \bar{B}_{1,5} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{B}_{5,0} & \bar{B}_{5,1} & \cdots & \bar{B}_{5,5} \end{bmatrix}, \bar{B} = \begin{bmatrix} p_{0,0} & p_{0,1} & \cdots & p_{0,80} \\ p_{1,0} & p_{1,1} & \cdots & p_{1,80} \\ \vdots & \vdots & \ddots & \vdots \\ p_{107,0} & p_{107,1} & \cdots & p_{107,80} \end{bmatrix} \quad (1)$$

## 4 Probabilistic Models

There are four models required for the implementation of a Markov localisation method such as MCL. These models are being employed to represent mainly the particles' pose and perception. A *motion model* expresses the probability for certain actions to move the robot to certain relative positions. A *sensor model* describes the probability for taking certain measurements at certain locations [5]. A *noise model* adds Gaussian noise to the motion model, so as to spread the particle density. Finally, an *odometry model* can be incorporated to conduct a waypoint navigation. The MCL's particle filter obviously acts on the odometry model, refining its direction towards a predefined location.

### 4.1 Motion Model

The effects of actions on the robot's pose are represented by a motion model [5]. At every time step the robot's kinematics updates the particle's kinematics. The kinematic update is instrumented by Eqs. 2, 3, and 4, with  $d$  denoting the delayed linear displacement, and  $\vartheta$  the delayed rotational one. The noise terms  $\mathcal{E}$  are random Gaussian noise explained in Section 4.3.

$$\begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} + (d + \varepsilon_x) \cos \theta \\ y_{k-1} + (d + \varepsilon_y) \sin \theta \\ \theta_{k-1} + (\vartheta + \varepsilon_\theta) \end{bmatrix} \quad (2)$$

$$d = \sqrt{(x_{k-1} - x_k)^2 + (y_{k-1} - y_k)^2} \quad (3)$$

$$\vartheta = |\theta_{k-1} - \theta_k| \quad (4)$$

In the current work, a priory virtual map  $M$  is provided for the interaction of the particles with their environment. The ones which violate the map's boundaries are regenerated with a random pose within the map. The map is represented here by a pair of points, each defining the start-end coordinates of a line. Fig. 3 depicts the line representation.

Algorithm 2 illustrates the map validation procedure. Step 1 describes a state estimate for the current particle  $q$ . The mapFlag at step 2 is set to true, as it decides later on whether the state of the evaluated particle violates the map’s boundaries. The for-loop in step 3 validates the particle’s  $x, y$  state coordinates with each line in the map. A newly random coordinate state is chosen for the particle which violates the boundaries, or, the particle is left intact otherwise.

---

**Algorithm 2** Particle map (M) validator

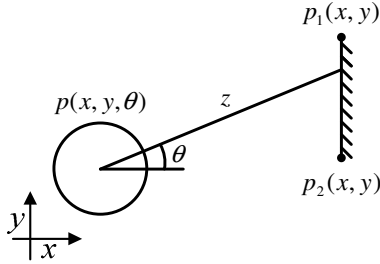
---

```

1: Estimate  $X_k^{[q]}$ 
2: mapFlag = true
3: for  $i = 1$  to  $M$  do
     $j = (i + M - 1) \% M$ 
    if  $\left( \begin{array}{l} (M_y^{[i]} > q_y) \neq (M_y^{[j]} > q_y) \ \& \\ q_x < \frac{(M_x^{[j]} - M_x^{[i]})(q_y - M_y^{[i]})}{(M_y^{[j]} - M_y^{[i]})} + M_x^{[i]} \end{array} \right)$ 
        mapFlag =  $\overline{\text{mapFlag}}$ 
    end if
end for
4: if mapFlag = true
    return rand( $q^{[x]}$ )
end if

```

---



**Fig. 3.** Beam representation of a particle

## 4.2 Sensor Model

The sensor model is described by the likelihood probability  $p(z|x)$  of Eq. 6. It is a probabilistic error between the particle’s virtual beam range  $z$  (Eq. 5), as shown in Fig. 1, and the real robot sensor range  $s$ , acquired from the visual depth descriptor. The product of the likelihood probabilistic error between virtual and real sensor ranges, for particle  $q$ , constitutes the particle’s goodness of fit (weight, Eq. 7). The

weight is an assessment of how close the particle approximates the robot's state vector  $\mathbf{x}$ .

$$z = \frac{[p_2(y) - p_1(y)][p_1(x) - x_q] - [p_2(x) - p_1(x)][p_1(y) - y_q]}{[p_2(y) - p_1(y)]\cos\theta - [p_2(x) - p_1(x)]\sin\theta} \quad (5)$$

$$p(z | \mathbf{x}) = \begin{cases} 1, & \text{if } |s - \bar{z}| > \sigma_z \\ 1 - \frac{|s - \bar{z}|}{\sigma_z}, & \text{otherwise} \end{cases} \quad (6)$$

$$w_q = \prod_{i=1}^S p(z^{[i]} | \mathbf{x}) \cdot w^{[i]} \quad (7)$$

where  $\bar{z}$  is the average of a measurement sample, and  $\sigma_z$  the standard deviation. Ultimately,  $S$  is the max number of sensors, which has been defined primarily from the visual descriptor as 6 (region matrix  $\mathbf{A}$ , Eq. 1). Here, for consistency the sensor model assigns six beam sensors to each particle with  $10^\circ$  angular difference.

### 4.3 Noise Model

The noise model is generated using a zero-mean Gaussian distribution. For the translational noise (see Eq. 8) the delayed distance  $d$  is multiplied with the Gaussian  $N$  to acquire a random noise scalar, which is added with  $d$  in the kinematic update of Eq. 1. The exponent  $n$  is chosen for introducing more ( $=1$ ) or less noise ( $>1$ ). Similarly for the rotational noise.

$$\begin{aligned} \mathcal{E}_x &\sim N(0, \sigma_x^2)^n \times d \\ \mathcal{E}_y &\sim N(0, \sigma_y^2)^n \times d \\ \mathcal{E}_\theta &\sim N(0, \sigma_\theta^2)^n \times \vartheta \end{aligned} \quad (8)$$

The random Gaussian noise is sampled from a multivariate distribution  $N$ , which is characterised by a density function [1] with  $\mu = 0$  and  $\sigma = 1$  (see Eq. 9).

$$N = \det(2\pi\Sigma)^{-1/2} \exp\left\{-\frac{1}{2} \mathbf{v}^T \Sigma^{-1} \mathbf{v}\right\} \quad (9)$$

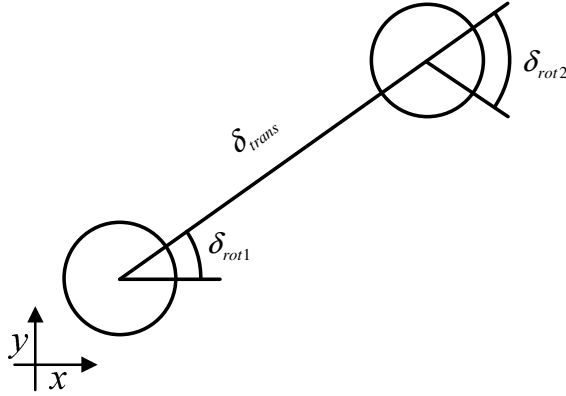
where  $\Sigma$  is the covariance matrix, and  $\mathbf{v}$  a random vector.

### 4.4 Odometry Model

An odometry model is particularly useful for waypoint localisation. The model describes the robot's motion in a rotational and a translational displacement. The rotational displacement is estimated by the arctangent between the current  $(x, y)$  and the next  $(x', y')$  coordinate pairs. The translational displacement is given by the difference of the same coordinate pairs, estimated with the Euclidean distance. The model's displacements are described by Eq. 8

$$\begin{aligned}
\delta_{rot1} &= \text{atan2}(y'-y, x'-x) \\
\delta_{trans} &= \sqrt{(x-x')^2 + (y-y')^2} \\
\delta_{rot2} &= \theta' - \theta
\end{aligned} \tag{8}$$

The robot's motion at  $(t-1, t]$  is approximated by a sequence of transitions as shown in Fig. 2 [1]. Initially, a rotational transition  $\delta_{rot1}$  is performed, followed by a translational transition,  $\delta_{trans}$ , and a consequent rotational  $\delta_{rot2}$ . The robot's transitional displacements over a number of nodes completes the waypoint localisation. The purpose of the MCL's particle filter is to act on the odometry model by refining the robot's heading direction towards a designated node.



**Fig. 4.** Representation of the odometry model in two rotational, and one translational transition

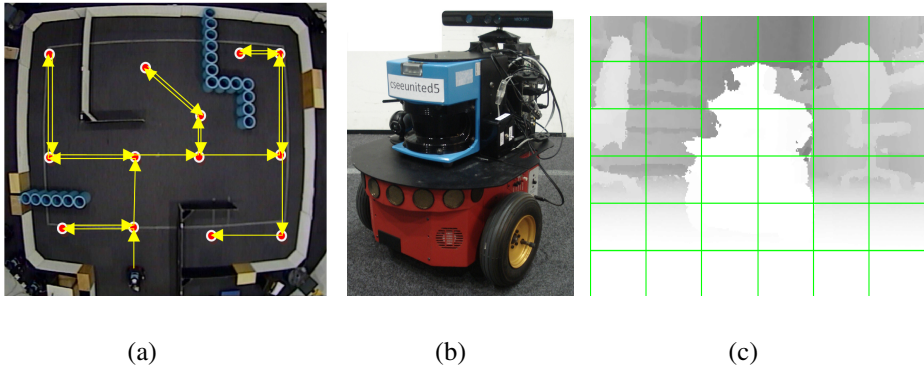
## 5 Experimental Results

### 5.1 Experimental Setup

The Essex robotic arena was the main experimental hall where the experimental procedure took place. The environmental setup illustrated in Fig. 5(a) was used for the conduction of the experimental cases. The experiments were separated in three different cases so as to show the efficiency of the visual descriptor when applied for MCL, and the Kinect sensor when compared to proximity sensor localisation. In addition, the initial experiments were taken by an Activmedia Pioneer robot (Fig. 5(b)) for testing purposes before the actual wheelchair is used to carry human beings.

We have used an odometry model to navigate the robot through 19 nodes (waypoints) as Fig. 5(a) also depicts. Lastly, Fig. 5(c) illustrates the region depth map using the robot's onboard Kinect sensor.





**Fig. 5.** (a) The experimental environment, (b) The Kinect-enabled mobile robot, (c) The grid depth map

## 5.2 Localisation Cases

### *a) Odometry Model*

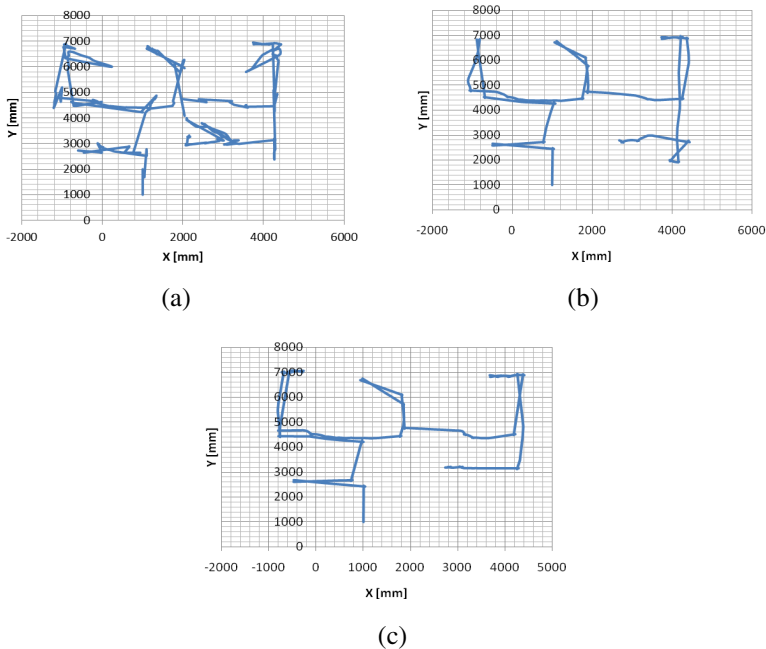
Employing an odometry model only, the robot is obviously lost so an incorporated obstacle avoidance undertakes to help the localisation. Fig. 6(a) presents an experimental trial for navigating with an odometry model. The trajectory seems rather unstable due to several maneuvers performed by the robot for the avoidance of the walls. From the poor localisation performance shown in this graph, we see the importance of how useful would be a probabilistic filter for the autonomous transportation of humans.

### *b) Sonar-Based MCL*

The classical MCL method, as initially proposed by [2, 3], employed proximity sensors. In this case, we performed MCL with an array of six sonar sensors. The resultant trajectory of Fig. 6(b) demonstrates a much smoother trajectory when we introduce the MCL method to the odometry model. A diminished use of the obstacle avoidance algorithm is observed, and the path now is drawn clearer.

### *c) Kinect-Based MCL*

The proposed MCL method, enhanced with an incorporated region locator descriptor, shows in Fig. 6(c) to perform better than the sonar-based MCL, and apparently better than the odometry model on its own. The performance enhancement is owed to fact that the Kinect sensor utilises a complete coverage of the frontal view. This turns out to be essentially useful for the sensor model, as now the particles are evaluated more accurately. The Kinect's depth coverage in  $x$  and  $y$  image coordinates, contrary to lasers acting in  $x$  only, is proven also to be safer and more robust for a robotic wheelchair use.



**Fig. 6.** Experimental trajectories. (a) Using an odometry model, (b) Using sonar-based MCL, (c) Using Kinect-based MCL

## 6 Conclusions

In this work an attempt has been made to introduce a region locator descriptor, through a depth image, to Monte Carlo Localisation method. The Microsoft's Kinect sensor was the main apparatus incorporated to enhance the localisation performance. Indeed, the experimental work showed that the suggested descriptor can improve the localisation accuracy when compared with proximity sensors.

Our future directions will focus on testing colour and edge histogram descriptors for MCL. Eventually, to test the performance of the current method on different wheelchair types for indoor and outdoor localisation.

**Acknowledgements.** This research has been financially supported by the EU Interreg IV A 2 Mers Seas Zeeën Cross-border Cooperation Programme – SYSIASS: Autonomous and Intelligent Healthcare System. More details can be found at the project's website <http://www.sysiass.eu/>. We would also like to thank Robin Dowling for the technical support.

## References

1. Thrun, S., Burgard, W., Fox, D.: Probabilistic Robotics. MIT Press, Cambridge (2005)
2. Dellaert, F., Fox, D., Burgard, W., Thrun, S.: Monte Carlo Localization for Mobile Robots. In: IEEE International Conference on Robotics and Automation, pp. 1322–1328 (1999)

3. Fox, D., Burgard, W., Dellaert, F., Thrun, S.: Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. In: Proceedings of the Sixteenth National Conference on Artificial Intelligence, pp. 343–349 (1999)
4. Elinas, P., Little, J.J.: sMCL: Monte-Carlo Localization for Mobile Robots with Stereo Vision. In: Proceedings of Robotics: Science and Systems, pp. 373–380 (2005)
5. Rofer, T., Jungel, M.: Vision-Based Fast and Reactive Monte-Carlo Localization. In: IEEE International Conference on Robotics and Automation, pp. 856–861 (2003)
6. Wikipedia contributors, “Kinect”, Wikipedia: The Free Encyclopedia. Wikimedia Foundation (2004), <http://en.wikipedia.org/wiki/Kinect>
7. Zhao, X., Li, X., Tan, T.: A Novel Landmark Tree Based Self-Localization and Path-Planning Method for an Intelligent Wheelchair. In: Proceedings of the 9th IEEE International Workshop on Robot and Human Interactive Communication, pp. 84–89 (2000)
8. Yaqub, T., Tordon, M.J., Katupitiya, J.: A Procedure to Make the Probabilistic Odometry Motion Model of an Autonomous Wheelchair. In: Proceedings of the 2006 IEEE International Conference on Mechatronics and Automation, pp. 526–531 (2006)

# Topological Mapping with Image Sequence Partitioning

Hemanth Korrapati<sup>1</sup>, Jonathan Courbon<sup>1</sup>, and Youcef Mezouar<sup>1,2</sup>

<sup>1</sup> Clermont Université, Université Blaise Pascal, Institut Pascal, BP 10448,  
CLERMONT-FERRAND, F-63000

<sup>2</sup> CNRS, UMR 6602, IP, Aubiere, F-63171  
firstname.lastname@univ-bpclermont.fr

**Abstract.** Topological maps are vital for fast and accurate localization in large environments. Sparse topological maps can be constructed by partitioning a sequence of images acquired by a robot, according to their appearance. All images in a partition have similar appearance and are represented by a node in a topological map. In this paper, we present a topological mapping framework which makes use of image sequence partitioning (ISP) to produce sparse maps. The framework facilitates coarse loop closure at node level and a finer loop closure at image level. Hierarchical inverted files (HIF) are proposed which are naturally adaptable to our sparse topological mapping framework and enable efficient loop closure. Computational gain attained in loop closure with HIF over sparse topological maps is demonstrated. Experiments are performed on outdoor environments using an omni-directional camera.

**Keywords:** Topological Mapping, Omni-directional Vision, Loop Closure.

## 1 Introduction

Powerful loop closing techniques for topological maps have been introduced recently [2], [4]. Most of them produce dense topological maps, in which every acquired image stands as a node in the topological graph. Sparser topological maps can be built by representing sets of contiguous images with similar appearance features as places. Each place is represented by a node in the topological graph. We refer to the partitioning of an image sequence into places as Image Sequence Partitioning (ISP). Maps with fewer nodes reduce computational complexity involved in loop closure and map merging.

We use a hierarchical topological mapping framework which represents places as nodes of a graph and the adjacency of places by edges. Each node/place consists of all the images acquired in that a set of representative features is constructed out of member images. The representative features are used in evaluating node-image similarity during ISP.

Another contribution of this paper is the proposal of Hierarchical inverted files (HIF) which combine the power of regular inverted files with our map structure and enable efficient loop closure at both node and image levels. Finally, a spatial constraint exploiting the omnidirectional image structure is introduced to reduce the number of false positives and increase the precision of loop closures. As omni-directional image data is used for our experimentation a robot heading invariant loop closure is possible.

## 2 Related Work

Many loop closure algorithms for topological mapping have been proposed and tested in both indoor [12], [2], [6] and outdoor environments [4], [5], [1], [9].

In [12], topological maps for indoor environments are built by segmenting the topological graph of the environment using normalized graph-cuts algorithm resulting in subgraphs corresponding to convex areas in the environment. In [6] SIFT features were used to perform matching over a sequence of images by detecting transitions between individual indoor locations depending on the number of SIFT features which can be successfully matched between the successive frames. In [10] fingerprint of an acquired image is generated using omni-directional image and laser readings, and these fingerprints are compared to those of the previous images. All of these works were focused on indoor environments which are simpler to partition due to convex spaces.

A topological mapping framework using incremental spectral clustering has been presented in [11]. Nodes containing similar images are constructed using incremental spectral clustering over the affinity matrix of the images, thereby producing a topological graph. An optical flow based ISP technique was presented in [9] for topological mapping in outdoor environments using a quad rotor robot. Optical flow is used to discover change in environmental appearance. In [1], gist features were used to cluster images with similar appearance for topological map construction.

## 3 Image Sequence Partitioning

Figure 1 depicts a global view of our framework, in which we can see a modular view of ISP enclosed by a red dashed line. As can be seen from Figure 1, ISP consists of three main modules: node level loop closure, evaluation of similarity to current place and new node formation.

Given a query image, initially SURF [3] features are extracted from the image. Using the SURF features, we evaluate the node-image similarity of the query image with all the nodes in the graph except the current place node and pick out the top  $k$  similar nodes. The top  $k$  similar nodes are assigned to the set of winning nodes  $N_w$ . This process is called node level loop closure as it finds the previously visited places (nodes) most similar to the query image. Only the representative feature sets of the nodes are used to compute the node-image similarities during node level loop closure. In our framework, the representative features of a node are the SURF features of the first image augmented to the node.

An empty  $N_w$  indicates loop closure failure ; that is, query image is not similar to any of the previously visited places. In that case, query image similarity to the current node is evaluated. If the similarity of query image with current place node is above a certain threshold, current node is augmented with the query image. If the query image is not similar to current node also, a new node is created with the query image. But if  $N_w$  is not empty indicating a loop closure, then the set of winning nodes can be considered for a thorough image level loop closure.

Algorithm 1 shows the steps involved in ISP. The 'node\_level\_loop\_closure' function in lines 4 is discussed in detail in sections 4. The function 'current\_node\_image\_similarity'

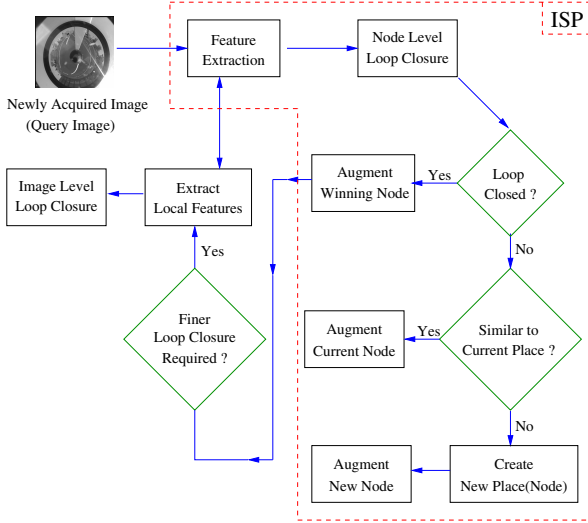


Fig. 1. A global view of our topological mapping framework

evaluates the similarity of the current node with that of the query image. This is done by matching the SURF features of the query image to that of the features of the current place node. Feature matching is performed as proposed in [7]. Two features are considered to be matched if the ratio of the best match and the second best match is greater than 0.6.

## 4 Loop Closure and Hierarchical Inverted Files

Node and Image level loop closures are performed at image level using visual words corresponding to the SURF features of the image. Given a query image, to find the most similar reference image in the database, [8] uses a Tf-Idf based similarity score for all the reference images in the database and the query image and select the reference images corresponding to the top  $n$  similarity scores. We use an inverse methodology to compute image similarities for loop closure. The steps involved are enumerated as follows:

1. Let the set of reference images be  $\mathbf{I} = \{I_1, I_2, \dots, I_M\}$ . We consider a histogram  $H$  with the number of bins corresponding to the number of reference images,  $M$ .
2. Extract the set of visual words  $\mathbf{W} = \{w_1, w_2, \dots, w_p\}$  from the query image,  $I_q$ .
3. For each visual word  $w_i$ , using the inverted file  $IF_i$  of the word, we extract the reference image indexes  $\mathbf{I}^{w_i} = \{I_1^{w_i}, I_2^{w_i}, \dots\}$  in which the word has been previously seen. The histogram bins corresponding to these extracted reference images are incremented by a factor of Tf-Idf of the corresponding word.

$$H[I_j^{w_i}] = H[I_j^{w_i}] + Tf - Idf(I_j^{w_i}, w_i) \quad (1)$$

**Algorithm 1.** Image Sequence Partitioning Algorithm

---

```

1: procedure PROCESS_QUERY_IMAGE( $\mathbf{T}, I_q, n_c$ )  $\triangleright \mathbf{T}, I_q, n_c, \mathbf{T}.\mathbf{N}$  are the Topological graph, query
   image, current node in topological graph & node-set of topological graph respectively.
2:    $\mathbf{N}' = \mathbf{T}.\mathbf{N} - \{n_c\}$   $\triangleright$  Reference node set excluding current node.
3:    $\mathbf{N}_w = \{\}$   $\triangleright$  Set of winning nodes.
4:    $\mathbf{N}_w = \text{Node\_Level\_Loop\_Closure}(\mathbf{N}', I_q, Th_s)$ 
5:   if is_empty( $\mathbf{N}_w$ ) then
6:     if current_node_image_similarity( $n_c, I_q$ ) >  $Th_t$  then
7:        $n_c.add\_image(I_q)$ 
8:     else
9:        $n = \text{new\_node}()$ 
10:       $n.add\_image(I_q)$ 
11:      update_map( $\mathbf{T}, n$ )
12:    end if
13:  else
14:     $\mathbf{N}_w = \text{get\_top\_k\_similar\_nodes}(\mathbf{N}_w)$ 
15:     $\mathbf{I}_w = \text{get\_images\_of\_nodes}(\mathbf{N}_w)$ 
16:     $n^* = \text{Image\_Level\_Loop\_Closure}(\mathbf{I}_w, \mathbf{N}_w, I_q)$ 
17:    add_image_to_node( $I_q, n^*$ )
18:    update_map( $\mathbf{T}, n^*$ )
19:  end if
20: end procedure

```

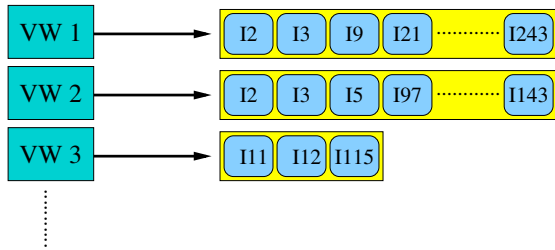
---

The resulting histogram can be interpreted to contain the degrees of similarity of the query image with respect to the reference images. As we can see, the loop closure computation time only depends on number of words in the query image and the average inverted file length at that instant. As a result loop closure time does not increase so steeply as is the case with forward method. A closely related work can be found in [5], [2]. But this method is suitable only for loop closure over dense topological maps but not for sparse topological maps in which each node represents multiple images. With a change in the inverted file structure, we can adapt this similarity evaluation method to sparse topological maps.

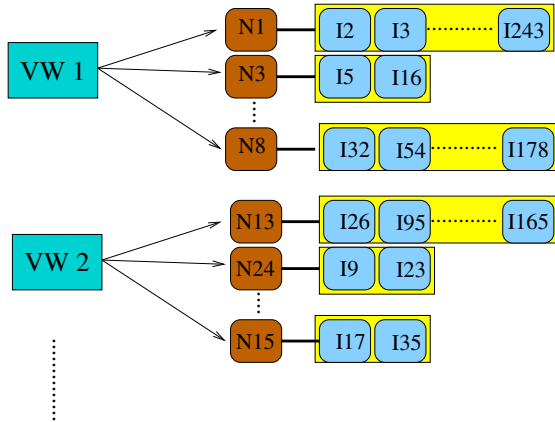
A regular inverted file corresponding to a visual word simply contains a list of all previous images which contained the word. We associate Hierarchical inverted files (HIF) to each visual word. As the name suggests, HIF contain two levels. The first level consists of the ids of the nodes in which the visual word occurred previously. The second level consists of small child inverted files attached to each of the node id. These image ids indicate the images belonging to the parent node in which the visual word has occurred. Each child inverted file corresponding attached to a node of the topological graph, contains the list of all previous images belonging to the parent node in which the word has occurred. The difference between traditional inverted files and HIFs is illustrated in figure 2. To perform a node level loop closure using HIF, we do not have to go through the entire HIF, but its sufficient to go through first level (node ids) of the HIFs. For an image level loop closure using HIF, we only have to traverse through those child inverted files corresponding to the winning nodes; which form only a fraction of the total HIF. Thus, HIFs offer computational gain in loop closure when compared to

regular inverted files which is demonstrated in section 5. Algorithms 2 and 3 give a clearer picture of the node level and image level loop closures. There can be multiple winning images given by the image level loop closure and hence multiple corresponding nodes, but for the sake of simplicity we do not represent that in the algorithm.

It can be observed in the image level loop closure in algorithm 3, that in lines 9 through 12 we impose spatial constraints on the visual word matches. On line 9, we get the best visual word matches - this becomes important in case of unequal number of occurrences of the same visual word in both images. In this case, we select the lower of the two and find their matches in the other images such that they are the closest w.r.t the vertical distance. In lines 10 through 12 we sum up the distances of these multiple occurrences using a gaussian kernel with zero mean and variance  $\sigma$ . This constraint helps in reducing false positives. There can be multiple winning images given by the image level loop closure and hence multiple corresponding nodes, but for the sake of simplicity we do not represent that in the algorithm.



(a)



(b)

**Fig. 2.** (a) represents a traditional inverted file. (b) represents a hierarchical inverted file. VW1, VW2,... represent visual words. N1, N2,... represent node ids in the first level of HIF and I1, I2, I3, ... represent the image ids in the inverted file.



---

**Algorithm 2.** Node Level Loop Closure Algorithm
 

---

```

1: procedure NODE_LEVEL_LOOP_CLOSURE( $\mathbf{N}'$ ,  $I_q$ ,  $Th_s$ )
2:    $\mathbf{N}' = \mathbf{T} \cdot \mathbf{N} - \{n_c\}$ 
3:    $H = \text{Histogram}(\text{size\_of}(\mathbf{N}'))$ 
4:    $\mathbf{W} = \text{extract\_and\_quantize\_Features}(I_q)$ 
5:   for each word  $w_i$  in  $\mathbf{W}$  do
6:      $HIF_i = \text{get\_hierarchical\_inverted\_file}(w_i)$ 
7:     for each node  $n_j$  in  $HIF_i$  do
8:        $H[n_j] = H[n_j] + 1$ 
9:     end for
10:  end for
11:   $\mathbf{N}^* = \text{get\_winners\_from\_histogram}(H, Th_s)$ 
12:  return  $\mathbf{N}^*$ 
13: end procedure

```

---



---

**Algorithm 3.** Image Level Loop Closure Algorithm
 

---

```

1: procedure IMAGE_LEVEL_LOOP_CLOSURE( $\mathbf{I}_w, \mathbf{N}_w, I_q$ )
2:    $S = \text{Array}(M)$ 
3:    $\mathbf{W} = \text{extract\_and\_quantize\_Features}(I_q)$ 
4:   for each word  $w_i$  in  $\mathbf{W}$  do
5:      $HIF_i = \text{get\_hierarchical\_inverted\_file}(w_i)$ 
6:     for each node  $n_j$  in  $\mathbf{N}_w$  do
7:        $IF_i^{n_j} = \text{get\_inverted\_file\_of\_node}(HIF_i, n_j)$ 
8:       for each entry  $f_i$  in  $IF_i^{n_j}$  do
9:          $D = \text{get\_matching\_word\_distances}()$ 
10:         $sim = 0$ 
11:        for each distance  $d_j$  in  $D$  do
12:           $sim = sim + g(d_j, \sigma)$ 
13:        end for
14:         $H[f_i] = H[f_i] + Tf - Idf(f_i, w_i) * sim$ 
15:      end for
16:    end for
17:  end for
18:   $\mathbf{I}^* = \text{get\_winner\_from\_histogram}(H)$ 
19:   $n^* = \text{get\_corresponding\_node}(\mathbf{I}^*)$ 
20:  return  $n^*$ 
21: end procedure

```

---

## 5 Experiments

Our experimental setup consists of a Pioneer P3DX robot equipped with an omnidirectional camera. A laptop equipped with an Intel Centrino 2 processor running ubuntu 9.04 is used for data processing. The experiments were carried out in our artificial urban environment - PAVIN, shown in Figure 3. The environment contains roads, artificial buildings, and a variety of real-world road settings like junctions, traffic lights, roundabouts, curved roads and dead ends.

Omnidirectional images were acquired at a frame rate of 2 fps, as the robot moves along a manually controlled trajectory. Image data was acquired in four installments(A, B, C and D) at very different times of two days and hence contained significant illumination variation. Figure 4 shows the parts of the environment through which the robot traversed during each installment. We took care that data from all the four installments contained overlaps so as to put our loop closure algorithm to test.

We constructed two datasets by combining data from all the four installments. Dataset-6560 was obtained by combining data of installments A, C and D. It contains 6560 images with 52 possible loop closures. Another dataset Dataset-11200 was obtained by a combination of all the four installments. It contains 11200 images and 71 possible loop closures. The number of loop closures were determined by manually examining the datasets.



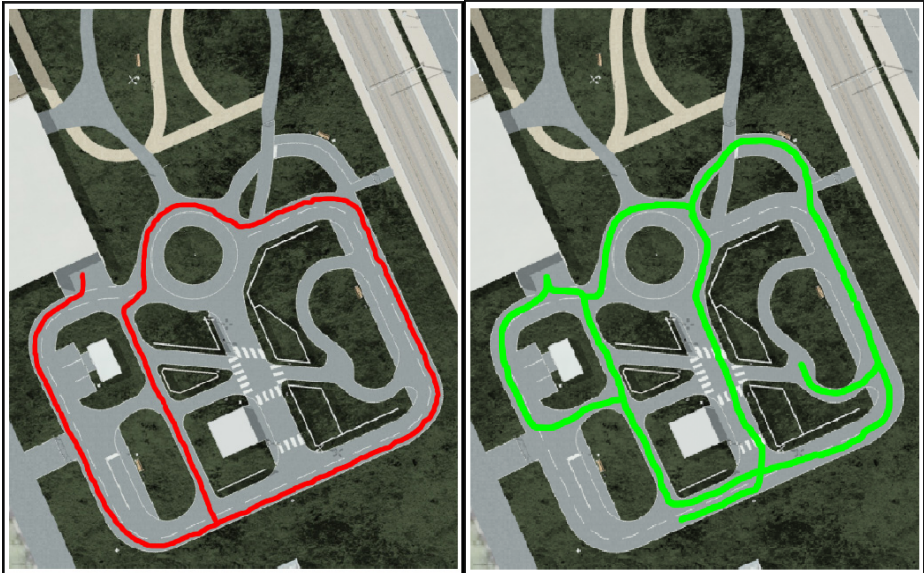
(a) Pavin - Straight View

(b) Pavin - Top View

**Fig. 3.** Pavin - Urban Testbed for Mobile Robots

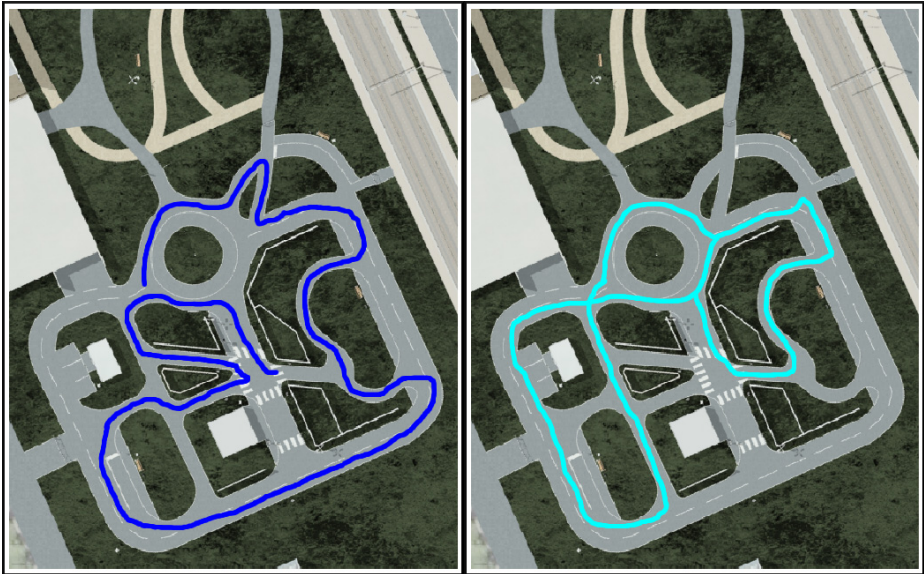
### 5.1 ISP - Sparsity

The number of nodes in a topological map indicate its sparsity. As each node in the topological graph should ideally represent a place in the environment, any feature which represents stable appearance throughout a place and changes with another place produces an ideal topological map. The sparsity of these ideal maps represents the optimal sparsity of the actual environment. But practically, an ideal topological map is far from being attained. The obtained map differs based on the features used to perform ISP.



(a) Installment A

(b) Installment B



(c) Installment C

(d) Installment D

**Fig. 4.** Shows paths traversed by the robot during each image acquisition installment

We have experimented using SURF128, U-SURF128, and SIFT features, out of which we found out that U-SURF128 is the best suitable feature for attaining sparse maps. The problem with SIFT is that it produces huge number of features out of which there are huge number of feature mismatches, which makes it impossible to perform feature matching in real-time and also cannot represent appearance properly. On the other hand SURF128 and U-SURF128 produce lesser number of features with relatively very few number of mismatches. However U-SURF128 becomes a better choice as it exhibits more discriminative power on planar environments and also has lower computation time. In our implementation we used a matching threshold of 60% for SURF feature matching and a threshold of 40% for SIFT features.

Another important factor that effects stability of appearance is the image distortion. The features directly extracted from warped omnidirectional images are unstable as the appearance of keypoint patch changes very much even with a small displacement of the camera. Hence it is likely that the maps produced using warped images contain greater number of place nodes. This happens because due to the feature instability, each place can be understood as multiple adjacent places and hence multiple nodes in the topological graph. The undistorted (unwrapped) images produce relatively sparser maps. Tables [1\(a\)](#) and [1\(b\)](#) show the sparsity of maps produced by ISP using different features on warped and unwrapped images. We can see that U-SURF128 is the best performing feature producing the most sparse maps.

**Table 1. SPARSITY**

(a) Sparsity - Warped

	<b>SURF128</b>	<b>U-SURF128</b>	<b>SIFT</b>
<b>DATASET-6560</b>	539	502	1582
<b>DATASET-11200</b>	756	742	2795

(b) Sparsity - Unwarped

	<b>SURF128</b>	<b>U-SURF128</b>	<b>SIFT</b>
<b>DATASET-6560</b>	504	473	1037
<b>DATASET-11200</b>	737	723	1257

## 5.2 Accuracy

The most sparse map may not guarantee an accurate map. Only those maps with accurate place partitioning are accurate and can lead to accurate loop closures. Thus a good mapping technique is one which provides an optimal combination of sparsity and accuracy.

Given a query image, first we perform loop closure at node level and then at image level if more accuracy is required. [Table 2](#) shows the number of loop closures detected and the number of false positives obtained on Dataset-6560 and Dataset-11200 respectively by node level loop closure. The results in these tables were obtained using unwrapped omni-directional panoramas. We can see that a few loop closures are missed

out, and some false positives are observed. The inaccuracy can be attributed to the imperfections in ISP. There is a direct relation between accuracy of the maps and ISP. The way in which we perform topological mapping does not induce any information loss. In other words we do not perform any kind of sampling or selection of reduced number of features and hence there is no information lost during the mapping process. However inaccurate loop closure detection might occur due to inaccurate partitioning of places. For example a place can be represented by two nodes by partitioning it inaccurately during ISP due to appearance feature instability. As a result, during node level loop closure using a query image, both of the nodes representing that place may not get high similarity scores and hence the loop closure becomes inaccurate. A good ISP algorithm produces maps with minimum number of these situations.

Image level loop closure accuracy depends on the accuracy of node level loop closure. If node level loop closure selects an inaccurate set of winning nodes, then as a result, image level loop closure also becomes inaccurate. However in case of an accurate node level loop closure, we have observed that 99% accuracy was possible in image level loop closure irrespective of the ISP technique. Figure 5 shows two loop closure scenarios that occurred in our mapping.

**Table 2.** NODE LEVEL LOOP CLOSURE ACCURACY

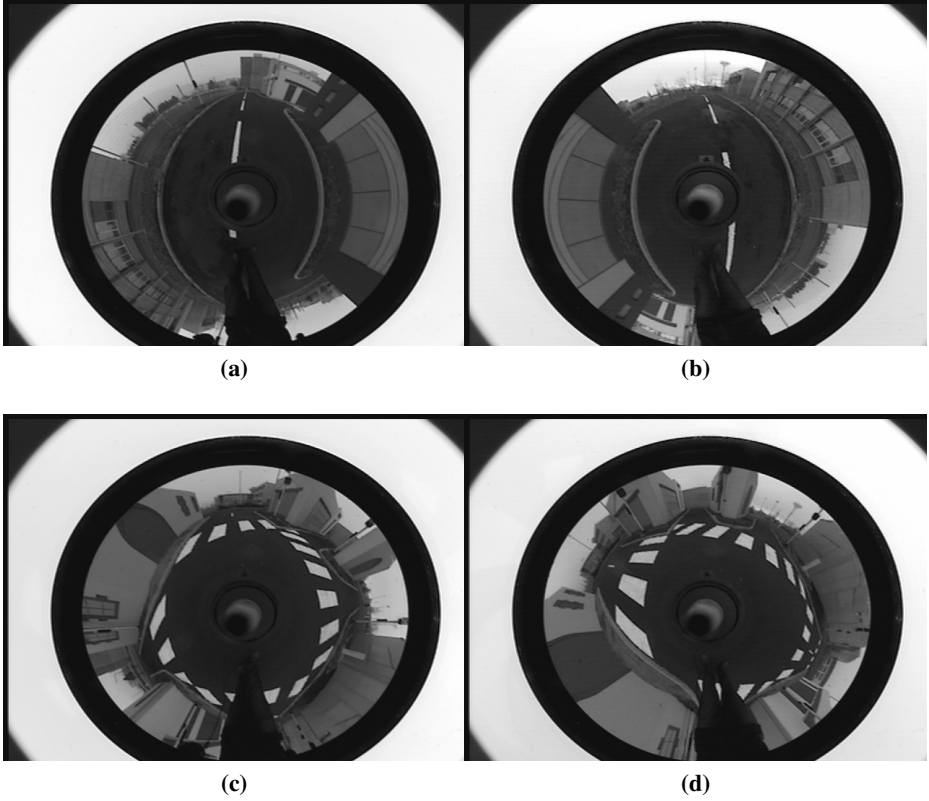
	#(LC)	#(FP)
<b>Dataset-6560</b>	46	2
<b>Dataset-11200</b>	67	4

### 5.3 Computational Time

Table 3 shows average computational time (in milliseconds) taken by each stage of our topological mapping framework in processing each query frame on both Dataset-6560 and Dataset-11200. The abbreviations NLLC, LFE+QUANT, ILLC stand for Node Level Loop Closure, Local Feature Extraction & Quantization and Image Level Loop Closure respectively. NLLC is the node level loop closure which involves extracting the most similar nodes using HIFs as mentioned in Algorithm 2. This takes  $10ms$  as shown in table 3 and requires additional  $50ms$  in order to compare with the current place node whenever needed. Obviously, local feature extraction (LFE) ( $200ms$ ) and quantization (QUANT) ( $70ms$ ) time is constant for every acquired frame. Actually, time required for both of these tasks increases with the number of features in an image.

Computation time of image level loop closure (ILLC) is too low. This low computation time is the result of using hierarchical inverted files(HIF). As we mentioned before, HIFs make the loop closure computation almost independent of the number of reference images and also in our case, nodes of the topological graph.

Figures 6(a) and 6(b) show graphs comparing the loop closure times of our HIF-based method and without using HIF (non-HIF based). Red curves in the graphs indicate the time taken for feature quantization, node level loop closure and image level loop closure, for each image frame in a sequence. The blue curves represent the time taken by

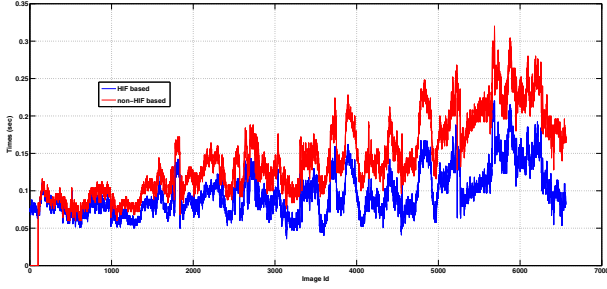


**Fig. 5.** Example loop closures

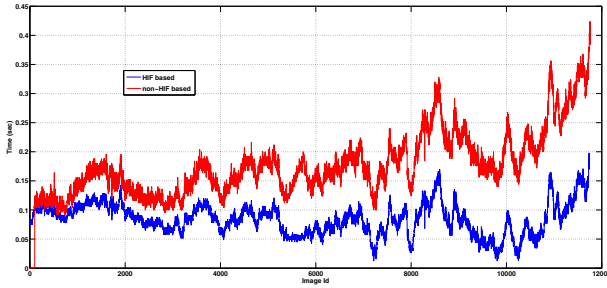
**Table 3.** AVERAGE COMPUTATION TIMES (in ms)

<b>NLLC</b>	<b>LFE+QUANT</b>	<b>ILLC</b>
10 + 50	200 + 70	21

feature quantization and similarity score generation using inverted files by using inverse similarity evaluation methodology. We can see that the loop closure time of non-HIF based method increases relatively more with the increase in the number of images in the map, while our method using our method, loop closure time increases more slowly. Also, the performance gain becomes more prominent in case of huge datasets (huge number of images) as can be seen in the figure 6(b) corresponding to Dataset-11200, which contains 11200 images. The non-HIF based loop closure time for Dataset-11200 increases less steeply than that of Dataset-6560. This happened because the average number of features of Dataset-11200 is lesser than that of Dataset-6560 and as a result it takes lesser time to process each reference frame. This efficiency of our HIF-based method can be attributed to the combination of sparse topological mapping and HIFs



(a) Loop Closure time - Dataset-6560



(b) Loop Closure time - Dataset-11200

**Fig. 6.** Loop closure computation times of non-HIF based loop closure and HIF based loop closure on maps generated on our datasets

for efficient map storage. The representational power of HIFs saved lot of computation involved in loop closure.

## 6 Conclusion

We proposed a sparse topological mapping framework involving two levels of representation. Image Sequence Partitioning(ISP) is employed to produce sparse maps. Hierarchical Inverted Files(HIF) were proposed to take advantage of the sparse maps and perform quick loop closure. Sparsity of the maps produced by different features are analysed and the accuracy is evaluated. Finally, our framework is evaluated on computational time required for loop closure.

**Acknowledgment.** This work has been supported by the National Research Agency (l'ANR) of France (under the projects CityVip and R-Discover) and a grant (I09200) from Gyeonggi Technology Development Program funded by Gyeonggi Province of Korea.

## References

1. Kosecka, J., Murillo, A.C., Campos, P., Guerrero, J.J.: Gist vocabularies in omnidirectional images for appearance based mapping and localization. In: 10th IEEE Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras (OMNIVIS), held with Robotics, Science and Systems (2010)
2. Angeli, A., Filliat, D., Doncieux, S., Meyer, J.-A.: A fast and incremental method for loop-closure detection using bags of visual words. *IEEE Transactions On Robotics, Special Issue on Visual SLAM* (2008)
3. Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: Speeded-up robust features (surf). *Comput. Vis. Image Underst.* 110(3), 346–359 (2008)
4. Cummins, M., Newman, P.: FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance. *The International Journal of Robotics Research* 27(6), 647–665 (2008)
5. Cummins, M., Newman, P.: Highly scalable appearance-only slam fab-map 2.0. In: *Robotics Science and Systems (RSS)*, Seattle, USA (June 2009)
6. Koseck, J., Li, F., Yang, X.: Global localization and relative positioning based on scale-invariant keypoints. *Robotics and Autonomous Systems* 52(1), 27–38 (2005)
7. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* 60, 91–110 (2004)
8. Nister, D., Stewenius, Scalable, H.: recognition with a vocabulary tree. In: *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2006*, vol. 2, pp. 2161–2168 (2006)
9. Nourani-Vatani, N., Pradalier, C.: Scene change detection for topological localization and mapping. In: *IEEE/RSJ Intl. Conf. on Intelligent Robotics and Systems, IROS* (2010)
10. Tapus, A., Siegwart, R.: Incremental robot mapping with fingerprints of places. In: *IEEE/RSJ Intl. Conf. on Intelligent Robotics and Systems (IROS)*, pp. 2429–2434 (2005)
11. Valgren, C., Duckett, T., Lilienthal, A.: Incremental spectral clustering and its application to topological mapping. In: *In Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 4283–4288 (2007)
12. Zivkovic, Z., Booi, O., Kröse, B.: From images to rooms. *Robot. Auton. Syst.* 55, 411–418 (2007)



# Visual Memory Update for Life-Long Mobile Robot Navigation

Jonathan Courbon<sup>1</sup>, Hemanth Korrapati<sup>1</sup>, and Youcef Mezouar<sup>1,2</sup>

<sup>1</sup> Clermont Université, Université Blaise Pascal, Institut Pascal, BP 10448,  
CLERMONT-FERRAND, F-63000

<sup>2</sup> CNRS, UMR 6602, IP, Aubière, F-63171

**Abstract.** A central clue for implementation of visual memory based navigation strategies relies on efficient point matching between the current image and the key images of the memory. However, the visual memory may become out of date after some times because the appearance of real-world environments keeps changing. It is thus necessary to remove obsolete information and to add new data to the visual memory over time. In this paper, we propose a method based on short-term and long term memory concepts to update the visual memory of mobile robots during navigation. The results of our experiments show that using this method improves the robustness of the localization and path-following steps.

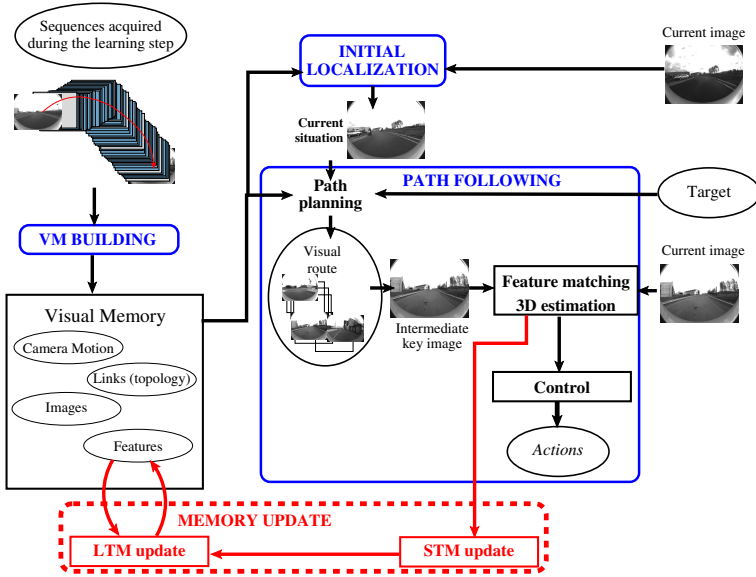
**Keywords:** Visual memory, life-long mapping, mobile robots.

## 1 Introduction

Visual memory-based navigation approaches have gained increasing interest in the last few years. They consist of representing the mobile robot environment with visual features gathered in a database. Basically, the navigation process from this visual memory can be split in three stages: 1) visual-memory acquisition, 2) initial localization and 3) path following (refer to Figure [1](#)). In the sequel, we will focus on strategies where key images are stored and used as references during the on-line steps whether with metrical or topological map. The initial localization is usually based on features matching to find the visual image of the memory the most similar to the current image. During the path following step, features matched between the current image and images of the visual memory can be exploited to provide the necessary input for the state estimation supplying the control law (refer to [2](#) for instance).

Most navigation strategies proposed in the literature assume that the environment where the robot works is static. However, this assumption does not hold for many real environments. Following the taxonomy proposed in [2](#), changes in the environment may be *transient* or *lasting*. Lasting *perceptual* changes will deteriorate the feature matching process and then the performance of vision-based navigation strategies. To improve the performances of the localization and of the

path following steps, it seems promising to modify the reference views as proposed in [34] using an information model based on the human memory model [5]. Basically, reference views are stored in a Long Term Memory. When features have been seen in many views during the localization step, they are transferred from the Short Term Memory to the long-term memory and missing features are forgotten. It is reported in [34] that localization performances are improved with respect to a static map.



**Fig. 1.** Navigation process from a visual memory. Visual-memory acquisition, initial localization and path following are the classical stages of such a process. Our contribution is the update of the memory to take into account lasting changes. The concepts of Short-Term Memory and Long-Term Memory are used in that aim.

This paper proposes a strategy to deal with the lasting changes within visual memory-based navigation frameworks using reference images. Each time the robot is realizing a navigation task, the content of its visual memory is updated to take into account relevant changes for future runs. The proposed approach well suits a large class of visual-memory based navigation strategies based on topological or metrical maps. Furthermore, it is not restricted to point features. The remaining of this paper is organized as follows. The problem is stated in Section 2. In Section 3, the principles of our approach are detailed. The Section 4 reports experimental results with a robot navigating in an indoor environment.

## 2 Problem Statement

We suppose that the environment contains a set of 3D features  $\{\mathcal{Q}_l \mid l = 1, 2, \dots, n\}$ . The observation (or projection) of a 3D feature  $\mathcal{Q}_l$  in an image  $\mathcal{I}^{i_a}$  is a visual feature noted  $\mathcal{P}_l^{i_a}$ . We assume that visual features can be located/detected from images and that they are described by feature vectors. Two features  $\mathcal{P}_{l_1}^{i_1}$  and  $\mathcal{P}_{l_2}^{i_2}$  from two images  $\mathcal{I}^{i_1}$  and  $\mathcal{I}^{i_2}$  are said to be *matched* or *in correspondence* if they are supposed to be the projections of a same 3D feature (i.e.  $l_1 = l_2$ ).

*Visual Memory.* The Visual Memory (VM) of the robot can store different features:

- a)  $n_{VM}$  key images  $\{\mathcal{I}^i \mid i = \{1, 2, \dots, n_{VM}\}\}$  extracted from the video sequence initially acquired,
- b) for each key image  $\mathcal{I}^i$ , a set  $P^i$  of  $n^i$  descriptive image features  $P^i = \{\mathcal{P}_{l_j}^i \mid j = \{1, 2, \dots, n^i\}, l_j \in \{1, 2, \dots, n\}\}$ ,
- c) a set of links between adjacent places  $\{(\mathcal{I}^{i_a}, \mathcal{I}^{i_b}), (i_a, i_b) \in \{1, 2, \dots, n_{VM}\}^2, i_a \neq i_b\}$ ,
- d) a set of 3D features  $\{\mathcal{Q}_l \mid l = 1, 2, \dots, n\}$ ,
- e) the motion of the camera between acquisitions of the key images.

*Initial Localization.* Let  $\mathcal{I}^c$  be the image acquired at the current time during the autonomous navigation stage. We suppose that the initial localization of the robot in its internal representation of the environment starts with finding the index  $i$  such that  $(\mathcal{I}^i, P^i)$  is the most similar to  $(\mathcal{I}^c, P^c)$ . A measure of similarity between images and/or features matching can be used in that aim.

*Path Following.* Once the robot is localized and a target is defined, we assume that it is then possible to extract a path from the current situation to the target. This path can be autonomously followed using a set of key images (called *visual route* in the sequel) defining successive references:  $\Psi = \{\mathcal{I}^{i_a} \mid a \in \{1, 2, \dots, n_\Psi\}, i_a \in \{1, 2, \dots, n_{VM}\}\}$ . Let  $\mathcal{I}^{i_a}$  be the current reference image of the visual route  $\Psi$ . A usual way to compute the state of the robot which supplies the control law can be summarized as follows. First, a set of image features  $P^c$  are extracted on the current image and matched with those of  $\mathcal{I}^{i_a}$  ( $P^{i_a}$ ). Then, 3D information are estimated and the state of the robot is extracted from the 3D information and the control law computed.

*Problem Statement.* To deal with perceptual lasting changes, it is necessary to update the content of the visual memory. According to Sections 2 and 2, visual features have to be fulfilled for the initial localization and during the path following stage. 3D features positions can also be required during path following. Our objective is thus to modify the set of features (visual features  $\{P^i\}$  and 3D features  $\{\mathcal{Q}_l\}$  if required) to improve the performance of the visual navigation process.

### 3 Map Update Process

To update the visual memory, we propose to use two memories: a Short-Term Memory (STM) and a Long-Term Memory (LTM) [5].

#### 3.1 Long-Term and Short-Term Memories

The Long-Term Memory stores the set of 3D features as well as 2D features. Each visual feature is associated with a state, representing its status at the current time. During the LTM update, states will be modified, resulting to the elimination of obsolete visual features.

The LTM is defined as:

$$LTM = \{ \{ \mathcal{Q}_l \mid l = 1, 2, \dots, n \} ; \{ (\mathcal{P}_{l_j}^i, s_{l_j}^i) \mid j = \{1, 2, \dots, n^i\}, \\ l_j \in \{1, 2, \dots, n\}, i = \{1, 2, \dots, n_{VM}\} \} \}$$

where  $s_{l_j}^i$  is the state of the visual feature  $\mathcal{P}_{l_j}^i$  (the projection of  $\mathcal{Q}_{l_j}$  in the image  $\mathcal{I}^i$  of the Visual Memory).

Let  $\mathcal{I}^{i_a}$  be the temporary reference image of the visual route. A Short-Term Memory (STM) is associated with this image. It stores 2D and 3D features extracted from the  $N$  images  $\{\mathcal{I}^{c_i} \mid 1 \leq i \leq N\}$  acquired during the path following step and such that their visual features have been matched with  $P^{i_a}$  for state estimation. More precisely, the *STM* contains three lists:

- $\mathbf{Q}^{mem}$  contains the  $n_{mem}$  3D features whose projections in  $\mathcal{I}^{i_a}$  have been matched with visual features of  $\{\mathcal{I}^{c_i}\}$ :

$$\mathbf{Q}^{mem} = \{ \mathcal{Q}_{l_j} \mid j = \{1, 2, \dots, n_{mem}\}, 1 \leq l_j \leq n \}$$

- $\mathbf{T}^{new}$  lists the  $N$  3D transformation matrices  $\mathbf{T}^{new} = \{ {}^{i_a}\mathbf{T}_{c_1}, {}^{i_a}\mathbf{T}_{c_2}, \dots, {}^{i_a}\mathbf{T}_{c_N} \}$  which give the pose of the  $N$  cameras' frames in the camera frame corresponding to the acquisition of  $\mathcal{I}^{i_a}$ . These transformations can be extracted from the 3D reconstruction estimated during path following.
- $\mathbf{Q}^{new}$  corresponds to *potentially new* features. A feature of  $\mathbf{Q}^{new}$  is projected on one image of  $\{\mathcal{I}^{c_i}\}$  at least but not in the reference image  $\mathcal{I}^{i_a}$ . The set  $\mathbf{Q}^{new}$  is noted:

$$\mathbf{Q}^{new} = \begin{cases} (\mathcal{Q}_1^{new}, \{ \mathcal{P}_1^{c_{1,1}}, \mathcal{P}_1^{c_{1,2}}, \dots, \mathcal{P}_1^{c_{1,n_1}} \}) \\ (\mathcal{Q}_2^{new}, \{ \mathcal{P}_2^{c_{2,1}}, \mathcal{P}_2^{c_{2,2}}, \dots, \mathcal{P}_2^{c_{2,n_2}} \}) \\ \dots \\ (\mathcal{Q}_n^{new}, \{ \mathcal{P}_n^{c_{n,1}}, \mathcal{P}_n^{c_{n,2}}, \dots, \mathcal{P}_n^{c_{n,n_n}} \}) \end{cases}$$

where  $\mathcal{P}_j^{l_i,j}$  is the projection of the new feature  $\mathbf{Q}_j^{new}$  in the image  $\mathcal{I}^{c_{l_i,j}}$  and  $n_j$  is the number of images where this 3D feature is projected.

### 3.2 Overview of the Update Process

The features of the LTM associated to the key images of the visual route  $\Psi$  are updated during the path following stage. The first key image of  $\Psi$  ( $\mathcal{I}^{i_1}$ ) is initially considered as the reference image  $\mathcal{I}^{i_a}$ . An empty STM is associated to this key image. Features are detected in the current image  $\mathcal{I}^{c_i}$  acquired by the embedded sensor and the STM is updated (refer to Section 3.3). Once the reference image  $\mathcal{I}^{i_a}$  is reached, the features of the Long Term Memory associated to this key image is updated using the content of the STM. Obsolete 3D features are deleted and new features are added and can be used in future runs for initial localization and state estimation. The LTM update process is detailed in Section 3.4. After this update, the STM associated to the key image is deleted and the update process is repeated for the next reference image  $\mathcal{I}^{i_a}$  until the end of the visual route is reached.

### 3.3 STM Update

$Q^{mem}$  is updated using 3D features whose projections are in the current image  $\mathcal{I}^{c_i}$  and in the reference image  $\mathcal{I}^{i_a}$ . 3D features not still present in  $Q^{mem}$  are added to this set. Each visual feature  $\mathcal{P}_r^{c_i}$  not matched with any visual feature of  $P^{i_a}$  is supposed to be the projection of a potentially new 3D feature. If  $\mathcal{P}_r^{c_i}$  is matched with the projection of a feature  $Q_j^{new}$  of  $Q^{new}$  (i.e. matched with a visual feature of the set  $\{\mathcal{P}_j^{c_{i,j}}\}$ ), then  $\mathcal{P}_r^{c_i}$  is added to the list of features associated to  $Q_j^{new}$ . Otherwise, a potentially new feature is added to  $Q^{new}$  and  $\mathcal{P}_r^{c_i}$  is associated to it.

### 3.4 LTM Update

The LTM is updated when the reference image  $\mathcal{I}^{i_a}$  is reached. This process can be split into two steps: feature update and feature addition.

*Feature Update.* The elements of  $Q^{mem}$  are used to update the state of features of the LTM. When a 3D feature  $Q_{l_j}$  belongs to  $Q^{mem}$  then the state of the corresponding projection  $\mathcal{P}_{l_j}$  is reset to the first state otherwise the state progresses (refer to Figure 2). A visual feature that passes through all the states without a “hit” is forgotten i.e. eliminated from the LTM. If any projection of the 3D feature  $Q_{l_j}$  exists, then  $Q_{l_j}$  is eliminated.

*Feature Addition.* A feature  $Q_i^{new}$  of the set  $Q^{new}$  is added to the LTM if  $n_i$  is upper than a threshold. As the number of images  $N$  acquired by the embedded camera and matched with a given key image may vary (depending on the acquisition framerate and on the longitudinal velocity of the robot among others), the threshold we propose is function of  $N$ :  $N\tau_{tf}$  with  $0 < \tau_{tf} \leq 1$ . It is supposed that this feature is projected on a sufficient number of images such that its 3D position can be computed in the frame attached to  $\mathcal{I}^{i_a}$  using the set  $T^{new}$  and the positions of its projections. The geometry of  $\mathcal{P}_i^{new}$  in the reference image  $\mathcal{I}^{i_a}$  can then be obtained by projecting this 3D feature. If the feature is not visible (i.e. it is outside the image), this feature is not added to the LTM.

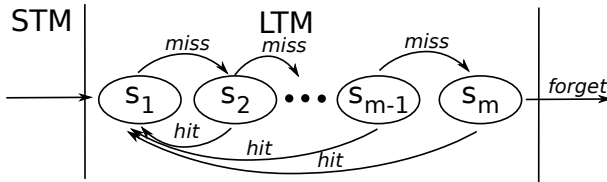


Fig. 2. LTM update, represented as a finite state machine

## 4 Implementation and Experiments

We use the complete framework for autonomous vehicle navigation proposed in [6]. The robot’s internal representation of the environment is a visual memory topologically organized. When running autonomously, the vehicle is guided along the reference visual route with a vision-based control law adapted to the nonholonomic constraint. The estimation of the input of the control law are computed from the camera motion parameters estimated between the current image and the reference image using point as visual features and a local 3D reconstruction.

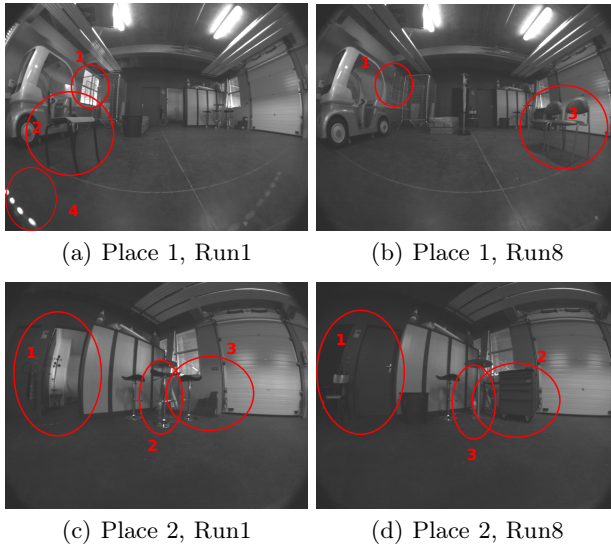
### 4.1 Set-Up and Implementation

Our experimental vehicle is a Pioneer AT3 robot equipped with a calibrated camera looking forward. Grey level images of resolution  $800 \times 600$  pixels are acquired at a rate of 7.5 fps. Vision and guidance algorithms are implemented in  $C^{++}$  language on a laptop using RTAI-Linux OS with a 2GHz Centrino processor. The considered visual features are Harris corners described by their neighborhood. The similarity score between two patches is based on the Zero Normalized Cross Correlation (ZNCC). The initial localization of the robot is the image of the visual memory with the highest number of features matched with the features  $P^c$  of the current image  $\mathcal{I}^c$ . During path following, features of the current image  $\mathcal{I}^c$  are matched with features of the reference image  $\mathcal{I}^{i_a}$  (*matched features*). The camera motion parameters are then determined from the essential matrix between  $\mathcal{I}^c$  and  $\mathcal{I}^{i_a}$  estimated using five couples of matched points and a RANSAC process. Once the robot reaches the reference image, the next image of the visual route is set as the reference image.

During the learning step, the robot is manually teleoperated in an warehouse-like indoor environment. The LTM built with the 22 selected key images contains 3039 3D points and 73132 visual features. During the navigation task, the robot is approximately situated where the initial image was acquired and the target is the last key image. The translational velocity is  $V = 80mm/s$ . A navigation task is first realized with a static visual memory. Images are stored and used to compare initial localization results with static and adaptive visual memories. A navigation task is then realized with the adaptive LTM. The environment is quiet similar during both navigation tasks. Later, modifications have been done manually and the same experimentations are repeated 8 times. Some changes

in appearance (objects or illumination) for two places at Run1 and Run8 are drawn in Figure 3.

For the memory update, the number of states of the LTM is set to  $m = 3$ . The threshold for feature addition is  $\tau_{tf} = 0.33$ . The 3D position of a new feature is computed using a triangulation process with 2 views and a RANSAC technique. The descriptor of its reprojection is set to be similar to the descriptor of the feature with the smallest reprojection error.



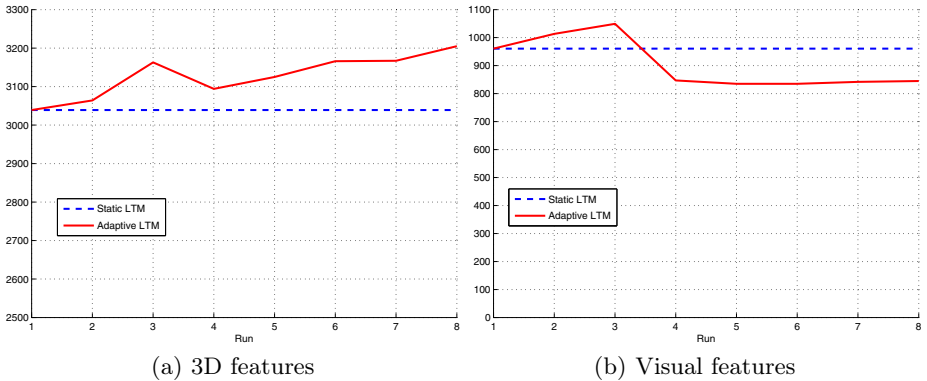
**Fig. 3.** Changes in appearance between Run1 and Run8 for two different places of the environment

## 4.2 Memory Content

The number of 3D features and the mean number of point features by key image are represented in Figure 4. The number of 3D features is clearly increasing over time using an adaptive memory. Between Run3 and Run4, the number of image features highly decreases. This is mainly due to the elimination of features detected in the images acquired during the learning stage but never used during the navigation tasks. The number of features then slowly varies, depending on the content of the environment. We observed that using an adaptive visual memory reduces the required resources in terms of memory.

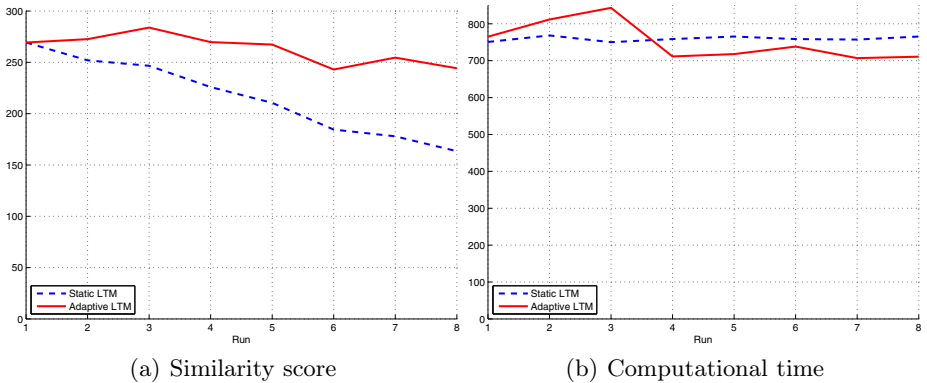
## 4.3 Initial Localization

The mean number of features matched between the current image and the most similar key image of the memory during the localization step is represented in Figure 5(a) while the mean computational time to process it is given in Figure 5



**Fig. 4.** Content of the LTM: (a) number of 3D features and (b) mean number of image features by key image versus run number

(b). The number of matched features is decreasing over time for both approaches but it decreases slower when using an adaptive memory. For Run8 for instance, 33% more features are matched when using adaptive memory. Moreover, after Run3, the computational cost is smaller when using an adaptive memory (7% of amount of time saved for Run8).



**Fig. 5.** Initial localization: (a) mean number of matched features and (b) mean computational time (in ms) versus run number

#### 4.4 Autonomous Navigation

For each run, the navigation tasks have been successfully executed using static and adaptive memories. Around 900 points are detected in each image. The mean number of features matched between the current image and the reference image as well as the number of features robustly matched are represented in Figure



6 It can be noticed that the percentage of robust matches over visual matches is decreasing (52% for Run1, under 40% for Run8). However, this percentage is higher when using an adaptive memory rather than a static memory (38% vs 26% for Run8). We are currently testing a L2-optimal triangulation technique for three views for the position of the feature in the reference image estimation in order to improve the gain in term of number of robust matches.

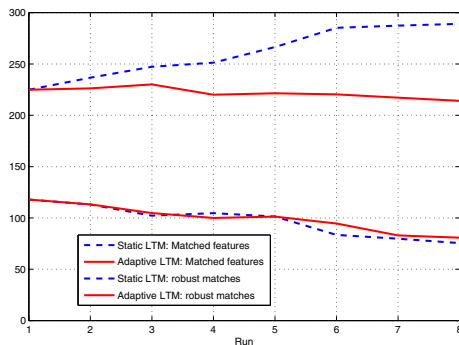


Fig. 6. Path following stage: mean number of matched features versus run number

## 5 Conclusion

We have presented a strategy to deal with perceptual lasting changes within visual memory-based navigation frameworks. This strategy is based on the concepts of short-term and long-term memories. During the path-following stage, visual features extracted in the current image is used to update a short-term memory. Once a key image is reached, features associated to it in the long-term memory are updated from the STM content. The results show that the performances of the initial localization and of the path following stages are improved. Current works deals with the inclusion of uncertainty propagation within the update process.

**Acknowledgment.** This work has been supported by the National Research Agency (ANR) of France (under the projects CityVip and R-Discover) and a grant (code I09200) from Gyeonggi Technology Development Program funded by Gyeonggi Province of Korea.

## References

1. Royer, E., Lhuillier, M., Dhome, M., Lavest, J.-M.: Monocular vision for mobile robot localization and autonomous navigation. *International Journal of Computer Vision, Special Joint Issue on Vision and Robotics* 74, 237–260 (2007)

2. Yamauchi, B., Langley, P.: Spatial learning for navigation in dynamic environments. *IEEE Transactions on Systems, Man and Cybernetics* 26(3), 496–505 (1997)
3. Bacca, B., Salvi, J., Batlle, J., Cufi, X.: Appearance-based mapping and localization using feature stability histograms. *Electronic Letters* 46(16), 1120–1121 (2010)
4. Dayoub, F., Cielniak, G., Duckett, T.: Long-term experiments with an adaptive spherical view representation for navigation in changing environments. *Robotics and Autonomous Systems* 59(5), 285–295 (2011)
5. Atkinson, R.C., Shiffrin, R.M.: Human memory: a proposed system and its control processes. In: Spence, K.W., Spence, J.T. (eds.) *The Psychology of Learning and Motivation*. Academic Press, New York (1968)
6. Courbon, J., Mezouar, Y., Martinet, P.: Autonomous navigation of vehicles from a visual memory using a generic camera model. *Intelligent Transport System (ITS)* 10, 392–402 (2009)

# A Software Architecture for RGB-D People Tracking Based on ROS Framework for a Mobile Robot

Matteo Munaro, Filippo Basso, Stefano Michieletto,  
Enrico Pagello, and Emanuele Menegatti

The authors are with the Intelligent Autonomous Systems Laboratory (IAS-Lab),  
at the Department of Information Engineering of the University of Padova,  
via Gradenigo 6B, 35131 - Padova, Italy.

{munaro,bassofil,michieletto,epv,emg}@dei.unipd.it

**Abstract.** This paper describes the software architecture of a distributed multi-people tracking algorithm for mobile platforms equipped with a RGB-D sensor. Our approach features an efficient point cloud depth-based clustering, an HOG-like classification to robustly initialize a person tracking and a person classifier with online learning to drive data association. We explain in details how ROS functionalities and tools play an important role in the possibility of the software to be real time, distributed and easy to configure and debug.

Tests are presented on a challenging real-world indoor environment and tracking results have been evaluated with the CLEAR MOT metrics. Our algorithm proved to correctly track 96% of people with very limited ID switches and few false positives, with an average frame rate above 20 fps. We also test and discuss its applicability to robot-people following tasks and we report experiments on a public RGB-D dataset proving that our software can be distributed in order to increase the framerate and decrease the data exchange when multiple sensors are used.

**Keywords:** People tracking, Robot Operating System, real-time, RGB-D data, mobile robots.

## 1 Introduction and Related Work

Autonomous service robots have to move and act in dynamic and populated environments, thus they must be endowed with fast and reliable perception skills. In particular, they must be able to distinguish people from the rest, predict their future positions and plan their motion in a human-aware fashion, according to their tasks.

People detection and tracking techniques have been widely studied in literature. Many works exist that use RGB cameras only ([8], [25], [5]) or 3D sensors only ([16], [23], [24], [6], [18]). However, when dealing with mobile robots, the need for robustness and real time capabilities usually led researchers to tackle these problems by combining appearance and depth information. In [3], both a

PTZ camera and a laser range finder are used in order to combine the observations coming from a face detector and a leg detector, while in [14] the authors propose a probabilistic aggregation scheme for fusing data coming from an omnidirectional camera, a laser range finder and a sonar system. Ess *et al.* [9], [10] describe a tracking-by-detection approach based on a multi-hypothesis framework for tracking multiple people in busy environments from data coming by a synchronized camera pair. The depth estimation provided by the stereo pair allowed them to reach good results in challenging scenarios, but their method relied on a people detection algorithm which took 30s to process each image. Stereo cameras continue to be widely used in the robotics community ([1], [21]), but the computations needed for creating the disparity map always impose limitations to the maximum frame rate achievable, thus leaving less room for further algorithms operating in series with the tracking one.



**Fig. 1.** Tracking output on some frames extracted from the test set

With the advent of reliable and affordable RGB-D sensors a rapid boosting of robots capabilities can be envisioned. For example, the Microsoft Kinect sensor allows to natively capture RGB and depth information at good resolution (640x480 pixels) and frame rate (30 frames per second). Even though the depth estimation becomes very poor over eight meters of distance and this technology cannot be used outdoors because the sunlight can change the infrared pattern projected by the sensor, it constitutes a very rich source of information that can be simply used on a mobile platform.

In [22] a people detection algorithm for RGB-D data is proposed, which exploits a combination of HOG and HOD descriptors. However, the whole frame is densely scanned to search for people, thus requiring a GPU implementation for being executed in real time. Also [7] relies on a dense GPU-based object detection, while [15] investigates how the usage of the people detector can be reduced using a depth-based ROI tracking. However, the obtained ROIs are again densely scanned by a GPU-based people detector. In [13] a tracking algorithm on RGB-D data is proposed, which exploits the multi-cue people detection approach described in [22]. It adopts an on-line detector that learns individual target models and a multi-hypothesis decisional framework. No information is given about the computational time needed by the algorithm and results are reported for some sequences acquired from a static platform equipped with three RGB-D sensors.

In our previous works ([2], [17]) a fast and robust algorithm for people detection and tracking has been proposed, that has been designed for mobile robots and reaches real time performance while relying on CPU-only computation. We reached state of the art tracking results on challenging scenarios in terms of accuracy and speed with a ROS-based implementation.

## 1.1 Robot Operating System

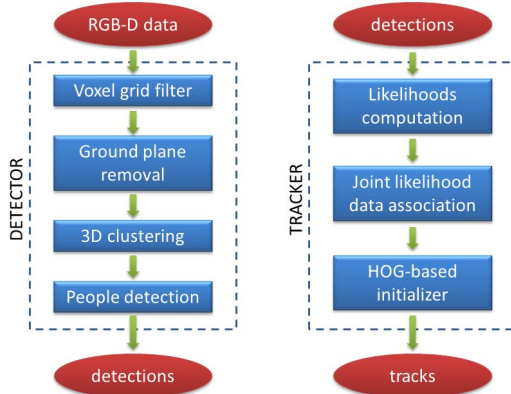
The *Robot Operating System* [19] by Willow Garage is an open-source, meta-operating system that provides services usually expected from an operating system, including hardware abstraction, low-level device control, message-passing between processes, package management, and tools and libraries useful for typical robotics applications, such as navigation, motion planning, image and 3D data processing. Among the many drivers provided with ROS there is also the OpenNI driver, which allows to interface with three RGB-D devices (the Microsoft Kinect, the Asus Xtion and the PrimeSense sensor) for obtaining their raw data. The primary goal of ROS is to support code reuse in robotics research and development and, in this direction, is designed to be as thin as possible and its libraries are ROS-agnostic and have clean functional interfaces.

In this paper, we describe the software architecture of a multi-people tracking and following algorithm based on RGB-D data and specifically designed for mobile platforms. In particular, we focus on how ROS has been used for development, optimization and debugging, providing also quantitative evaluation of our implementation choices. Moreover, we report tests to prove such a system to work also for tracking people with multiple sensors in a distributed configuration.

The remainder of the paper is organized as follows: in Section 2 an overview of the people tracking algorithm we implemented is given, while in 3 we detail how ROS tools and functionalities have been exploited for this purpose. Section 4 reports experimental results obtained by our method when tracking and following people from a mobile robot and when used in a multi-sensor and distributed people tracking scenario. In Section 5 we report conclusions and future works.

## 2 People Tracking Algorithm

In this section, we give a brief overview of the two main conceptual blocks of our people tracking system, whose single algorithms are described in [17]. As reported in Fig. 2, the RGB-D data acquired by the sensor are processed by a detection module that filters the point cloud data, removes the ground and performs a 3D clustering of the remaining points. The clustering method is composed by an *Euclidean Clustering* algorithm followed by a sub-clustering pipeline specifically designed for detecting persons even when close to each other or to the background. Furthermore, we apply a HOG-based people detection algorithm to the RGB image of the resulting clusters in order to keep only those that are more likely to belong to the class of people. The resulting output is a set of detections that are then passed to the tracking module.



**Fig. 2.** Block diagram describing input/output data and the main operations performed by our detection and tracking modules

Our tracking algorithm performs detection-track association as a maximization of a joint likelihood composed by three terms: motion, color appearance and people detection confidence. A person classifier for every target is learned online for evaluating color appearance. It is based on the *Online Adaboost* algorithm [12] and on features extracted from the color histogram of the target. At the learning stage, new detections associated to a track are considered as positive examples for that track’s classifier, while the other detections inside the image are chosen as negative examples. The HOG confidence is also used for robustly initializing new tracks when no association with existing tracks is found.

### 3 Software Architecture

This section details our algorithm by focusing on how ROS tools and functionalities have been used for the architectural design of our system, its implementation, debug and testing.

#### 3.1 Modularity with Nodes, Topics and Message-Passing

ROS nodes are processes that perform computation and communicate with each other by asynchronous message passing. Messages are exchanged within a network by means of TCP or UDP and written/read to/from topics, that are named buses over which nodes exchange well defined types of messages.

ROS nodes structure highly incentives software modularity, so that most of the code can be reused also for other applications. In particular, sensors drivers are usually confined into single nodes, while data processing is implemented within other nodes. Following this policy we structured our code as composed

by five different nodes<sup>1</sup> interconnected as shown in Fig. 3. This figure is very similar to what can be obtained at runtime with the ROS tool `rxgraph`, but it has been re-drawn for better visualization with colors. Nodes are reported in green, topics in cyan and sensors in orange. This tool allows to easily understand the code structure and also to find bugs related to missed/wrong connections of nodes and topics. Here below, a description of the single nodes is reported:

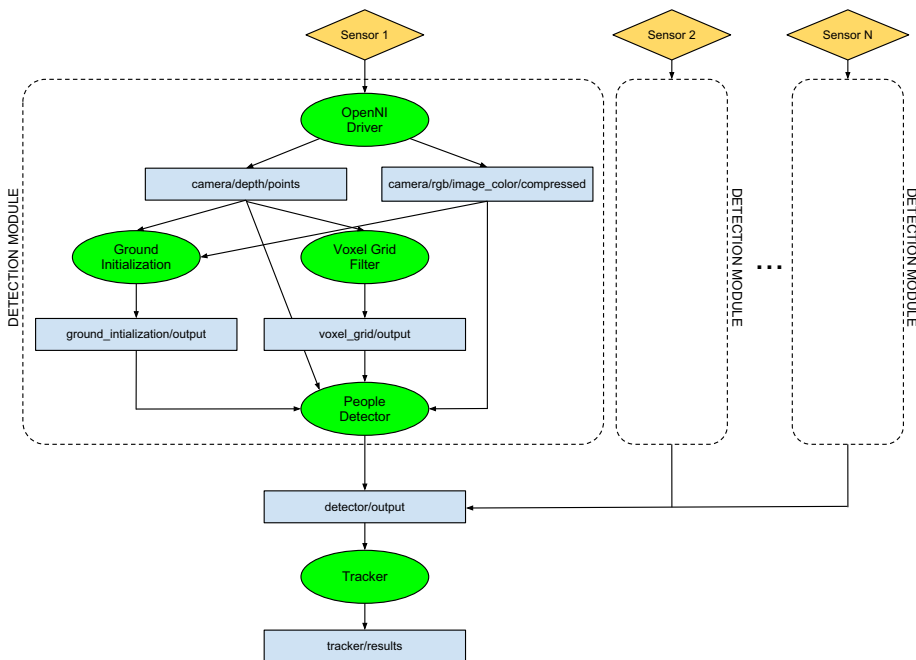


Fig. 3. Block scheme of nodes (green) and topics (cyan) interconnections

- **OpenNI\_driver**: this node is provided by ROS repositories<sup>2</sup> and consists in the open source driver for the Microsoft Kinect, the ASUS Xtion and the PrimeSense sensor. It publishes their raw data both as colored/not colored point clouds and as RGB and depth images.
- **ground\_initialization**: it is used for manual initialization of the ground plane at system startup. It takes as input sensor raw data and displays the RGB image, then a user is requested to click on three points referring to the ground. Given this input, it computes the ground plane parameters, that are then written to the corresponding topic. It is worth to notice that this node

<sup>1</sup> The second, fourth and fifth nodes have been implemented by the authors, while the first and the third nodes come as part of ROS libraries.

<sup>2</sup> [http://www.ros.org/wiki/openni\\_kinect](http://www.ros.org/wiki/openni_kinect).

is executed only once, thus it is usefully structured as a separated node that terminates after the ground plane has been initialized.

- `voxel_grid_filter`: this node performs a smart downsampling of the raw point cloud and it is implemented in the Point Cloud Library<sup>3</sup> [20], a 3D data processing library integrated within ROS. At each frame, the space is subdivided into a set of voxels (volumetric pixels) and all points inside each voxel are approximated with the coordinates of their centroid. Other than for reducing the number of points, this operation is also useful for obtaining point clouds with approximately constant density, where points density no longer depends on their distances from the sensor. In that condition the number of points of a cluster is directly related to its real size. As an example, in Fig. 4 we compare the raw point cloud of the Microsoft Kinect sensor with the result of the voxel grid filtering when choosing the voxel size to be of 0.06m.

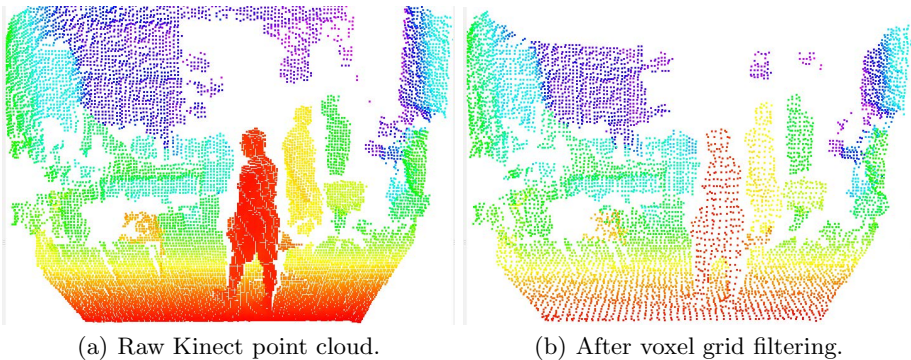


Fig. 4. The effect of the voxel grid filter on Kinect 3D data

- `people_detector`: this node takes as input the raw sensor data and the output of the `voxel_grid_filter` node. It removes the ground plane from the latter point cloud and performs an efficient 3D clustering optimized for people detection that allows to obtain clusters candidate to belong to people. These candidates are then validated with a HOG people detector and good clusters properties are published to a topic as the output of this node. On the same output topic (in this case named `/detector/output`) more nodes of `people_detector` type can publish. That is the case illustrated in Fig. 3, where many RGB-D sensors are present and, for every sensor within the network, a detection block is run that publishes the corresponding detections to the common topic. This node exploits also the raw sensor data to extract the highest possible number of points from the clusters belonging to

<sup>3</sup> <http://www.pointclouds.org>.



people. The RGB values corresponding to these points are then written to the output topic in order to be used by the `tracker` node to compute color histograms of the targets.

- `tracker`: it has the task to perform prediction and data association by having as input the detections published to the topic `/detector/output`. As stated above, these detections can come from different sensors connected to different computers and they are all used for performing people tracking.

The main part of the algorithms performed by the described nodes is contained in callback functions, executed every time a new message is published to their corresponding input topic. In Table 1 we report the bandwidth occupied by publishing to the main topics reported in Fig. 3. These data are very useful since they highlight which messages are too heavy and could slow down the whole application because they have to be copied between two different memory locations. They have been obtained with the ROS command `rostopic bw topic_name`. For the topic relative to the raw XYZ point cloud (`camera/depth/points`) we report the measured bandwidth at three different depth resolutions, that can be set by means of the OpenNI driver. The requested bandwidth for topic `detector/output` is quite high because the raw RGB image is also passed from the detection module to the tracker. However, it could be easily decreased by an order of magnitude by transmitting the RGB image in compressed format by means of the `image_transport` ROS package.

**Table 1.** Bandwidth used (in MB/s) when publishing messages to the listed topics

Topic name	Bandwidth (MB/s)
camera/depth/points (VGA)	148
camera/depth/points (QVGA)	37
camera/depth/points (QQVGA)	9.3
camera/rgb/image_color/compressed	1.1
voxel_grid/output	8.1
detector/output	30

### Nodelets for Avoiding Data Copying

Every node in ROS runs in a different process, thus interprocess communication is needed in order to exchange data between nodes. That means that the computer has to spend more time and memory for passing data between them. For such a reason, as an alternative to nodes, nodelets have been designed to provide a way to run multiple algorithms in the same process with zero copy transport between algorithms.

We implemented a nodelet version of our detection module where we substituted the nodes with nodelets managed by the same nodelet manager. In Table 2 we report the framerate of the people detector for our two different implementations and for three different resolutions of the sensor depth point cloud. As it

**Table 2.** Comparison between nodes and nodelets versions of our people detection software in terms of framerate (fps) and at different depth resolutions

	VGA	QVGA	QQVGA
<b>nodes</b>	14.5	21.7	28.1
<b>nodelets</b>	18.3	24.8	29.9

can be noticed, the nodelets implementation led to a higher framerate and the higher is the resolution, the higher is the framerate gain because we avoid to copy large amounts of data between different processes<sup>4</sup>.

### 3.2 Easy Configuration with `tf`

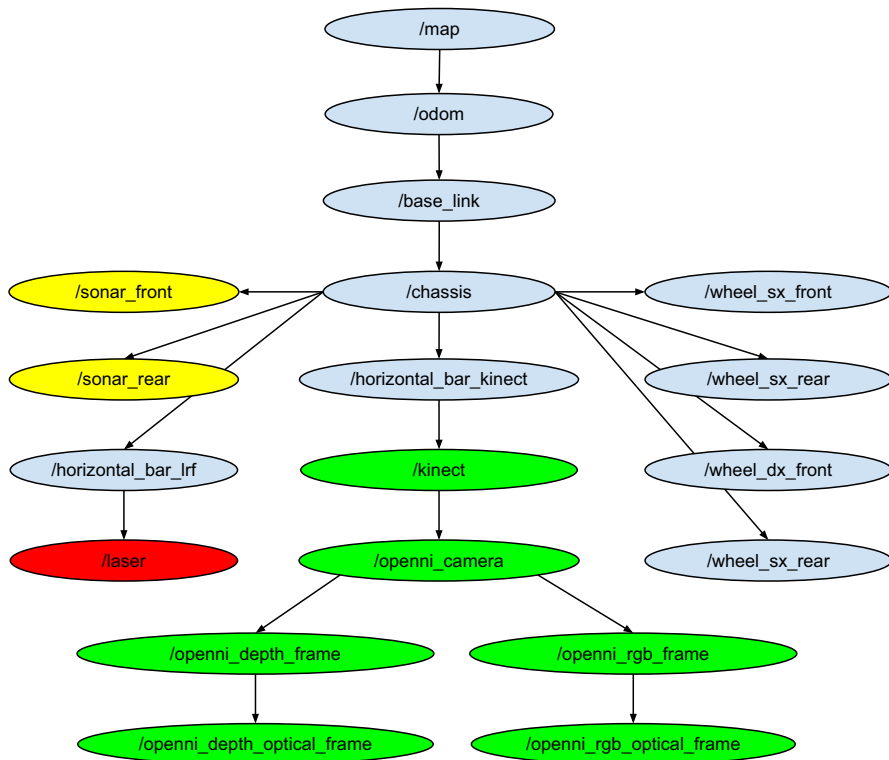
Some of the most frequent errors when dealing with robots with multiple joints or with multiple sensors is related to wrong reference system transformations. ROS `tf` package has been designed to easily refer data to multiple reference systems, that can vary over time, and maintains the relationship between coordinate frames in a tree structure buffered in time. This package is distributed, in the sense that there is no central source of information and all the coordinate frames are known to all ROS components on any computer in the system. Moreover, there is no loss in accuracy when transforming data multiple times. These transforms can be published or listened to with a fixed rate over time and they can be visualized with both the ROS visualizer (`rviz`), that shows their position in real time, and with the command `roslaunch tf view_frames`, that saves to a file the full tree of coordinate transforms. In Fig. 5, we report the `tf` tree showing the parent/child relationship of the coordinate frames related to our robotic platform. `tf` relative to the world map and physical links of the robot are colored in cyan, while the other colors highlight the frames referring to the different sensors mounted on the robot, namely sonars, laser range finder and Kinect. In Fig. 6 we report a photo of the robotic platform (a) we used for the people following application, together with its URDF model as it is visualized in `rviz` (b). In addition, also `tf` reference axes are drawn for every robot joint and every sensor (c). This visualization allowed to easily check the correctness of the reference frames. Moreover, `rviz` makes possible visualization with hardware in the loop, so that a direct comparison between the simulated and the real behavior of the robot can be performed.

## 4 Experiments

### 4.1 Tests with a Mobile Robot

We present here some results obtained with our tracking system on RGB-D video sequences collected in an indoor environment with the mobile robot shown in Fig. 6. It consists of a Pioneer P3-AT platform equipped with a Microsoft Kinect

<sup>4</sup> All the other tests reported in this paper refer to the implementation based on nodes.



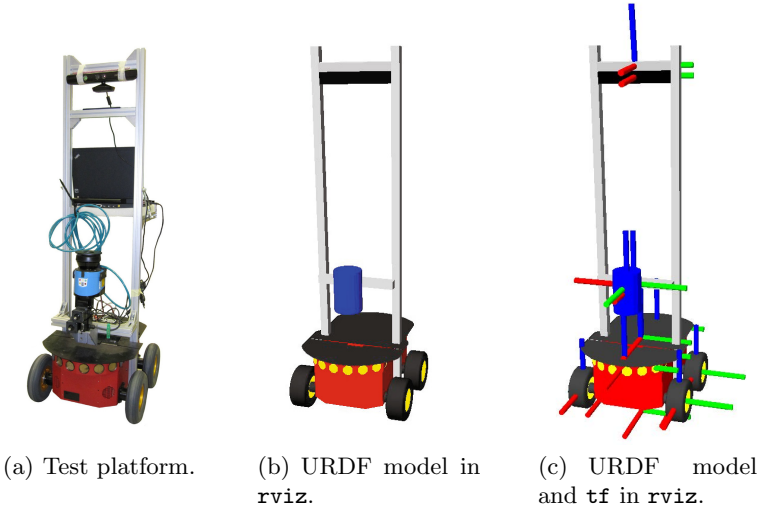
**Fig. 5.** Transforms (tf) tree representing the reference systems defined for our robotic platform and their parent-child relationships

sensor, which is endowed with a standard RGB camera, an infrared camera and an infrared projector. This low cost hardware can provide RGB-D data with  $640 \times 480$  pixel resolution at 30 frames per second. In these tests we acquired depth data at QQVGA resolution, namely  $160 \times 120$  pixels, for speed. It is worth to notice that this choice does not affect the accuracy achievable by our system, because of the voxel dimension we chose for the voxel grid filter we apply.

We performed tests while the robot was moving along one direction in three different scenarios of increasing difficulty:

1. no obstacle is present, people move with simple (linear) trajectories;
2. no obstacle is present, people move with complex trajectories and interact with each other;
3. obstacles (chairs, a whiteboard) are present, people move with complex trajectories and interact with each other.

Every video sequence extends over about 750 frames, thus the total test set includes 2371 frames, 6172 instances of people and 13 tracks that have been manually annotated on the RGB image and that constitute the ground truth.



**Fig. 6.** The robotic platform (a) used in the tests, its model displayed with the ROS visualizer (b) and the reference frames of every model joint (c)

The minimum distance between people is 0.2m while the minimum people-object distance is 0.05m.

For the purpose of evaluating the tracking performance we adopted the CLEAR MOT metrics [4], that consists of two indexes: MOTP and MOTA. The MOTP indicator measures how well exact positions of people are estimated, while the MOTA index gives an idea of the number of errors that are made by the tracking algorithm in terms of false negatives, false positives and mismatches. In particular, given that our ground truth does not consist of the metric positions of all persons, but of their positions inside the image, we computed the MOTP index as the average PASCAL index [11] (intersection over union of bounding boxes) of the associations between ground truth and tracker results by setting the validation threshold to 0.5. We computed the MOTA index with the following formula

$$MOTA = 1 - \frac{\sum_t (fn_t + fp_t + ID_t^{sw})}{\sum_t g_t} \quad (1)$$

where  $fn_t$  is the number of ground truth people instances (for every frame) not found by the tracker,  $fp_t$  is the number of output tracks instances that do not have correspondences with the ground truth,  $ID_t^{sw}$  represents the number of times a track corresponding to the same person changes ID over time and  $g_t$  is the total number of ground truth instances present in all frames.

In Table 3 we report, for every test sequence, the MOTP and MOTA indexes, the percentage of false positives and false negatives and the number of ID switches. The results are very good for these tests: the only ID switches are due to people who change motion direction when occluded by other people or

outside the camera field of view. In Fig. 4 we report some examples of correctly tracked frames from our test set. Different IDs are represented by different colors and the bounding box is drawn with a thick line if the algorithm estimates a person to be completely visible, while a thin line is used if a person is considered occluded.

**Table 3.** Tracking results for tests with a moving robot

	MOTP	MOTA	FP	FN	ID Sw.
Simple traj.	82.2%	95.8%	2.5%	1.6%	3
Complex traj.	83.5%	90.9%	4.7%	4.4%	1
With obstacles	83.3%	94.3%	4.7%	0.9%	3

#### 4.1.1 People Following Module

As a further test for proving the robustness and the real time capabilities of our tracking method we implemented an additional module with the task of following a specific tracked person. At the implementation level this is done with a ROS node which reads the tracking results and drives the robot towards the person with the specified ID. It is worth to notice that we wrote this new node in Python, while the rest of the software has been written in C++, by exploiting the property of ROS messages to be language and platform-independent. We then asked the robot to follow a particular person along a narrow corridor with many lighting changes and a hall where many people are present. In Fig. 5 the output of the tracking algorithm is visualized for some frames collected during this experiment. In the first row it can be seen that the person is correctly tracked and followed by the platform along the corridor, while the lighting conditions considerably change. This is allowed by a good prediction of people position in ground plane coordinates, obtained with the use of the Unscented Kalman Filter that can also benefit of many available measures thanks to the high frame rate. In the second row the tracking (and following) robustness is shown when other people walk next to the followed person or between him and the robot. As a real world example, we show in Fig. 6 some screenshots of our robot that successfully managed to detect and track people within groups and to follow a particular person within a crowded environment only by means of data provided by the tracking algorithm.

## 4.2 Tests with a Distributed Tracking System

When doing people tracking from multiple cameras, it is useful to distribute the computational cost to multiple agents/computers. The advantage is two-fold: there is no need for a powerful computer as central unit and only selected data have to be sent over the network. If a software has been developed using ROS, distributing the computation among different machines becomes an easy task



**Fig. 7.** People following test. First row: examples of tracked frames while a person is robustly followed along a narrow corridor with many lighting changes. Second row: other examples of correctly tracked frames when other people are present in the scene.



**Fig. 8.** Sample images of our mobile robot following the person with the blue sweater within a crowded environment. It is worth to notice that the sweater has the same color of the carpet on the floor.

**Table 4.** Computers of the wired network used for the distributed tests

	PC1	PC2	PC3	PC4
<b>CPU</b>	Xeon Quad 3.10GHz	i5-520M 2.40 GHz	i7-620M 2.67Ghz	Core 2 Quad 2.66Ghz
<b>RAM</b>	4GB DDR3	4GB DDR3	4GB DDR3	4GB DDR3
<b>Type</b>	Desktop	Laptop	Laptop	Desktop

because a node makes no assumption about where in the network it runs, thus computation can be relocated at run-time to match the available resources.

In order to test our people tracking algorithm in a distributed configuration we used the *RGB-D People Dataset*<sup>5</sup> ([22],[13]), which contains about 4500 RGB-D frames acquired from three vertically mounted Kinect sensors pointing towards adjacent, but not overlapping regions. As sketched in Fig. 3, we used three independent people detection modules (one for each sensor) that publish detection messages to the same topic and a tracking node that fuses them in tracks as they arrive. Detections coming from different sensors are referred to the same reference system by means of static `tf`. Data were pre-recorded in ROS `bag` files that contain Kinect depth in QQVGA resolution and RGB image in VGA resolution. We evaluated the tracking framerate of the whole process both when executed on a single computer and also when distributed on a heterogeneous computer network composed by PCs whose main specifications are reported in Table 4. PC1 and PC4 are desktop computers, while PC2 and PC3 are laptops suitable to be used on a mobile robot. As it can be seen in Table 5, the stream of a single Kinect of the dataset can be processed on PC1 at 28 fps by our detection module, while the whole detection and tracking algorithm runs at 26 fps. When processed on PC2, these framerates decrease at 23 and 19 fps respectively. When processing the whole *RGB-D People Dataset* (three detection modules and one tracking node) only on PC2, the framerate decreased at 15 fps only. Therefore, we tested a distributed configuration where the three detection modules have been run on PC2, PC3 and PC4 and the tracking node have been executed on PC1. All the computers were connected within a gigabit ethernet network. In this configuration, we measured a framerate of 58 fps as the sum of the frames processed by the three detection modules in the three computers and 31 fps for the tracker node in the fourth computer. As a result, we can notice the doubling of the tracking framerate. Moreover, it is worth to notice that the centralized processing of three Kinect streams has been possible because data were pre-recorded, while dealing with live acquisition in a centralized configuration could further decrease the tracking framerate or could not be possible due to bandwidth limitations of the USB bus on standard computers. A video with the qualitative results obtained with our distributed tracking system can be found at this link: <http://youtu.be/b70vLKFsr1M>, while in Table 6 a quantitative comparison of our system with that in [13] is reported. For further reference, please refer to [17].

<sup>5</sup> <http://www.informatik.uni-freiburg.de/~spinello/RGBD-dataset.html>

**Table 5.** Framerate of the detection and tracking modules with different test configurations

	Detector (fps)	Tracker (fps)
Single stream on PC1	28	26
Single stream on PC2	23	19
Three streams centralized on PC2	20	15
Three streams distributed	58	31

**Table 6.** Tracking evaluation with RGB-D People Dataset

	MOTP	MOTA	FP	FN	ID Sw.
Ours	73.7%	71.8%	7.7%	20.0%	19
[13]	N/A	78%	4.5%	16.8%	32

## 5 Conclusions and Future Works

In this paper we described the architectural design of a RGB-D people tracking software based on the ROS framework. We focused on how ROS tools and functionalities led to a modular code structure, easy to configure and to debug. Tests with a mobile robot were performed in scenarios of increasing complexity and our tracking system proved to be very effective even when dealing with strong occlusions and complex trajectories. Moreover, we proved we can reach real time performance and that our implementation can be distributed on multiple computers in order to increase the framerate and decrease the data exchange over the network with respect to a centralized computation.

As future works, we envision to decrease the bandwidth requested by the data exchange between the detection and tracking modules and to test this people tracking application when using data from multiple RGB-D sensors placed on each member of a team of autonomous robots that can communicate by means of a wireless network. Moreover, we are planning to release a new RGB-D dataset that could allow to measure the 3D accuracy obtainable when tracking people with consumer RGB-D sensors, such as Microsoft Kinect<sup>6</sup>.

## References

1. Bajracharya, M., Moghaddam, B., Howard, A., Brennan, S., Matthies, L.H.: A fast stereo-based system for detecting and tracking pedestrians from a moving vehicle. *International Journal of Robotics Research* 28, 1466–1485 (2009)
2. Basso, F., Munaro, M., Michieletto, S., Pagello, E., Menegatti, E.: Fast and robust multi-people tracking from RGB-D data for a mobile robot. In: Lee, S., Cho, H., Yoon, K.-J., Lee, J. (eds.) *Intelligent Autonomous Systems 12. AISC*, vol. 193, pp. 265–276. Springer, Heidelberg (2012)

<sup>6</sup> For further information, please visit

<http://www.dei.unipd.it/~munaro/KTP-dataset.html>



3. Bellotto, N., Hu, H.: Computationally efficient solutions for tracking people with a mobile robot: an experimental evaluation of bayesian filters. *Auton. Robots* 28, 425–438 (2010)
4. Bernardin, K., Stiefelhagen, R.: Evaluating multiple object tracking performance: the clear mot metrics. *J. Image Video Process.* 2008, 1:1–1:10 (2008)
5. Breitenstein, M.D., Reichlin, F., Leibe, B., Koller-Meier, E., Van Gool, L.: Robust tracking-by-detection using a detector confidence particle filter. In: *IEEE International Conference on Computer Vision* (October 2009)
6. Carballo, A., Ohya, A., Yuta, S.: People detection using range and intensity data from multi-layered laser range finders. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5849–5854 (2010)
7. Choi, W., Pantofaru, C., Savarese, S.: Detecting and tracking people using an rgb-d camera via multiple detector fusion. In: *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 1076–1083. IEEE (2011)
8. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: *Computer Vision and Pattern Recognition 2005*, vol. 1, pp. 886–893 (June 2005)
9. Ess, A., Leibe, B., Schindler, K., Van Gool, L.: A mobile vision system for robust multi-person tracking. In: *IEEE Conference on Computer Vision and Pattern Recognition 2008*, pp. 1–8 (2008)
10. Ess, A., Leibe, B., Schindler, K., Van Gool, L.: Moving obstacle detection in highly dynamic scenes. In: *Proceedings of the 2009 IEEE International Conference on Robotics and Automation, ICRA 2009, Piscataway, NJ, USA*, pp. 4451–4458 (2009)
11. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. *Int. J. Comput. Vision* 88, 303–338 (2010)
12. Grabner, H., Bischof, H.: On-line boosting and vision. In: *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Washington, DC, USA*, vol. 1, pp. 260–267 (2006)
13. Luber, M., Spinello, L., Arras, K.O.: People tracking in rgb-d data with on-line boosted target models. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2011* (2011)
14. Martin, C., Schaffernicht, E., Scheidig, A., Gross, H.-M.: Multi-modal sensor fusion using a probabilistic aggregation scheme for people detection and tracking. *Robotics and Autonomous Systems* 54(9), 721–728 (2006)
15. Mitzel, D., Leibe, B.: Real-time multi-person tracking with detector assisted structure propagation. In: *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*. IEEE (2011)
16. Mozos, O., Kurazume, R., Hasegawa, T.: Multi-part people detection using 2d range data. *International Journal of Social Robotics* 2, 31–40 (2010)
17. Munaro, M., Basso, F., Menegatti, E.: Tracking people within groups with rgb-d data. In: *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, Algarve, Portugal (October 2012)
18. Navarro-Serment, L.E., Mertz, C., Hebert, M.: Pedestrian detection and tracking using three-dimensional ladar data. In: *FSR*, pp. 103–112 (2009)
19. Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., Ng, A.: Ros: an open-source robot operating system. In: *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA* (2009)
20. Rusu, R.B., Cousins, S.: 3D is here: Point Cloud Library (PCL). In: *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 (2011)

21. Satake, J., Miura, J.: Robust stereo-based person detection and tracking for a person following robot. In: Workshop on People Detection and Tracking IEEE ICRA (2009)
22. Spinello, L., Arras, K.O.: People detection in rgb-d data. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2011 (2011)
23. Spinello, L., Arras, K.O., Triebel, R., Siegwart, R.: A layered approach to people detection in 3d range data. In: Proc. 24th AAAI Conference on Artificial Intelligence, PGAI Track (AAAI 2010), Atlanta, USA (2010)
24. Spinello, L., Lubner, M., Arras, K.O.: Tracking people in 3d using a bottom-up top-down people detector. In: IEEE International Conference on Robotics and Automation (ICRA 2011), Shanghai (2011)
25. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: CVPR 2001, vol. 1, pp. 511–518 (2001)

# Drivable Road Modeling Based on Multilayered LiDAR and Vision

Sukhan Lee<sup>1,2</sup> and Yongjin Park<sup>1</sup>

<sup>1</sup> Intelligent Systems Research Institute, Sungkyunkwan University

<sup>2</sup> Dept. of Interaction Science,

School of Information and Communication Eng

Suwon 440-746, Rep. of Korea

lsh@ece.skku.ac.kr, isrc2011@skku.edu

**Abstract.** This paper presents a real-time modeling of drivable road along with the vehicle movement by integrating a 4-layered Light Detection and Ranging device (LiDAR) and a vision camera. A drivable road is represented virtually as a set of articulated multi-plates linked by rotational joints. The road surface, curbs and obstacles identified by the multilayer LiDAR are fused with the lanes and crossings recognized by the vision camera to determine the multi-plate model of drivable road as well as the pose of vehicle and obstacles on the modeled road. The road model parameters as well as the vehicle and obstacle poses on the modeled road are subject to update in real-time using the encoder/IMU readings to compute the propagation while the fused LiDAR and vision to provide the measurements. The main contribution of this paper is an integrated framework of LiDAR and vision for the real-time modeling of a drivable road as well as for the identification of vehicle and obstacle poses on the modeled road. Experimental results are demonstrated at the end of this paper.

**Keywords:** Drivable road recognition, multilayer LiDAR, vision, data Fusion.

## 1 Introduction

The technology for unmanned ground vehicles (UGVs), especially autonomous navigation under the capability of self-identifying drivable roads, has emerged as being significant for the intelligent transportation system (ITS) of the future. The motivations behind the above emerging trend are as follows: First, there arises a need for a safer transportation means to cope with almost 95% of the accidents on roads that are known to be caused by human errors. A driving assistance to human drivers by intelligent systems or a complete elimination of human intervention in driving may enhance the overall safety on the roads.

Second, there is a need to reduce the traffic congestion on the roads as the existing roads become ineffective for accommodating the increasing number of vehicles introduced, although a considerable portion of the road space remained unused due to the inter-vehicle distance required to maintain the safety. Introducing autonomous vehicles on the roads may shorten considerably the inter-vehicle distance with faster

reaction time and platooning capability, resulting in a more efficient use of road space. Third, there is a need to implement a more environmentally friendly transportation system not only to cope with air pollution but also to slow down CO<sub>2</sub> accumulation. To do so, a direct and environmentally friendly link between a mass transportation and a personal transport has been proposed, where intelligent autonomous, environmentally friendly, vehicles are proposed for personal transport. Fourth, there is a need to upgrade personal transport service to be more pervasive, providing a means of pervasive personal transportation at such places as tourist attractions, campuses, airports, etc. where roads are poorly structured and/or infested by pedestrians. A series of such personal transport means have been introduced and demonstrated as a prototype or a commercial product since the introduction of Segway. Integrating the personal transport means with robotic functions, such as autonomous navigation as well as intelligent assistance to the driver, safer and more convenient personal services can be in place. Fifth, for those applications requiring a tele-robotic operation due to the difficulty or infeasibility for human to access, for instance, hazardous workplaces, disaster areas, battlefields, etc., supporting tele-operation with autonomy and intelligence can greatly extend its reliability and performance especially when there involves a time-delay in tele-operation.

Recently, some remarkable progresses are demonstrated to the world by the winners at DARPA grand challenge and Google car program. The Robotics Institute in the Carnegie Mellon University has successfully developed the Navlab which can drive not only on a paved road but also on a poorly structured road with the capabilities of driver assistance system, pedestrian detection and environment recognition [3-4]. The Google's driverless car has shown a remarkable performance especially under the setting of urban transportation. In 2009, Google obtained 3,500 miles of Street View images from driverless cars with minor human intervention. As of 2010, Google has tested several vehicles equipped with this system, driving approximately 1,609 km without any human intervention, in addition to 225,308 km with occasional human intervention. Google expects that the increased accuracy of its automated driving system could help reduce the number of traffic-related injuries and deaths, while using energy and space on roadways more efficiently [3]. Although it brought about a strong impression on the feasibility of deploying autonomous vehicles in the real street with its performance beyond expectation, to solve the problem of its high cost remains a challenge.

To achieve the desired performance and robustness with the sensors affordable yet limited in performance, there have been invested much efforts to develop and innovate viable solutions for such issues as drivable road recognition, 3D map generation and localization, static/dynamic obstacle tracking, and real time path generation and control [7-12]. Among those issues, recognition and modeling of a drivable road under poorly structured and pedestrian/obstacle infested road conditions is considered as one of the most critical yet challenging problems to solve for the successful implementation of autonomous navigation. A drivable road is defined here as an area that a vehicle can be safely driven to without collision. It may consider only a paved road or include an open cluttered area, depending on applications. It is important for the drivable road modeled to allow the platform to follow a safe track or trajectory with an obstacle avoidance strategy. The recognition represents a chain of procedures for detection, extraction and conversion from the raw data obtained by sensors to the

information on the road and vehicle that is significant for safe driving. Among the many visual sensors, an active laser range sensor has performance of fine resolution and accuracy in depth measurement, possibly in tens of millimeter scale, with a longer dynamic range and relatively faster speed. However, the sensor scans only on a 2D plane, unless it mechanically rotates the scan plane for 3D, resulting in limited information on the frontal scene. To overcome the limitation from a single layer of scanning, recently, LiDAR with multiple layers of planar scanning beams has been introduced [9-10]. On the other hand, a passive vision camera provides with a wide view of, densely pixelated, gray and/or color information by projecting a frontal scene onto an image plane. However, due to the passive nature of sensing, it also has its own problems for coping with the variation of light distributions and sensitiveness to weather induced road conditions. In [8], a monocular vision solution is provided for the concurrent recognition of multiple lanes, using the parametric transformation robust to shadows as well as invariant to color and texture of the road. Incorporating edges into color intensity, highly curved roads are recognized and followed, where the use of saturation channel in HSI representation and the estimation of a smooth multiple clothoid parameters using Kalman filter are presented [15]. Latest, a more attention has been given to the development of a multi-sensor system that works well with various road environments [10][12]. For instance, the integration of a laser range sensor and a vision camera is receiving a high attention for the robust recognition of a drivable road, as the problem of drivable road recognition becomes more dependable by fusing the two different data sets.

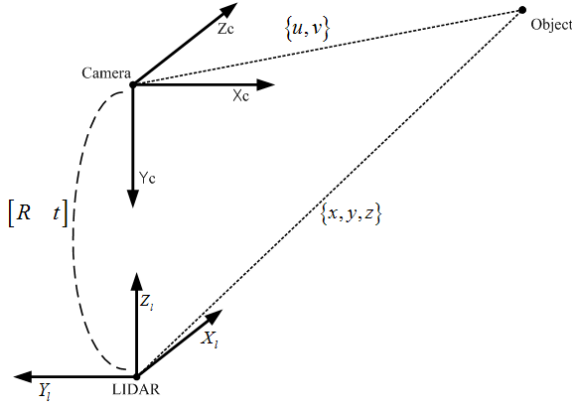
This paper presents a real-time modeling of drivable road along with the vehicle movement by integrating a 4-layered LiDAR and a vision camera. A drivable road is represented virtually as a set of articulated multi-plates linked by rotational joints. The road surface, curbs and obstacles identified by the multilayer LiDAR are fused with the lanes and crossings recognized by the vision camera to determine the multi-plate model of drivable road as well as the pose of vehicle and obstacles on the modeled road. The road model parameters as well as the vehicle and obstacle poses on the modeled road are subject to update in real-time using the encoder/IMU readings to compute the propagation while the fused LiDAR and vision to provide the measurements. The main contribution of this paper is an integrated framework of LiDAR and vision for the real-time modeling of a drivable road as well as for the identification of vehicle and obstacle poses on the modeled road. Experimental results are demonstrated at the end of this paper.

The rest of this paper is organized as follows: after the introduction in Section-1, Section-2 describes about sensor calibration. Road plate-model is presented in Section-3. The drivable road modeling by a vision camera and LiDAR are explained in Section-4 and Section-5, respectively. Section-6 covers sensor fusion with Section-7 dealing with obstacle detection. Experimental results are presented in Section-8 and finally conclusion is given in Section-9.

## **2 Calibration of a Camera and LiDAR Configuration**

To integrate camera and LiDAR, the parameters of both sensors and their geometric relationship need to be established. Note that the data acquired by the camera and the

LiDAR are represented in their own coordinate frames i.e data from the camera is expressed in terms of their respective image coordinates, (u,v) and that of LiDAR in Cartesian coordinates, (x, y, z). Fig.1 shows a schematic representation of the calibration geometry between a camera and LiDAR.

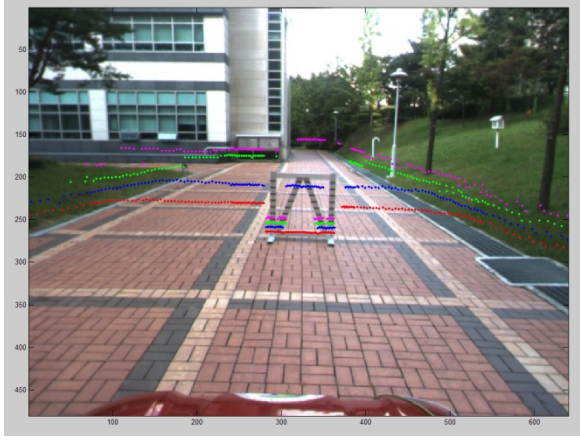


**Fig. 1.** A Schematic Representation of the Calibration Geometry between a Camera and LiDAR

We assume that the camera and LiDAR internal parameters are calibrated apriori, independently of the calibration of external parameters,  $[R, t]$ . For the calibration of  $[R, t]$ , we use the calibration block of known geometry that we designed especially for our purpose. As shown in the center part of the image of Fig. 2, the calibration block is designed on purpose in the shape of an isosceles triangle inside a rectangle, such that the LiDAR data from the calibration block can be analyzed geometrically in such a way as to determine their exact location on the block. Then, we can compute the camera pixel coordinate, (u,v), that corresponds to the location, (x,y,z), of the individual LiDAR data on the block, based on the image processing of the calibration block. Obtaining a collection of (u,v) and (x,y,z) correspondences from the LiDAR data on the block, we can calibrate the external parameter,  $[R, t]$ . Fig. 2 illustrates the camera image of the calibration block onto which the 3D data from the LiDAR are projected and overlaid. Note that the performance of the calibration depends on the size of the calibration block relative to the measurement errors included in LiDAR data.

The calibration method described above is formalized by the following equations:

$$\begin{aligned}
 x^c &= k [R \ t] \cdot x^l \\
 T &= [R \ t] \\
 k^{-1} x^c &= T \cdot x^l
 \end{aligned} \tag{1}$$



**Fig. 2.** Camera and LIDAR Calibration

where  $x^c$  represents the pixel coordinate in the image frame that corresponds to a LiDAR data point,  $x^l$ , and  $k$  represents the Camera projection matrix. Eq. (1) can be rewritten in the homogeneous coordinate frame as:

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} T_{11} & T_{12} & T_{13} & T_{14} \\ T_{21} & T_{22} & T_{23} & T_{24} \\ T_{31} & T_{32} & T_{33} & T_{34} \end{bmatrix} \cdot \begin{bmatrix} x^l \\ y^l \\ z^l \\ 1 \end{bmatrix} \quad (2)$$

Where  $w$  represents the scale parameter. Let us define  $u'$  and  $v'$  as

$$u' = \frac{u}{w} = \frac{T_{11}x^l + T_{12}y^l + T_{13}z^l + T_{14}}{T_{31}x^l + T_{32}y^l + T_{33}z^l + T_{34}}$$

$$v' = \frac{v}{w} = \frac{T_{21}x^l + T_{22}y^l + T_{23}z^l + T_{24}}{T_{31}x^l + T_{32}y^l + T_{33}z^l + T_{34}}$$

Then, we have

$$0 = u - (u'w)$$

$$0 = v - (v'w)$$

$$T_{11}x^l + T_{12}y^l + T_{13}z^l - u'x^lT_{31} - u'y^lT_{32} - u'z^lT_{33} - u'T_{34} = 0$$

$$T_{21}x^l + T_{22}y^l + T_{23}z^l - v'x^lT_{31} - v'y^lT_{32} - v'z^lT_{33} - v'T_{34} = 0 \quad (3)$$

Considering that there are “n” LiDAR data points collected for calibration, from the above equation, we have:

$$\begin{bmatrix}
 x'_1 & y'_1 & z'_1 & 1 & 0 & 0 & 0 & 0 & -u'_1x'_1 & -u'_1y'_1 & -u'_1z'_1 & -u'_1 \\
 0 & 0 & 0 & 0 & x'_1 & y'_1 & z'_1 & 1 & -v'_1x'_1 & -v'_1y'_1 & -v'_1z'_1 & -v'_1 \\
 x'_2 & y'_2 & z'_2 & 1 & 0 & 0 & 0 & 0 & -u'_2x'_2 & -u'_2y'_2 & -u'_2z'_2 & -u'_2 \\
 0 & 0 & 0 & 0 & x'_2 & y'_2 & z'_2 & 1 & -v'_2x'_2 & -v'_2y'_2 & -v'_2z'_2 & -v'_2 \\
 \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
 x'_n & y'_n & z'_n & 1 & 0 & 0 & 0 & 0 & -u'_nx'_n & -u'_ny'_n & -u'_nz'_n & -u'_n \\
 0 & 0 & 0 & 0 & x'_n & y'_n & z'_n & 1 & -v'_nx'_n & -v'_ny'_n & -v'_nz'_n & -v'_n
 \end{bmatrix}
 \begin{bmatrix}
 T_{11} \\
 T_{12} \\
 T_{13} \\
 T_{14} \\
 T_{21} \\
 T_{22} \\
 T_{23} \\
 T_{24} \\
 T_{31} \\
 T_{32} \\
 T_{33} \\
 T_{33}
 \end{bmatrix}
 = 0
 \tag{4}$$

Eq. (4) can be reorganized using the singular value decomposition (SVD) as follows:

$$\begin{aligned}
 A \cdot T &= 0 \\
 U \cdot D \cdot V^T &= SVD(A) \\
 A &= U \cdot D \cdot V^T
 \end{aligned}$$

where the last column of the V is the smallest singular value corresponding to A. Then, [R, t] can be obtained by the following equations:

$$\begin{bmatrix}
 T_{11} & T_{12} & T_{13} & T_{14} \\
 T_{21} & T_{22} & T_{23} & T_{24} \\
 T_{31} & T_{32} & T_{33} & T_{34}
 \end{bmatrix}
 = [R \quad t]$$

$$R = \begin{bmatrix}
 T_{11} & T_{12} & T_{13} \\
 T_{21} & T_{22} & T_{23} \\
 T_{31} & T_{32} & T_{33}
 \end{bmatrix}$$

$$U \cdot D \cdot V^T = SVD(R)$$

$$D = \begin{bmatrix}
 1 & 0 & 0 \\
 0 & 1 & 0 \\
 0 & 0 & 1
 \end{bmatrix}$$

$$R = U \cdot D \cdot V^T
 \tag{5}$$

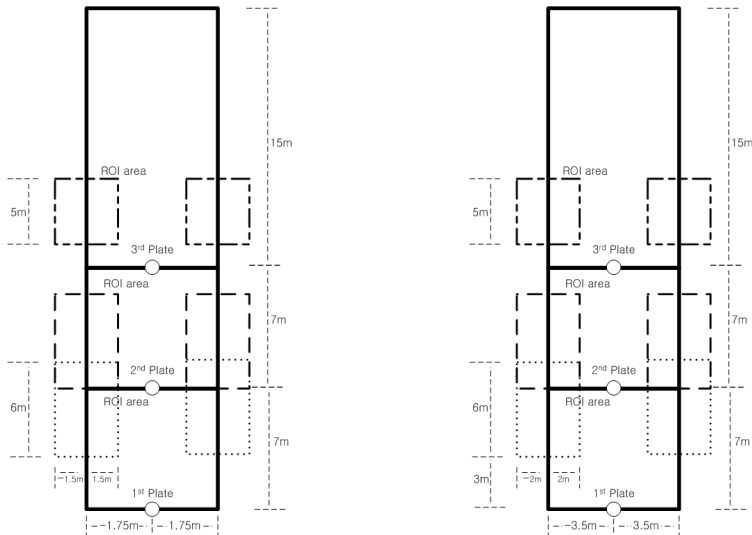


### 3 Road Model

A drivable road is represented virtually as a set of articulated multi-plates linked by rotational joints[1,2], as shown in Figs. 3 and 4. The rotational joint that links between plates may be either of a single rotation or of multiple rotations, depending on whether the road model can be defined on a flat planar surface or should take a hilly 3D surface into consideration with 2 or 3 rotations. For simplicity, here, we consider only the case where the road can be modeled on a flat planar surface, meaning a single rotational joint about the vertical axis between the plates.

#### 3.1 Modeling with Serially Linked Articulated Plates

The plate model of a road consists of three plates as shown in Fig.3. Fig.3 (a) shows the plate model with the region of interest (ROI) defined for a vision sensor, while Fig. 3(b) shows the plate model with ROI defined for LiDAR. The ROIs are defined to represent the areas to search for the predefined features such as curbs and/or lanes. The size of the 1<sup>st</sup>plate for vision is set to be 3.5m in width and 7m in length. The width was defined to be 3.5m to reflect the width of a one way road. The 2<sup>nd</sup>plate is the same size as the 1<sup>st</sup>plate. The size of the 3<sup>rd</sup>plate is set be of the same width but of the longer length of 15m. The size of the ROI areas of the 1<sup>st</sup>and 2<sup>nd</sup>plates is defined to be 3m in width and 6m in length and that of the 3<sup>rd</sup>plate is 5m in length, a little shorter than that of the 1<sup>st</sup>and 2<sup>nd</sup>plates.



(a) The Plate model with ROI defined for vision (b) The Plate model with ROI defined for LiDAR

**Fig. 3.** Road Model with the Plates for Vision and LiDAR

On the other hand, the width of the 1<sup>st</sup>plate for LiDAR is set to be 7m, reflecting the entire width of a two-way road, and the length was defined to be 7m, which is the same as that for vision. The 2<sup>nd</sup>plate is of the same width and length as the 1<sup>st</sup>plate. The 3<sup>rd</sup>plate is of the same length as that for vision but 7m in width. The ROIs are different from those for vision in terms of their width and length. The size of ROI areas of the 1<sup>st</sup>plate and the 2<sup>nd</sup> plate are the same: 2m in width and 6m in length, and the ROI area of the 3<sup>rd</sup>plate is defined to be 2m in width and 5m in length.

### 3.2 State Variables as Model Parameters

In order to formalize the proposed plate model, we assign coordinate frames to individual plates as well as to the vehicle, as shown in Fig. 4. The coordinate frame for a plate is set at the center of the bottom line of the plate such that we can define a translation on (x, y) plane and a rotation  $\theta$  about z axis perpendicular to the plate. The vehicle coordinate is defined at any position and orientation on the bottom line of the plate 1. In Fig. 4,  $c_1$ ,  $c_2$ , and  $c_3$  represent the origin points of individual plate coordinates,  $v_0$  represents the orientation of the vehicle,  $p_{10}$  the orientation of the 1<sup>st</sup>plate,  $p_{20}$  as the orientation of the 2<sup>nd</sup> plate,  $p_{30}$  represents the orientation of the 3<sup>rd</sup>plate and  $d$  represents the vehicle offset from center of the first plate.  $\theta$  corresponds the angle between the orientation of the vehicle and the orientation of the 1<sup>st</sup> plate,  $\alpha$  represents the angle between the orientation of 1<sup>st</sup>plate and that of the 2<sup>nd</sup> plate,  $\beta$  represents the angle between the orientation of the 2<sup>nd</sup> and 3<sup>rd</sup> plate and the location of the obstacle is defined as  $x_o$ ,  $y_o$ . State variable  $X$  is defined by the six parameters mentioned above:  $d$ ,  $\theta$ ,  $\alpha$ ,  $\beta$ ,  $x_o$ , and  $y_o$ . The angle  $\theta$  represents the steering control angle for controlling the orientation of the vehicle in lane with changes in the road curvature and angles  $\alpha$  and  $\beta$  represent steering control angles for controlling future orientations of lanes with changes in the road shape. State vector is given as in (3):

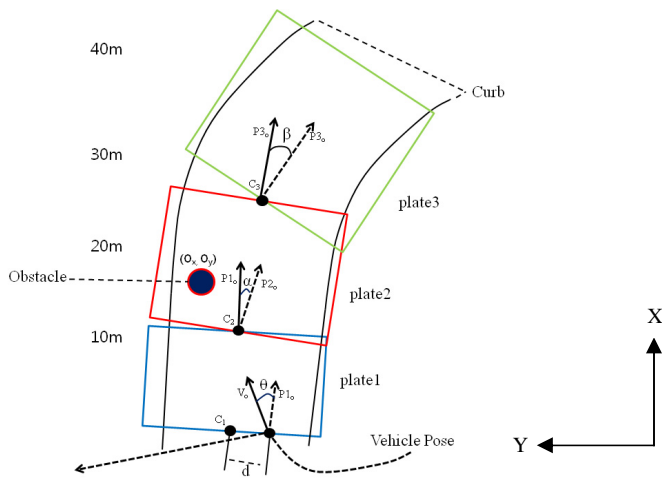


Fig. 4. Parameters for the 3 Plate Road Model

$$X = \{d \quad \theta \quad \alpha \quad \beta \quad x_o \quad y_o\}^T \quad (6)$$

### 3.3 Model Prediction: Update of Plates in Time

As the time changes from  $t_0$  to  $t_1$ , the result of the road plate model also changes. To predict these changes, a plate model at time point  $t_1$  is predicted based on the road plate model at time point  $t_0$ . The state variable of the prediction plate model is defined as shown in the following equation:

$$X' = \{d' \quad \alpha' \quad \beta' \quad \varphi\}^T \quad (7)$$

In equation(4),  $d'$  is the distance from the predicted road plate model,  $\alpha'$  is the angle between the vehicle and the 1<sup>st</sup>plate,  $\beta'$  is the angle between the 1<sup>st</sup>plate and the 2<sup>nd</sup>plate and  $\varphi$  is the angle between the 2<sup>nd</sup>plate and the 3<sup>rd</sup> plate.

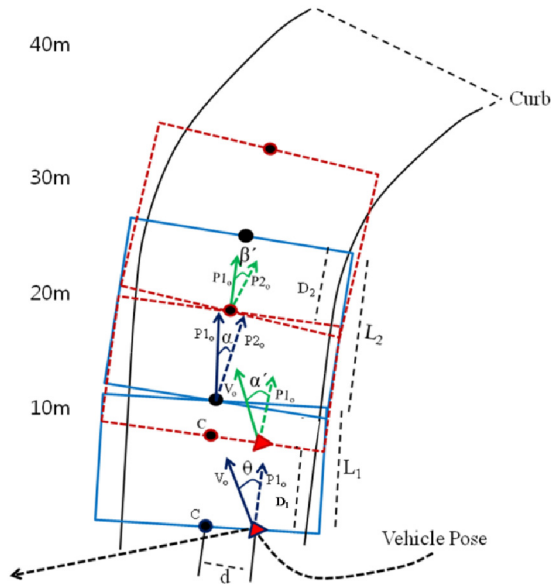


Fig. 5. Prediction of the 3 Plate Road Model and Vehicle Pose

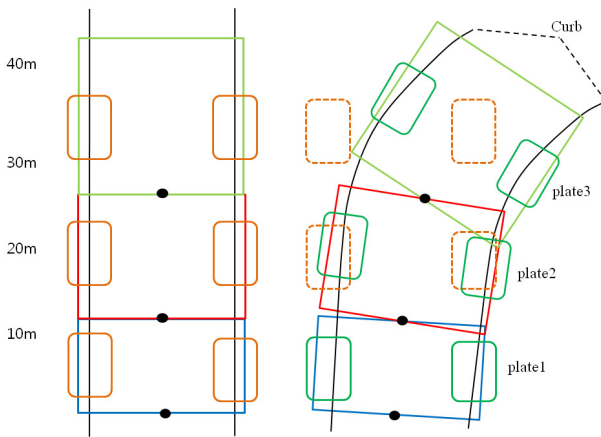
In Fig.5,  $D_1$  represents the distance travelled by the vehicle and  $L_1$  represents the length of the 1<sup>st</sup>plate. The prediction plate model can be defined by the following equation:

$$\begin{aligned}
 d' &= d + \Delta d \\
 \alpha' &= \left(1 - \frac{D_1}{L_1}\right)\theta + \frac{D_1}{L_1}\alpha \\
 \beta' &= \left(1 - \frac{D_2}{L_2}\right)\alpha + \frac{D_2}{L_2}\beta \\
 \varphi &= \frac{\beta'}{\alpha'}\beta
 \end{aligned}
 \tag{8}$$

where  $\Delta d$  implies lateral displacement of the vehicle during sampling time. This equation implies that as the vehicle’s travel distance becomes close to  $L_1$ , the predicted angle of the 1<sup>st</sup>plate indicates a value closer to  $\alpha$ . This also means that in the case of the 2<sup>nd</sup>plate too, as the vehicle’s travel distance becomes close to  $L_2$ , the angle  $\alpha$  becomes closer to angle  $\beta$ , and in the case of the 3<sup>rd</sup>plate, the angle  $\beta$  becomes closer to angle  $\varphi$  and so on.

### 3.4 Update Region of Interests

When the structure of a road plate model is defined, the structure of the relevant ROI is also defined simultaneously. Even when the size or shape of the ROI has been defined, setting the ROI area as a region that contains features (lane features or curb features) is difficult. For instance, whereas the location of the ROI area hardly changes in straight sections, as shown on the left-hand side figure of Fig. 6; the location of the ROI area changes in curved sections as shown in the right-hand side figure.



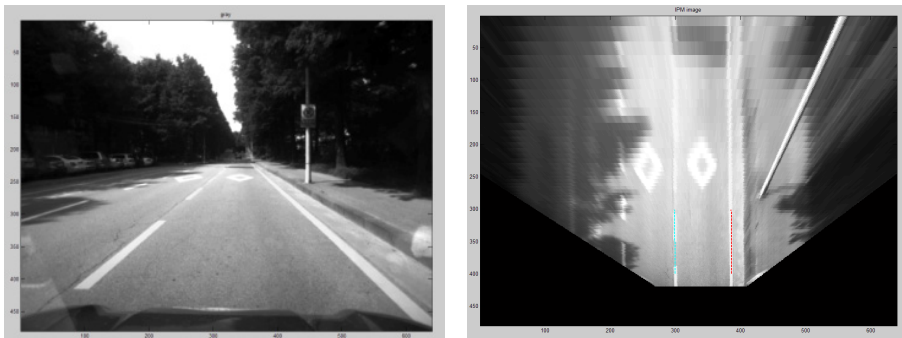
**Fig. 6.** ROIs defined for Searching Features

To set a region that contains features as an ROI area, the ROI area should be defined based on the prediction plate model and the location of the ROI area should be changed to correspond to the location of the prediction plate model. Since the prediction plate model indicates the locations of features that can be predictable, the probability of features to be in an ROI area based on the prediction plate model is much higher than the probability of the features to be in an ROI area set at an arbitrary location.

## 4 Drivable Road Identification by Vision

### 4.1 Inverse Perspective Mapping(IPM)

Road images are converted into top-view images before being used. There are several advantages of using top-view images like the images in Fig.7. First, the perspective effect in general camera images can be removed by using IPM images. For instance, parallel lanes on a road are shown as triangular shapes in general camera images. However, in IPM images, the lanes are shown to be parallel to each other and this makes the detection of vertical lanes much easier when compared to general camera images. Second, when an IPM image is used to detect the lanes of a road, only the necessary part of the image is used using the Region of Interest (ROI) so that the total processing time is reduced compared to using the entire image. This is important in autonomous vehicles that have a real-time constraint. Third, the shapes of lane features are uniform and thus desired features can easily be obtained by using only simple edge masks. A method used to obtain top-view images is projective transformation and this method can define 2D images using Eq(6). In Eq (6) and are expressed as homogeneous 3-vectors and is a non-singular 3x3 matrix (Eq(7),



**Fig. 7.** Inverse Perspective Mapping (IPM) image

$$x' = Hx \tag{9}$$

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \tag{10}$$

Eq(7) can be re-organized to obtain a solution for H. Then, Eq(7) can be expressed as Eq(9). To obtain the solution, there should be at least four corresponding points.

$$A = \begin{bmatrix} x_1 & y_1 & z_1 & 1 & 0 & 0 & 0 & 0 & -u_1x_1 & -u_1y_1 & -u_1z_1 & -u_1 \\ 0 & 0 & 0 & 0 & x_1 & y_1 & z_1 & 1 & -v_1x_1 & -v_1y_1 & -v_1z_1 & -v_1 \\ & & & & & & & \vdots & & & & \\ x_n & y_n & z_n & 1 & 0 & 0 & 0 & 0 & -u_nx_n & -u_ny_n & -u_nz_n & -u_n \\ 0 & 0 & 0 & 0 & x_n & y_n & z_n & 1 & -v_nx_n & -v_ny_n & -v_nz_n & -v_n \end{bmatrix} \tag{11}$$

$$AH' = 0 \tag{12}$$

An H matrix can be obtained from the solution in Eq(9) using SVD, and the original image can be mapped into an IPM image using the H matrix obtained from this equation. Then, the result as shown in Fig.7 is obtained.

## 4.2 Lane Detection

### 4.2.1 Edge Detection

To detect edges from camera images, Gaussian smoothing and Sobel masking are performed as pre-processes. To detect lanes, camera images are converted into

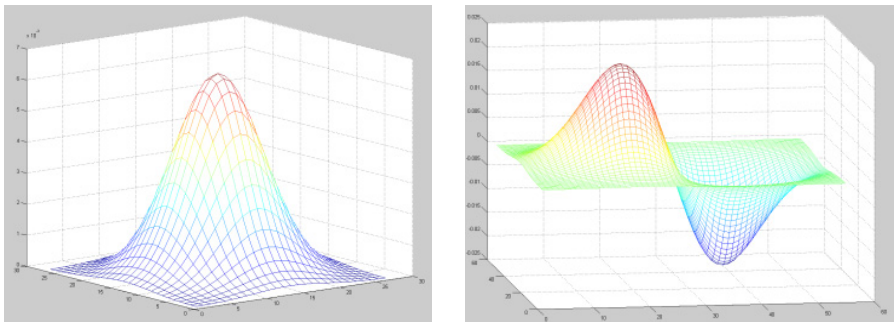
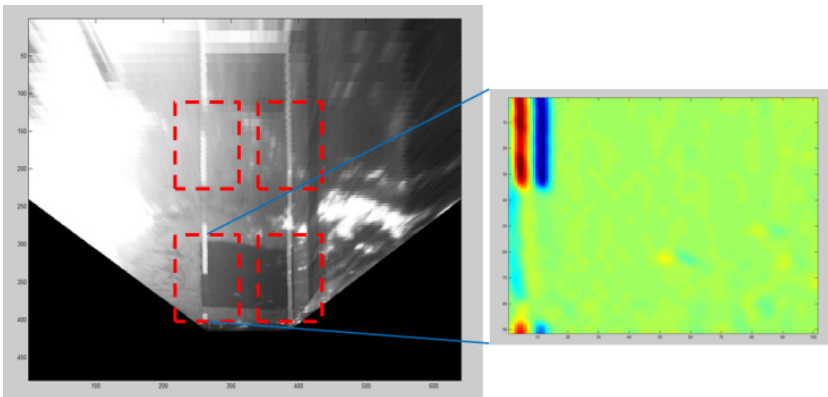


Fig. 8. Gaussian Mask and Sobel Mask

top-view images as if the road were actually viewed from above through the IPM (Inverse Perspective Mapping) method. In the case of IPM images, lane features are detected as vertical lines so that edges can be extracted more easily than from general camera images. Although there are many methods to detect edges, Sobel masking was used since it can produce good outcomes in a short time and Gaussian smoothing was used to remove noise in camera images. The shapes of a Gaussian mask and a Sobel mask are as shown in Fig.8.

#### 4.2.2 Land Feature Detection

If Gaussian Masking and Sobel Masking are performed simultaneously in an ROI area, the result as shown on the right-hand side of Fig.9 can be obtained. However, the result in Fig.9 cannot be said to be a lane feature. First, on the right-hand side of Fig.9, the red edge is a high edge obtained by performing Sobel masking and the blue edge is a low edge obtained through Sobel masking. If the edges are extracted from the lane, the high and low edges will be simultaneously extracted. The high edge will be extracted from the right-hand side of the lane and the low edge will be extracted from the left-hand side. However, when detecting the lane, it is difficult to determine whether to detect the right side edge or the left side edge of the lane because, strictly speaking, neither of the two edges can be regarded as a lane but they are just locations that indicate the edges of the lane. In general, when the location of a lane is defined, the center of the lane is defined as the lane. Therefore, the center of the lane is detected using the high edge and the low edge. To detect the center of a lane, both the high edge and the low edge of the lane are used. First, to find the peak points of the high edge and the low edge, peak detection was performed for each of the two edges. Although peak detection can be performed by finding the points that are the peaks of edges using general parabola fitting, the normalized weighted average method was additionally used to detect more accurate peaks of the edges. A general parabola equation is as follows.



**Fig. 9.** Land Feature Image

$$f(x) = ax^2 + bx + c \tag{13}$$

The above equation is differentiated as follows:

$$f'(x) = 2ax + b \tag{14}$$

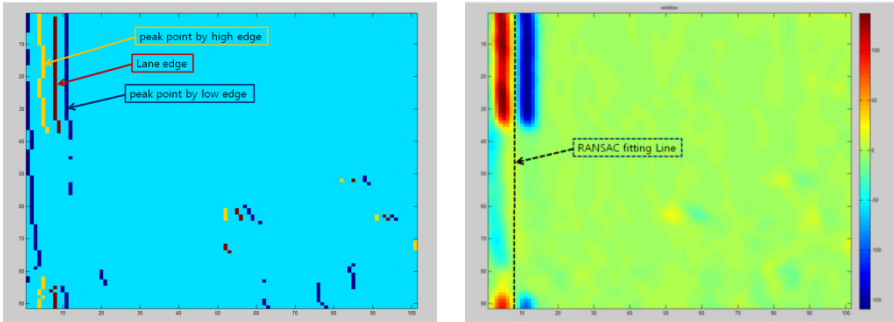
By putting  $f'(x) = 0$ , the local maximum can be obtained from the parabola. The value of  $x$  from which the local maximum can be obtained is as follows:

$$x = -\frac{b}{2a} \tag{15}$$

If the normalized weighted average is applied to the above equation, the equation will become as follows:

$$\begin{aligned}
 x_{peak} &= \frac{\sum_{t=-n}^n (w_{x+t} - \min(w))(x+t) / (\max(w) - \min(w))}{\sum_{t=-n}^n (w_{x+t} - \min(w)) / (\max(w) - \min(w))} \\
 &= \frac{\sum_{t=-n}^n (w_{x+t} - \min(w))(x+t)}{\sum_{t=-n}^n (w_{x+t} - \min(w))}
 \end{aligned} \tag{16}$$

In the above equation,  $w$  is the image intensity and  $n$  is the mask size.



**Fig. 10.** Peak Edge Extraction, RANSAC Line Fitting

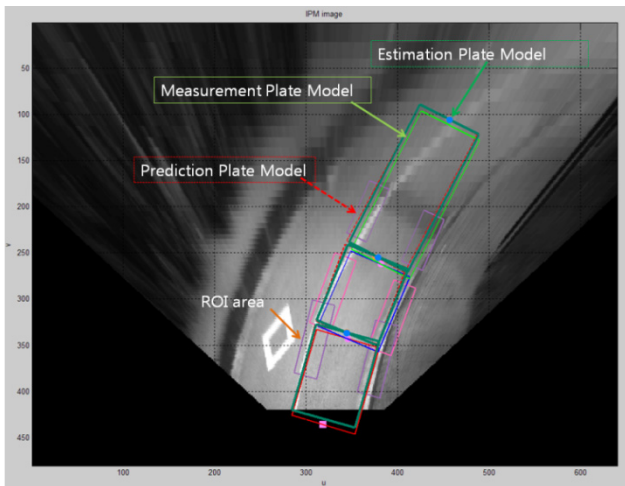
On the left-hand side of Fig. 10 shown above, the yellow point indicates the resultant peak extracted from the high edge and the blue point represents the result of extraction from the low edge. The lane edge point can be extracted using the intermediate value of



the edge points obtained from these two results. The lane edge point obtained as such was used to perform RANSAC line fitting. Where, approximately 0.5 pixels was used as a threshold because a pixel represents a distance of approximately 0.2m, the width of a general lane is approximately 0.12m and this lane width corresponds to approximately 0.5 pixels. Through this method, the result as shown on the right-hand side of Fig.10 was obtained.

### 4.3 3 Plate Road Model for Vision

Fig.11 is the result of the application of a plate model to vision information. First, lanes were detected from each plate using two ROI areas. The angles of the lanes were calculated using the lines detected from the lanes. To obtain the angles, line vectors were made, the reference vector was defined as  $[0 \ 1]'$  and then “atan2” function was used. Although arc-cosines are generally used, but here atan2 was used to obtain parameter  $\theta$  of the state variable since the results of arc-cosine are indicated in the range of  $0 < \theta < \pi$  when calculating them on the computer. The extracted angles were used to set the slope of the 1<sup>st</sup>plate and the distance from the center of the bottom side of the 1<sup>st</sup>plate to the sensor was calculated to obtain the state variable  $d$ , indicating the distance of the vehicle from the road center. The same method was used to calculate the angles of lanes on the 2<sup>nd</sup> and 3<sup>rd</sup> plates. The results calculated like this are the results of the measurement plate model shown in Fig.11. Using the measurement plate model, the prediction method mentioned in section 3.3 was applied to create prediction results. The plate model expressed in red dotted lines in Fig.11 represents the prediction results. Finally, using the prediction results and measurement results, the method set forth under section 4.4 was applied to obtain estimation results.



**Fig. 11.** Plate Road Model overlaid on Vision Data

#### 4.4 Weighted Average of Vision Plate Model

To obtain estimation plate models using prediction plate models and measurement plate models, a weighted average method was used. The weights were determined based on the results obtained through experiments. In the following equation,  $\Psi$  represents the weights. The results of the prediction have higher weighted values than measurement results because the results of prediction are relatively more accurate than measurement results since completely wrong measurement are sometimes produced when lanes are detected from areas where there is no lane in fact. Weighted averages are expressed as an equation as follows:

$$\begin{aligned} \Psi &: \textit{weight} \\ \Psi &> 0.5 \\ \gamma &= \Psi * S_m + (1 - \Psi) S_p \end{aligned} \tag{17}$$

In the above equation,  $S_m$  represents the state variable of the measurement plate model and  $S_p$  represents the state variable of the prediction plate model.  $\gamma$  represents the estimation plate model calculated from the two state variables.

## 5 Drivable Road Identification by LiDAR

### 5.1 Multi Layered LiDAR Sensor

From the control point of view, the drivable road region is a certain frontal area from the vehicle head since each sensor has limited effective range. The LiDAR sensor has range information intuitively at the scanned line which makes cumulative data set definitely flat or not as the vehicle proceeds. Generally some curbs or obstacles are detected and positional information has been used for the vehicle navigation. However, these schematics may have some risky conditions in case of intersection or roads where there is no curb. Since it is considered that the vehicle navigates both the paved lane road and open terrain, the accumulated summing on each scanned line and registering by some polygonal area in memory are useful for identification.

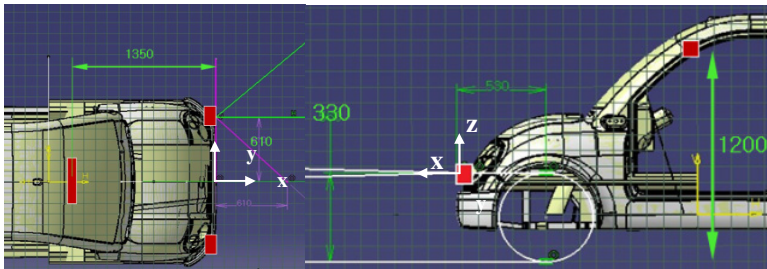
The LiDAR sensor was developed for the distance warning system in 1996 and improved for Adaptive Cruise Control functionality in 1997[10]. The curb was used as one of the most useful targets for road detection and track following control since the curb is generally located at the road boundary and its distinction between road and footpath makes clear for detecting the road [13]. Some research calculated the lane parameters such as the curvature, yaw angle, i.e. angle between the lane direction and ego-vehicle motion, relative offset between the vehicle and the road boundary and the lane width by separating lane from the road of different reflectivity [14].

In this work, the ibeo LUX Multilayered LiDAR was used for range measurement and its specifications are as follows:

**Table 1.** The Specifications of ibeo Multilayer LiDAR

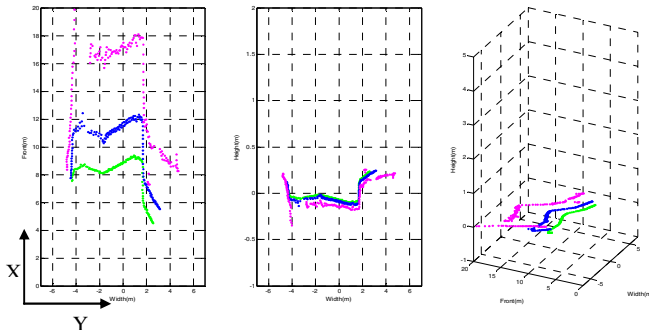
Horizontal FOV	85 °
Vertical FOV	3.2 °
Update rate	12.5 Hz
Max. Position errors	10 cm
Range	200 m
Angular resolution	H: 0.125 ° , V: 0.8 °

The system has four layers which allows for angle compensation by means of multi scan levels with different vertical angles and enables the vehicle to detect the object robustly even under a bumped ground. Two LiDAR sensors are installed in the head of the vehicle and the position and range for left one is presented as below.



**Fig. 12.** Sensor configuration for LiDAR and camera

The sensor is tilted downward by an angle of 12° to detect road plane where the lower three layers take part in the road plane detection and upper layer data is analyzed for obstacle detection as this layer is emitted and reflected almost horizontally.



**Fig. 13.** LiDAR scanned results of the road

Fig. 13 shows a result of LiDAR scanning by the vehicle driving. The left one is plotting the X-Y plane data where X axis is corresponding to the road/driving

direction and the Y axis is road width. Middle plot of the Fig. 13 gives depth information of the road and the right shows the three dimensional construction by sensor coordinate transformation.

## 5.2 Curb Detection

To detect curbs using LIDAR, the sizes of roads were assumed to be uniform. The width of roads is determined as 7m and the RANSAC line-fitting method used for vision is used as a curb detection algorithm. Since the accuracy of the sensor was approximately 10cm, 0.1 m was used as a threshold. The right-hand side of Fig.14 shows an expanded image of the red dotted line area of the left-hand side figure which represents the result of RANSAC line fitting. From the figure, it can be seen that curb features are accurately detected even when noise occurs.

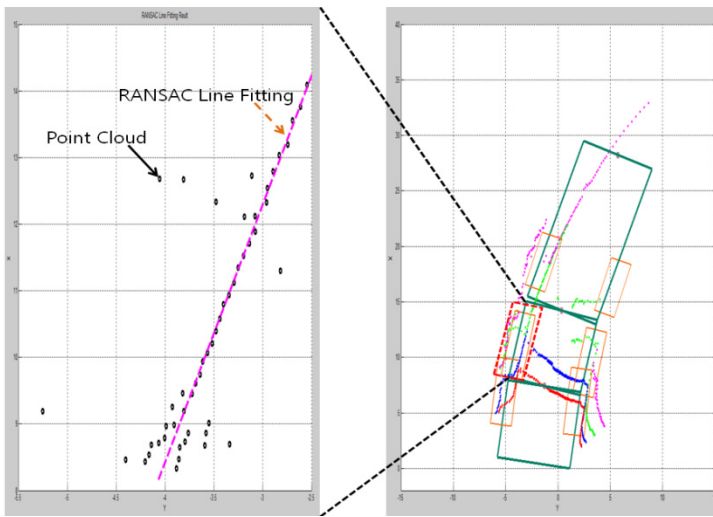


Fig. 14. Curb Detection based on LiDAR

## 5.3 Weighted Average of LiDAR Plate Model

We use a weighted average method to estimate the fusion result of the measurement and prediction plate model results. The weights were determined based on the results obtained through experiments. In below,  $\Psi$  represents the weight between measurements and prediction results. Measurements which have high accuracy have higher weight values. In this case, higher weighted values were used for prediction results than for the measurement results because wrong measurement results are sometimes produced when curbs are detected from areas without any curb or wrong areas. Weighted averages are expressed as an equation as follows:

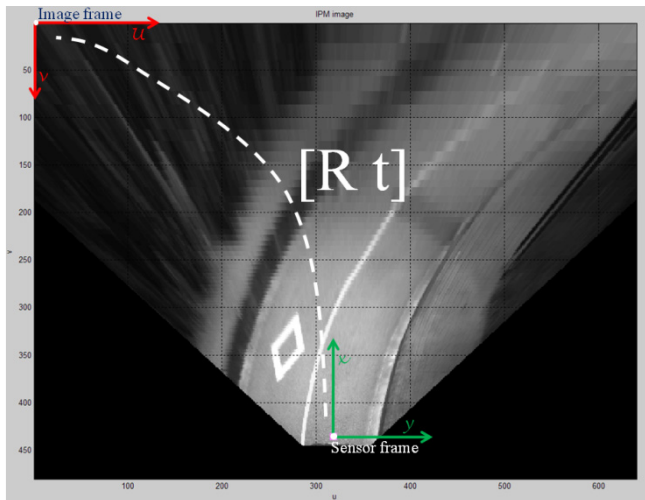
$$\begin{aligned}
 \Psi &: \textit{weight} \\
 \Psi &> 0.5 \\
 \gamma &= \Psi * S_m + (1 - \Psi) S_p
 \end{aligned}
 \tag{18}$$

In the above equation,  $S_m$  represents the state variable of the measurement plate model and  $S_p$  represents the state variable of the prediction plate model.  $\gamma$  represents the estimation plate model calculated from the two state variables.

## 6 Fusion of Vision and LiDAR

### 6.1 Image and Sensor Frame Transformation

To express plate models of camera images, the coordinates of image frames should be converted into those of sensor frames. The coordinates of an image frame begin with the top left of the image. However, the coordinates of a sensor frame are based on the position of the sensor in the image. The reason why sensor frames are used in this way is that since plate models are made based on sensor frames, the coordinates of image frames should be changed into those of sensor frames to express plate models of images. If the origin point of a sensor frame in an image frame is assumed to be  $P_{org}(u_{sensor}, v_{sensor})$ , and a current arbitrary point in the image frame is assumed to be  $P_{image}(u_{sensor}, v_{sensor})$ , the equation for changing the coordinates of the point in the image frame into the coordinates of the point in the sensor frame is as follows:



**Fig. 15.** Coordinates Transformation

$$\begin{aligned}
 P_{sensor} &= \begin{bmatrix} u_{current\_sensor} \\ v_{current\_sensor} \end{bmatrix} = R(-90^\circ)(P_{image} - P_{org}) \\
 &= \begin{bmatrix} \cos(-90^\circ) & -\sin(-90^\circ) \\ \sin(-90^\circ) & \cos(-90^\circ) \end{bmatrix} \begin{bmatrix} u_{current\_img} - u_{sensor} \\ v_{current\_img} - v_{sensor} \end{bmatrix}
 \end{aligned}
 \tag{19}$$

### 6.2 Fusion of Vision Results and LIDAR Results

Fig. 16 shows a flow chart of a method to fuse vision results and LIDAR results. The results were fused using a simple method of weighted averages which gives more weight values to sensors that produce better results. First, the state variables used for the fusion are as follows:

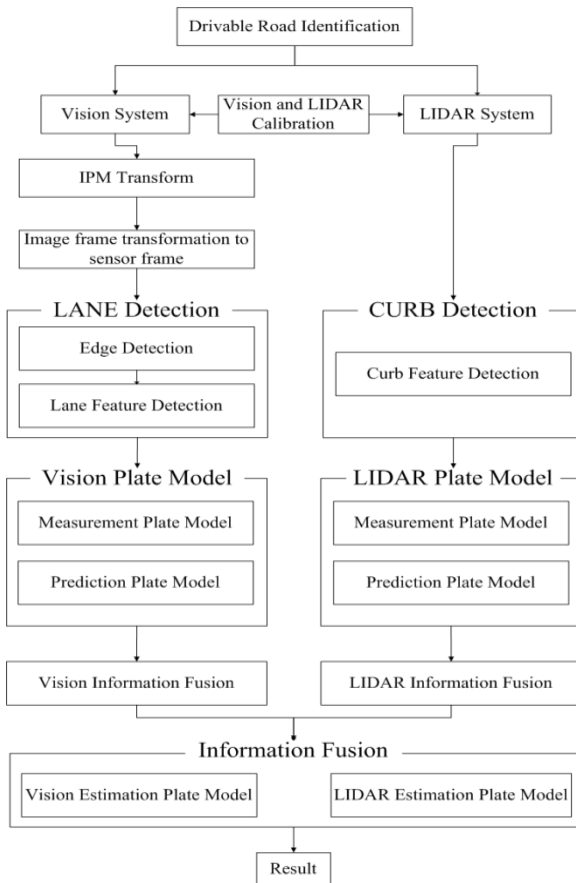


Fig. 16. Flow chart of Fusion Process

$$\begin{aligned}
 X_{Ve} &= \{s_{Ve1} = d_v, s_{Ve2} = \theta_v, s_{Ve3} = \alpha_v, s_{Ve4} = \beta_v\}^T \\
 X_{Le} &= \{s_{Le1} = d_v, s_{Le2} = \theta_v, s_{Le3} = \alpha_v, s_{Le4} = \beta_v\}^T
 \end{aligned} \tag{20}$$

where  $X_{Ve}$  is the result of estimation using the vision plate model in the form of vectors and  $X_{Le}$  is the result of estimation using the LIDAR plate model in the form of vectors. The equation used for the fusion using vision information is as follows:

$$\begin{bmatrix} s_{ve1} \\ s_{ve2} \\ s_{ve3} \\ s_{ve4} \end{bmatrix} = \begin{bmatrix} \varphi_{v1} \\ \varphi_{v2} \\ \varphi_{v3} \\ \varphi_{v4} \end{bmatrix} \cdot \begin{bmatrix} s_{vm1} \\ s_{vm2} \\ s_{vm3} \\ s_{vm4} \end{bmatrix} + \left( 1 - \begin{bmatrix} \varphi_{v1} \\ \varphi_{v2} \\ \varphi_{v3} \\ \varphi_{v4} \end{bmatrix} \right) \cdot \begin{bmatrix} s_{vp1} \\ s_{vp2} \\ s_{vp3} \\ s_{vp4} \end{bmatrix} \tag{21}$$

$s_{ve1}$  is the pose of the vehicle measured by the vision sensor,  $s_{ve2}$  is the angle between the orientation of the vehicle and the 1<sup>st</sup> plate,  $s_{ve3}$  is the angle between the 1<sup>st</sup> plate and the 2<sup>nd</sup> plate and  $s_{ve4}$  is the angle between the 2<sup>nd</sup> plate and the 3<sup>rd</sup> plate. The equation used for the fusion using LIDAR information is as follows:

$$\begin{bmatrix} s_{Le1} \\ s_{Le2} \\ s_{Le3} \\ s_{Le4} \end{bmatrix} = \begin{bmatrix} \varphi_{L1} \\ \varphi_{L2} \\ \varphi_{L3} \\ \varphi_{L4} \end{bmatrix} \cdot \begin{bmatrix} s_{Lm1} \\ s_{Lm2} \\ s_{Lm3} \\ s_{Lm4} \end{bmatrix} + \left( 1 - \begin{bmatrix} \varphi_{L1} \\ \varphi_{L2} \\ \varphi_{L3} \\ \varphi_{L4} \end{bmatrix} \right) \cdot \begin{bmatrix} s_{Lp1} \\ s_{Lp2} \\ s_{Lp3} \\ s_{Lp4} \end{bmatrix} \tag{22}$$

$s_{Le1}$  is the pose of the vehicle measured by the LIDAR sensor,  $s_{Le2}$  is the angle between the orientation of the vehicle and the 1<sup>st</sup> plate,  $s_{Le3}$  is the angle between the 1<sup>st</sup> plate and the 2<sup>nd</sup> plate and  $s_{Le4}$  is the angle between the 2<sup>nd</sup> plate and the 3<sup>rd</sup> plate.

The equation to fuse the results of estimation by the vision plate model and the results of estimation by the LIDAR plate model produced using the above equations is as follows:

$$X_E = \varphi_E \cdot X_{LE} + (1 - \varphi_E) X_{VE} \tag{23}$$

$X_E$  is the result of fusion of vision information and LIDAR information,  $\varphi_E$  are the weights used to fuse the two data;  $X_{LE}$  is the estimation result obtained using LIDAR and  $X_{VE}$  is the estimation result obtained using vision. Figure 17 shows the results.

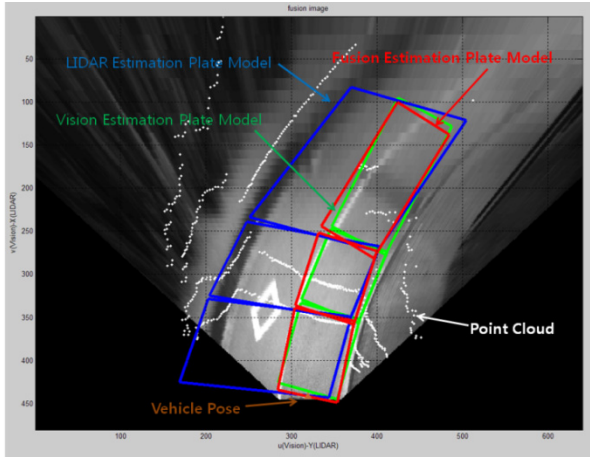


Fig. 17. Fusion Result

In Fig.17, the blue plate is the estimation result obtained using LIDAR and the green plate is the estimation result obtained using vision. Finally, the result obtained by fusing the two data sets is the red plate model. The brown shows the pose of the vehicle.

## 7 Obstacle Detection and Modeling

The method presented here for the obstacle detection based on LiDAR data is as follows: First, we compute the virtual line formed by the intersection of the planar LiDAR beam and the first plate of the model. This virtual line serves as the reference line for detecting obstacles, since the data captured in front of this virtual line may represent obstacles. Considering the possible deviation of the actual road surface from the flat planar surface assumption, for instance, a slightly elevated road center, we define a tolerance region in front of the virtual line, such that the tolerance line thus defined serves the actual reference line for detecting obstacles. Then, we project LiDAR data inside the plate onto the axis perpendicular to the virtual line we obtained for the reference, or, equivalently, the vehicle forward direction axis, in order to generate a histogram along the axis. Any peaks existing in front of the actual reference line are taken as obstacles. Note that the height of an obstacle peak indicates indirectly the size of the obstacle. In Fig. 18, we experimented our algorithm with single stationary pedestrian present in road area. When the vehicles moves, each layer detects the pedestrian at around 6m from the vehicle, which is the actual position of pedestrian on road. Histogram plots in Fig.18(b) shows high peaks at obstacle position whereas the small peak in fig.18(b) shows the measurement noise. Results in Fig 18(b) are shown for obstacle detection in layer 1~4 from bottom to top respectively.



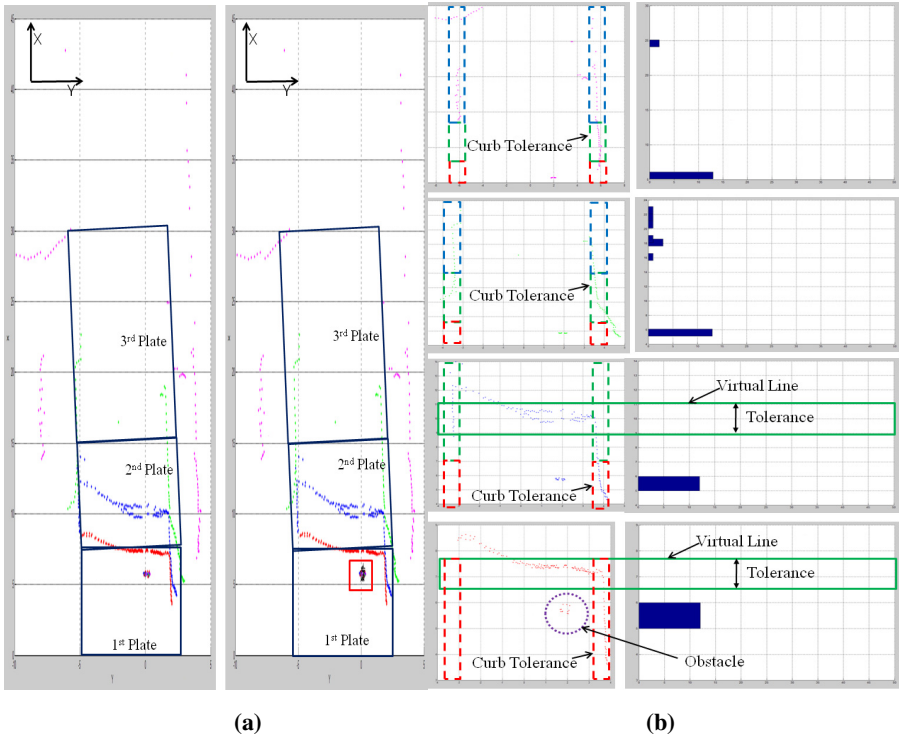


Fig. 18. The Result of Obstacle Detection

## 8 Experimentation

The above Figure shows the results of the actual application of the algorithm proposed in the present study. The results are for 10, 20, 30, 40, 50, and 60 frames which are part of the entire drive. The images on the left-hand side show the actual road and


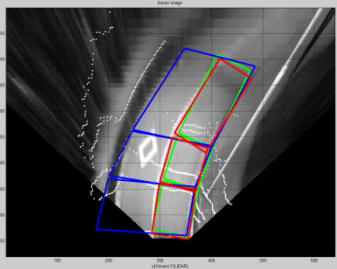
time	Original	Fusion Result
10 frame		

Fig. 19. Test Result

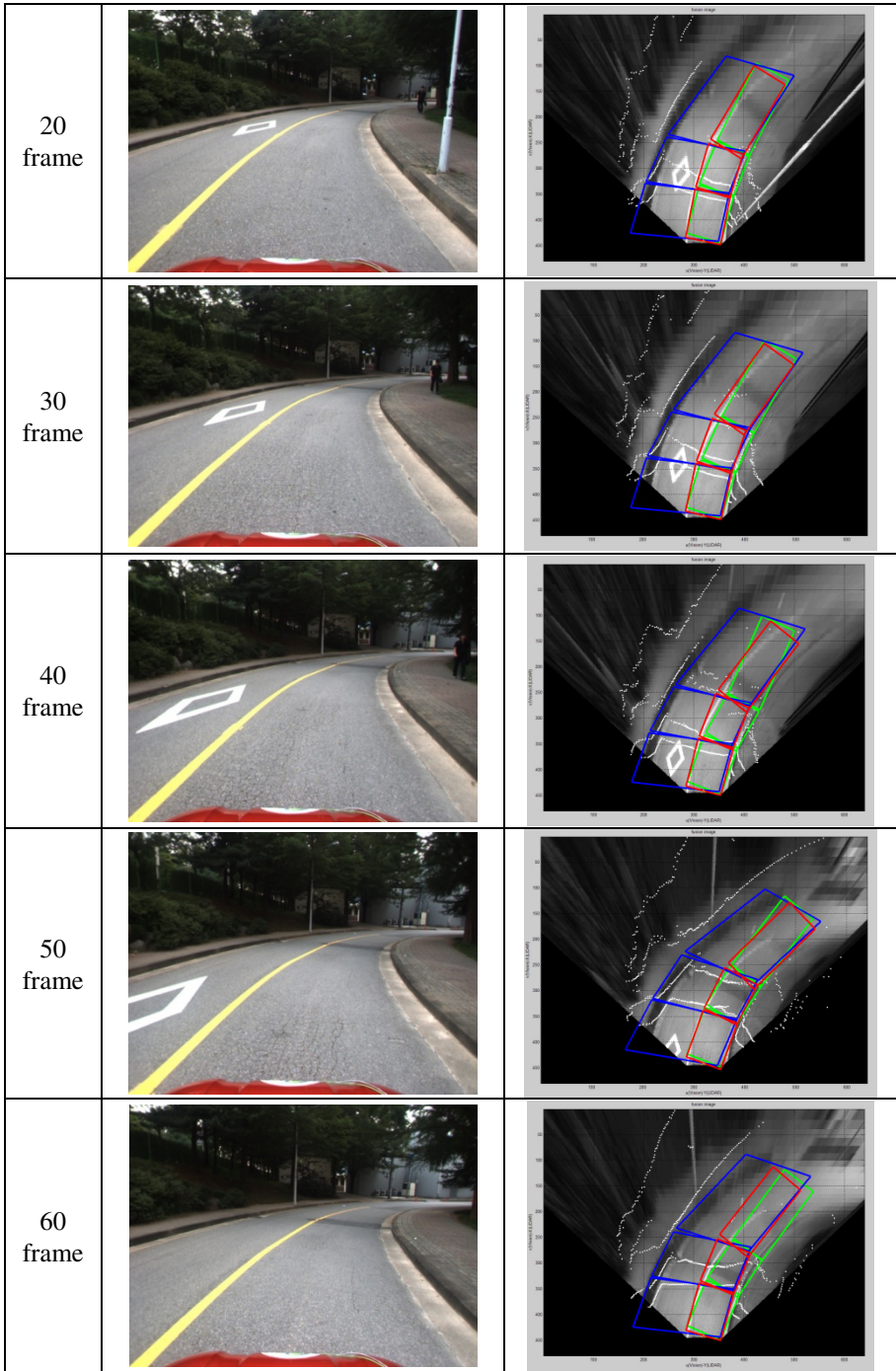


Fig. 19. (continued)

the images on the right-hand side show the results of the application of vision and LiDAR information to the proposed algorithm at individual frames. In the right-hand image, the green area is the result obtained using vision information, the blue area is the result obtained using LiDAR information and the red area is the result obtained using the fusion of vision and LiDAR data. Through these results, it is identified that the proposed algorithm works well in actual drives.

## 9 Conclusion

A real-time modeling of drivable road along with the vehicle movement by integrating a 4-layered Light Detection and Ranging device (LiDAR) and a vision camera is presented. The experimental results demonstrate that the road surface, curbs and obstacles identified by the multilayer LiDAR fused with the lanes and crossings recognized by the vision camera can provide a viable solution for determining the multi-plate model of drivable road as well as the pose of vehicle and obstacles on the modeled road. The contribution of this paper is the integrated framework of LiDAR and vision for the real-time modeling of a drivable road as well as for the identification of vehicle and obstacle poses on the modeled road.

**Acknowledgements.** The authors would like to thank Mr. Youngjin Suh and Mr. Ilyas Meo for their contributions to collecting some experimental data as well as editing the manuscript. This research was supported by a grant (I091003) from Gyeonggi Technology Development Program funded by Gyeonggi Province. This work was also supported in part by KORUS-Tech (Korea-US Tech Program KT-2010-SW-AP-FS0-0004). Their support during the period of this research is greatly acknowledged.

## References

1. Lee, S., Lee, J.-W., Shin, D., Kwon, W., Kim, D.-Y., Roh, K., Boo, K.S.: Estimation of Vehicle Pose and Road Curvature Based on Perception-Net. In: Proceedings of the 1999 IEEE International Conference on Robotics and Automation (October 1999)
2. Lee, S., Kwon, W., Lee, J.-W.: A Vision Based Lane Departure Warning System. In: Proceedings of the 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems, Detroit, Michigan (May 1999)
3. Hillel, A.B., Lerner, R., Levi, D., Raz, G.: Recent progress in road and lane detection: a survey. In: Machine Vision and Applications. Springer Online (February 2012)
4. Bagnell, J.A., Bradley, D., et al.: Learning for autonomous navigation: Advances in machine learning for rough terrain mobility. *IEEE Robotics and Automation Magazine* 17(2), 74–84 (2010)
5. Markoff, J.: Google Cars Drive Themselves, in Traffic. *The New York Times*, <http://www.nytimes.com/2010/10/10/science/10google.html> (retrieved October 11, 2010)
6. Markoff, J.: Google Cars drive themselves, in traffic. *The New York Times* (October 9, 2010)

7. Tsogas, M., Floudas, N., Lytrivis, P., Polychronopoulos, A.: Combined lane and road attributes extraction by fusing data from digital map, laser scanner and camera. *Information Fusion* 12, 28–36 (2011)
8. Gupta, R.A., Snyder, W.S., Pitts, W.S.: Concurrent visual multiple lane detection for autonomous vehicles. In: *IEEE Int. Conf. on Robotics and Automation*, Anchorage, USA, May 3-8, pp. 2416–2422 (2010)
9. Kim, Z.: Realtime lane tracking of curved local road. In: *Proc. of IEEE Int. Transportation Sys. Conf.*, Toronto, Canada, pp. 17–20 (September 2006)
10. Lipski, C., Scholz, B., Berger, K., Linz, C., Stich, T.: A fast and robust approach to lane marking detection and lane tracking. In: *Southwestern Symp. on Image Analysis and Interpretation*, pp. 57–60 (2008)
11. Peterson, K., Ziglar, J., Rybski, P.E.: Fast feature detection and stochastic parameter estimation of road shape using multiple LIDAR. In: *IEEE Int. Conf. on Int. Robots and Systems*, Nice, France, September 22-26, pp. 612–619 (2008)
12. Ogawa, T., Takagi, K.: Lane recognition using on-vehicle LIDAR. In: *Int. Vehicles Symposium*, Tokyo, Japan, June 13-15, pp. 740–741 (2006)
13. Lindner, P., Rochter, E., Wanielik, G., Takagi, K., Isogai, A.: Multi channel LiDAR processing for lane detection and estimation. In: *Proc. of Int. IEEE Conf. on Intelligent Transportation Systems*, St.Louis, USA, October 3-7, pp. 202–207 (2009)
14. Manz, M., von Hundelshausen, F., Wuensche, H.-J.: A hybrid estimation approach for autonomous dirt road following using multiple clothoid segments. In: *IEEE Int. Conf. on Robotics and Automation*, Anchorage, USA, May 3-8, pp. 2410–2415 (2010)
15. Chun, C., Suh, S., Lee, S., Roh, C.-W., Kang, S., Kang, Y.: Autonomous navigation of KUVE. *Journal of Institute of Control, Robotics and Systems* 16(7) (July 2010)

# Omnidirectional Vision for Indoor Spatial Layout Recovery<sup>\*</sup>

Jason Omedes, G. López-Nicolás, and J.J. Guerrero

Instituto de Investigación en Ingeniería de Aragón. Universidad de Zaragoza, Spain  
jason.omedes@gmail.com, {gonlopez, jguerrer}@unizar.es

**Abstract.** In this chapter, we study the problem of recovering the spatial layout of a scene from a collection of lines extracted from a single indoor image. Equivalent methods for conventional cameras have been proposed in the literature, but not much work has been done about this topic using omnidirectional vision, particularly powerful to obtain the spatial layout due to its wide field of view. As the geometry of omnidirectional and conventional images is different, most of the proposed methods for standard cameras do not work and new algorithms with specific considerations are required. We first propose a new method for vanishing points (VPs) estimation and line classification for omnidirectional images. Our main contribution is a new approach for spatial layout recovery based on these extracted lines and vanishing points, combined with a set of geometrical constraints, which allow us to detect floor-wall boundaries regardless of the number of walls. In our proposal, we first make a 4 walls room hypothesis and subsequently we expand this room in order to find the best fitting. We demonstrate how we can find the floor-wall boundary of the interior of a building, even when this boundary is partially occluded by objects and show several examples of these interpretations.

## 1 Introduction

Indoor structure recovery from images is an easy task for humans but not that easy for computers. At the same time, it is a very useful task since knowing floor-wall boundaries can give us valuable information for navigation, motion planning, obstacle detection or 3D reconstruction.

This problem has been studied several times and still attracts the effort of many researchers to implement each time better algorithms. Most of these contributions work under the Manhattan-World assumption [1], which assumes the scene is composed of 3 main directions orthogonal to each other. Indoor environment usually satisfies this condition so is understandable this hypothesis is extensively used. Some examples are [2], that uses extracted lines and geometric reasoning to generate hypothesis and select the best fit, or [3] which represents the room as a 3D box and tries to recognize floor-wall boundary in cluttered rooms. There are also other works as [4] that uses Bayesian filtering over a set of floor-wall boundary hypotheses without the restriction

---

<sup>\*</sup> This work was supported by the Spanish projects DPI2012-31781, DPI2009-14664-C02-01 and FEDER funds.

of Manhattan-World assumption, but there still being 3 main directions without the imposition of orthogonality between them.

Lately, research on omnidirectional vision is taking more importance due to the wide range of vision of these images, which helps in the detection of VPs and makes visible lines much longer. However, in central catadioptric images, straight lines from the real world become conics adding the issue of geometrical complexity, implying that many of existing algorithms for conventional images are not applicable. So it is needed to come up with new methods that take in account characteristics of this kind of images.

Here, we present our work for structure recovery from images. Starting from a single omnidirectional indoor image, we extract lines from it, classify them depending on their orientation. From this classification, we select a set of points which will lead us to generate possible wall-floor boundaries, and imposing geometrical constraints we generate a first 4 walls-room hypothesis to later on expand or not this room according on how the data is distributed.

Inspired by [5], we propose a new method. This approach is more robust since it does not rely in finding corners which often are not easy to detect, also we do not need to specify the number of walls we are looking for. In addition, it is much faster, as trying every possible combination of normals vector to classify the extracted lines or combination of corners to find the room hypothesis was high time-consuming. With our new approach we avoid all these long iterations making viable its use in a sequence of images at real-time.

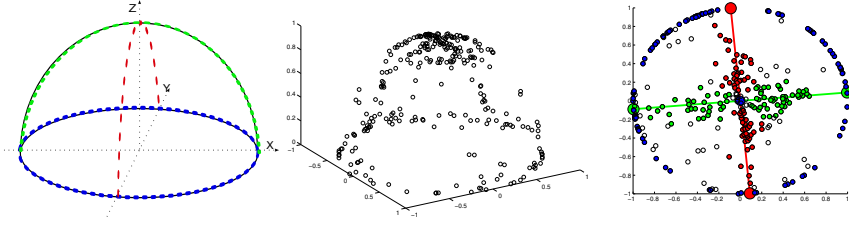
## 2 Vanishing Point Estimation through Line Detection

The first step of our proposal begins with extraction of lines from the image. Regarding to line extraction for catadioptric systems two methods are [6] [7]. Both start using Canny edge detector and linking edge pixels. The main difference is that [7] works on the catadioptric image, where lines and VPs are extracted by RANSAC. Whereas [6] uses the unitary sphere model proposed in [8] where points from the image  $\mathbf{p}_I = (X_o, Y_o, 1)$  are projected as  $\mathbf{p}_S = (X_S, Y_S, Z_S)$ . By doing this, each chain of pixels from a line in the image defines a *great circle* on the sphere which can be represented by its normal vector  $\mathbf{n} = (n_x, n_y, n_z)$ .

We use Bazin's Matlab toolbox<sup>1</sup>, but adapting the equations from para-catadioptric ( $\xi = 1$ ) to hyper-catadioptric ( $0 < \xi < 1$ ) system in order to generalize the method for a more general mirror shape. From this point, [6] proposes to test every possible combination between pairs of normal vectors to identify main directions (1 vertical, 2 horizontal) at the same time as the 3 corresponding VPs. However, this method is time-consuming and sometimes comes up with misclassifications.

We propose a new robust and fast method to classify lines parallel to the 3 dominant directions, taking in consideration two hypotheses: a) Manhattan-world assumption [1] which states the scene is build on a cartesian grid, b)  $\mathbf{Z}$  camera's axis is aligned with  $\mathbf{Z}$  reference's axis of the world, since catadioptrical systems are mainly used in wheel-based robots, so planar-motion is assumed. It is easily demonstrable that under these

<sup>1</sup> <http://graphics.ethz.ch/~jebazin>



**Fig. 1.** In the sphere model, every line from the image is represented by its normal on the sphere. The figure represents the sphere where each point corresponds to a normal vector (Colorcode: X=Red, Y=Green, Z=Blue). From left to right: Sphere with perfect data; Sphere of a real image; classification of the previous data using our algorithm in the horizontal plane. Big dots represent VPs.

assumptions and with perfect data, normal vectors  $\mathbf{n} = (n_x, n_y, n_z)$  corresponding to the three different directions  $\mathbf{X}$ ,  $\mathbf{Y}$ ,  $\mathbf{Z}$  does not has its own component (i.e. line from the real world belonging to direction  $\mathbf{X}$ , has a normal vector  $\mathbf{n}$  whose component  $n_x = 0$ ), Fig. 1. However, often data is not perfect so it will suffer deviations from this configuration. The classification process for these normals is done as follows:

1) Under assumption **b**, image lines whose normal vectors has  $n_z$  component below a threshold (experimentally we find 0.2 is a good value) are automatically classified as vertical lines, and removed for next steps.

2) Suppress  $n_z$  component of remaining normals so every  $\mathbf{n} = (n_x, n_y, 0)$  will fall in a 2D plane, and using RANSAC we seek two orthogonal lines which minimize  $\frac{\text{error}}{\text{inliers}}$  (with number of *inliers* greater than a minimum). These lines will define the two horizontal main directions.

3) Image lines are labeled depending on the distance between its normal vector and one of the two main directions. It is remarkable that normal vectors whose component  $n_z \simeq 1$  are conflictive as they are conics which degenerate into circles and can not be properly classified, so it is better remove them to avoid errors.

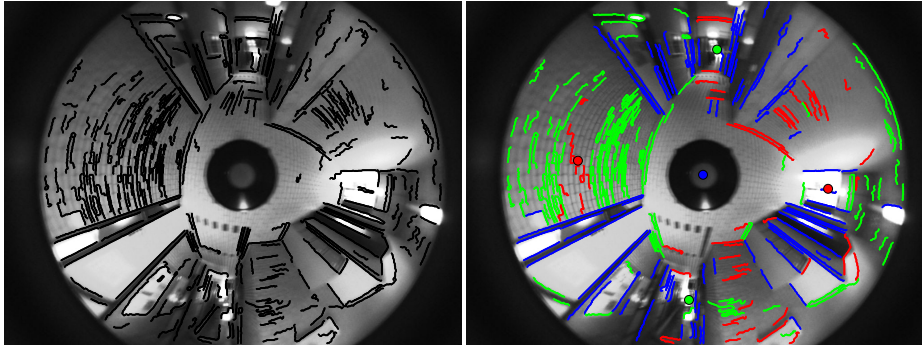
4) Finally, VPs are estimated as the points where the lines defining main directions cut the sphere at the hemisphere ( $Z = 0$ ), see Fig. 1.

### 3 Hierarchical Layout Hypothesis Method

Due to noise and imperfections of real images often there is not enough information to clearly define where the floor-wall boundary is, so with the extracted lines and a set of geometric constraints we must seek the best approach to find where these boundaries are. In order to do this we generate conics (possible boundaries) from a set of points belonging to the lines previously classified.

#### 3.1 Selection of Set of Points

The first factor to notice is that information obtained by vertical lines is more robust and less susceptible to noise than horizontal lines, which sometimes are difficult to classify



**Fig. 2.** Left: Lines extracted by Canny Edge detector after pruning step. Right: Same lines grouped in the 3 dominant directions according to our classification. Big dots represent VPs.

or are not well detected. Furthermore, studying typical images (see examples Fig. 7), we have noticed that in absence of objects most of vertical lines have their origin around the region that define floor-wall boundary, and if objects are present, they are standing over the floor and use to be close to the wall. So, unless these objects are placed all around the room, the origin of the lines that define them still being close to the desired boundary.

With these points of vertical lines we have to generate conics which will define possible floor-wall boundaries. In [9], it has been demonstrated how with a calibrated camera it is possible to define a conic in the image from only two points. If we apply the condition that every line in the image must pass through a vanishing point, just one point belonging to the floor-wall intersection is needed to define a conic in the image being a boundary.

Due to these 2 facts, let us denote as group  $G_Z$  the set of points composed by the closest point from each vertical to the center of the image. Additionally, we select a homogeneously distributed set of points from horizontal lines situated at the same height as the points in  $G_Z$ . Carrying out this selection for lines in  $X$  and  $Y$  direction, we obtain two more groups,  $G_X$  and  $G_Y$ , respectively. This is done in order to remove noisy horizontal segments, such as those found in objects, windows, doors,... and prevent them voting, Fig. 3(left).

### 3.2 Generation of Conics

Since we do not know which points of these groups are situated in the floor area, we apply RANSAC to identify the most voted conic, candidate to represent our desired boundary.

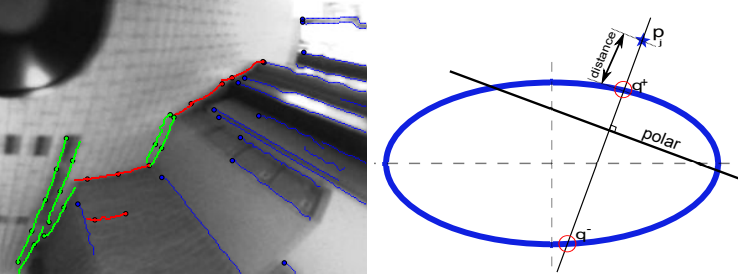
As we mentioned before, only two points are needed, one Vanishing Point and one point from the previous sets. Cross product between VP and each of these points  $\mathbf{p}_i$  generates a normal vector  $\mathbf{n}_i$ , which defines a conic  $\hat{\mathcal{O}}$  finally obtaining  $\hat{\mathcal{O}}$  after a projective transformation  $H_C$  [10].



$$\mathbf{n}_i = \begin{pmatrix} n_{i_x} \\ n_{i_y} \\ n_{i_z} \end{pmatrix} = \begin{pmatrix} VP_x^S \\ VP_y^S \\ VP_z^S \end{pmatrix} \times \begin{pmatrix} P_{i_x}^S \\ P_{i_y}^S \\ P_{i_z}^S \end{pmatrix} \quad (1)$$

$$\overline{\Omega}_i = \begin{bmatrix} n_{i_x}^2(1-\xi^2) - n_{i_z}^2\xi^2 & n_{i_x}n_{i_y}(1-\xi^2) & n_{i_x}n_{i_z} \\ n_{i_x}n_{i_y}(1-\xi^2) & n_{i_y}^2(1-\xi^2) - n_{i_z}^2\xi^2 & n_{i_y}n_{i_z} \\ n_{i_x}n_{i_z} & n_{i_y}n_{i_z} & n_{i_z}^2 \end{bmatrix} \quad (2)$$

$$\widehat{\Omega}_i = H_C^{-t} \overline{\Omega}_i H_C^{-1} \quad (3)$$



**Fig. 3.** Left: Selection of points as explained in Section 3.1. Right: Graphic explanation for distance measurement between point and conic in Section 3.2.

Now, distance between conic and every point  $\mathbf{p}_j$  is computed using an approximation [9] to [11]. We compute the polar line of a point  $\mathbf{p}_j$  in the conic  $\widehat{\Omega}$ , calculate the perpendicular specifying it lies over  $\mathbf{p}_j$ . This perpendicular line intersects the conic in two points  $\mathbf{q}^+$  and  $\mathbf{q}^-$ , the minimum Euclidean distance between  $\mathbf{p}_j$  and  $\mathbf{q}^+$  or  $\mathbf{q}^-$  corresponds to distance from point to conic, Fig. 3.

A new normal vector is estimated from the average of all points with minor distance than a threshold, and we iterate the whole process until its convergence (no more points are added). Points voting for this conic are removed from the list, and one of the remaining is chosen to generate a new conic, repeating the procedure and stopping when every point has been assigned.

### 3.3 Initial Boundaries Hypothesis

Computers cannot tell from a bunch of raw data how many walls a room is made of, but it is known that the most common indoor places are halls and rooms with similar shape to those shown in Fig. 4. All these geometrical shapes can be depicted by a central square with branches arising from all or some of its faces which at the same time must meet a geometric constraint: Parallel faces have to be one at each side of the imaginary line formed by joining their two corresponding VPs. This comes from the definition of vanishing point as the geometric place where parallel lines appear to converge.

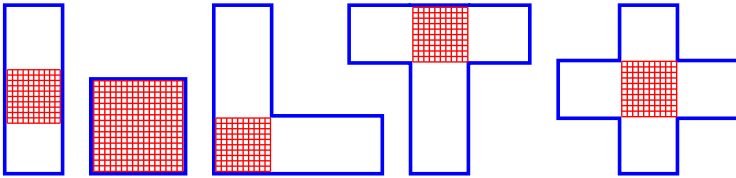
Due to this constraint, the searching algorithm to extract conics (Section 3.2) is executed for four different cases, in order to find the first four boundaries (better seen in Fig. 5):

- Boundaries 1 and 3: Are sought using points of  $G_Z$  and  $G_X$  at each sides of the imaginary line defined by  $VPs$  in direction  $X$  (Fig. 5(left)) .
- Boundaries 2 and 4: Are sought using points of  $G_Z$  and  $G_Y$  at both sides of the imaginary line defined by  $VPs$  in direction  $Y$  (Fig. 5(center)) .

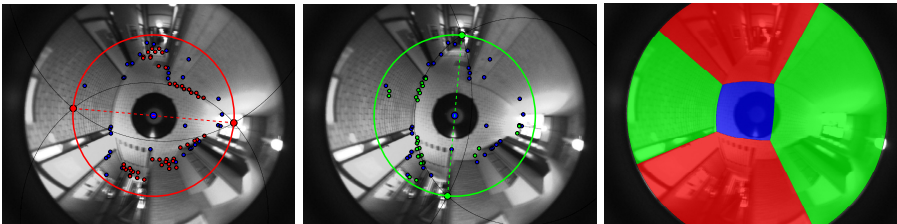
Another property is that the four vanishing points define a conic which corresponds to points situated at the same height as the camera, so every point falling within the conic might be on the floor and will be a possible candidate, while the ones out of the conic are automatically eliminated.

Remaining points are projected onto the sphere for each of the four cases previously defined, and we proceed to generate conics with these points.

Conics more voted are now selected as possible candidates and the nearest to the center of the image is chosen. We rather chose this closest line to other which might be more voted because it will have chance to be selected in the expansion process (Section 3.4), and if chosen now, it would be possible the loss of information. Once the four boundaries have been found, they are combined to conform walls and floor of our first hypothesis (Fig. 5).



**Fig. 4.** Most common room/hall shapes (top view). Red grid represents the basic square we are seeking in section 3.3



**Fig. 5.** First two images show points from groups  $G_Z$ ,  $G_X$  and  $G_Y$  under constraints exposed in section 3.3, where blue, red and green dots correspond to  $G_Z$ ,  $G_X$  and  $G_Y$  points respectively. Dashed red and green lines are the imaginary lines, going through the  $VPs$ , which divide the image in 2 parts. Finally, black conics represent the most voted boundaries for each case. Right image shows the result of combining those boundaries to generate the first hypothesis.

### 3.4 Hierarchical Expansion Process

Let us denote as  $B_1, B_2, B_3$  and  $B_4$  the four boundaries defined in previous section 3.3. The area between those and the end of the image defines four sectors. These sectors may correspond to actual walls or may exist the possibility they can be expanded, understanding *expand* as replacing the boundary  $B_i$  for others which enlarge the area of the first-hypothesis room layout. For each of these sectors we repeat the same method described in Section 3.3, obtaining a maximum of 3 new boundaries. Let them be  $B_i^L, B_i^M$  and  $B_i^R$  in clockwise order as shown in Fig. 6.

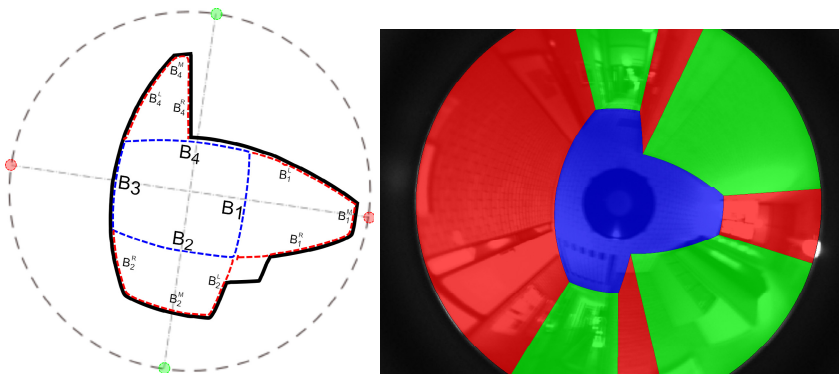
When looking for expansion three cases can happen:

- Enough data is available to define the 3 boundaries  $B_i^M, B_i^L$  and  $B_i^R$ ; so there will be expansion in the current sector.
- $B_i^M$  is very close to  $B_i$ , this means the most voted wall still being the same and will not be expansion.
- Data only allow us to find 1 or 2 boundaries. This last case can be originated for different situations and should be studied.

Third case is present when we lack of data, caused by lines not detected or by an occluded corner (Fig. 6(left)). Both cases imply expanding floor area but results are completely different, therefore care must be taken.

If missing boundary is a lateral ( $B_i^L, B_i^R$ ) or lateral plus middle ( $B_i^M$ ), and points from well-detected boundaries only fall at one side from the VP within the current sector, this is due to an occluded corner. Thus the missing border is defined as a radial line through the center of the image and the point, belonging to the well-detected boundary, whose angle is the closest to the angle defined by the VP.

On the other hand, if previous conditions are not satisfied, we assume some line was not detected, hence if the missing boundary is any of  $B_i^L$  or  $B_i^R$ , it will be defined as the resulting conic passing through its corresponding VP and the last point belonging



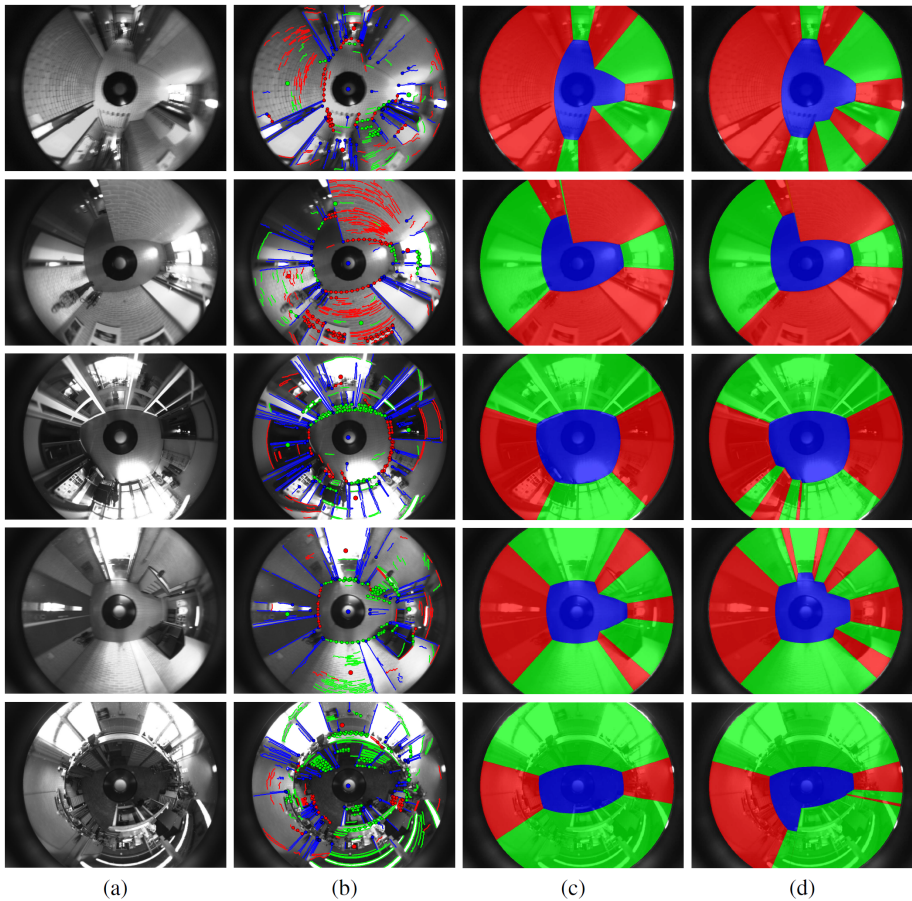
**Fig. 6.** Left: Synthetic example depicting the possible cases ( $B_1$  and  $B_2$  are expandable regions,  $B_3$  will not be expanded, and  $B_4$  corresponds to an occluded corner). Black line represents the actual room boundaries, first hypothesis in dashed blue, and final expansions in dashed red. Right: final result of a real example.

to  $B_i^M$ . By contrast, if the missing edge is  $B_i^M$ , we consider it should be no expansion except if there are a relevant number of voting points in the lateral boundaries  $B_i^L$  and  $B_i^R$ .

## 4 Results

Our experiments have been performed using Matlab, running at 3 sec per frame, and with the dataset COGNIRON composed of indoor images ( $768 \times 1024$ ) with a wide variety of rooms. These images were taken by a camera with a hyperbolic mirror spotted on a mobile robot. The calibration of the camera is also available online [12].

We show some of our results in different kind of indoor situations Fig. 7. First two examples correspond to T and L shape halls (like the ones shown in Fig. 4), walls are



**Fig. 7.** Examples of experimental results obtained for five different images. (a) Input images. (b) Line classification and extracted points which vote for boundary selection. (c) Output images by our algorithm. (d) Ground truth, manually labeled.

not too saturated with objects so the result is accurate. In this second example we also observe an occluded corner at the superior part of the image. Third picture is taken in a room where walls are made of glass (top and bottom of the image); due to these walls very bright areas appear in the scene, but we still achieve a good approximation of its structure.

Forth case shows a hall with a desk and a shelf, where our algorithm is able to recognize these obstacles. However, it does not detect the open door situated at the top part of the image, probably due to all the light going through it.

Last scene correspond to a room with many objects, colors are very dark, which makes difficult line extraction at some areas. At the same time, most of the longest detected lines fall over the objects, what might lead to misclassify wall-floor boundaries, but as we can see, we are still achieving good results.

Comparing results from our algorithm with their respective ground truth, we define as true positives (tp) the number of pixels both have in common, false positives (fp) the number of pixels identified as floor by our method but do not correspond to the floor in the ground truth, and false negatives (fn) the number of pixels identified as not floor when they result to be floor in the ground truth. With these values we compute **precision** ( $\frac{tp}{tp+fp}$ ), **recall** ( $\frac{tp}{tp+fn}$ ) and **F<sub>1</sub>** ( $\frac{2 \text{precision} \text{recall}}{\text{precision} + \text{recall}}$ ) for several images, Table 1.

**Table 1.** Performance values obtained for images of Fig. 7

	Image1	Image2	Image3	Image4	Image5
Precision	0.973	0.984	0.896	0.964	0.904
Recall	0.887	0.969	0.992	0.937	0.878
$F_1$	0.928	0.977	0.942	0.950	0.891

## 5 Conclusion and Future Work

We have proposed a new method to extract vanishing points from an omnidirectional image and to perform line classification. Since for the database of images the Z axis from camera and world reference were aligned, our proposal have been designed with this constraint, but we believe this can be extended for the general case where the alignment of axis is not known. We also have proposed a new simple and robust method for scene layout recovery and we have shown its performance in experimental results. This is useful in many applications, since knowing where floor-wall boundary are and the height of the camera, we can known exactly where every point of the scene is. Currently we are working into spread out this method for a whole sequence of images in order to improve accuracy in those images with possible misclassifications.

## References

1. Coughlan, J.M., Yuille, A.L.: Manhattan world: Compass direction from a single image by bayesian inference. In: Int. Conf. on Computer Vision, pp. 941–947 (1999)
2. Lee, D., Hebert, M., Kanade, T.: Geometric reasoning for single image structure recovery. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2136–2143 (June 2009)

3. Hedau, V., Hoiem, D., Forsyth, D.: Recovering the spatial layout of cluttered rooms. In: IEEE International Conference on Computer Vision, pp. 1849–1856 (2009)
4. Tsai, G., Xu, C., Liu, J., Kuipers, B.: Real-time indoor scene understanding using bayesian filtering with motion cues. In: ICCV, pp. 121–128 (2011)
5. Ozisik, N.D., López-Nicolás, G., Guerrero, J.J.: Scene structure recovery from a single omnidirectional image. In: ICCV Workshops, pp. 359–366 (2011)
6. Bazin, J.C., Kweon, I., Demonceaux, C., Vasseur, P.: A robust top-down approach for rotation estimation and vanishing points extraction by catadioptric vision in urban environment. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 346–353 (2008)
7. Bermudez, J., Puig, L., Guerrero, J.J.: Line extraction in central hyper-catadioptric systems. In: OMNIVIS (2010)
8. 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)
9. Bermudez-Cameo, J., Puig, L., Guerrero, J.J.: Hypercatadioptric line images for 3d orientation and image rectification. *Robotics and Autonomous Systems* 60(6), 755–768 (2012)
10. Barreto, J.: General central projection systems: Modeling, calibration and visual servoing. Ph.D. dissertation (2003)
11. Sturm, P., Gargallo, P.: Conic fitting using the geometric distance. In: Proceedings of the Asian Conference on Computer Vision, Tokyo, Japan, pp. 784–795 (2007)
12. Zivkovic, Z., Booij, O., Krose, B.: From images to rooms. *Robotics and Autonomous Systems* 55(5), 411–418 (2007)

# Development and Experiences of an Autonomous Vehicle for High-Speed Navigation and Obstacle Avoidance\*

Jee-Hwan Ryu<sup>1</sup>, Dmitriy Ogay<sup>2</sup>, Sergey Bulavintsev<sup>3</sup>,  
Hyuk Kim<sup>3</sup>, and Jang-Sik Park<sup>3</sup>

<sup>1</sup> School of Mechanical Engineering, Korea University of Technology and Education,  
Cheonan, South Korea  
jhryu@kut.ac.kr

<sup>2</sup> School of Computer Science and Engineering, Korea University of Technology  
and Education, Cheonan, South Korea  
cactus@kut.ac.kr

<sup>3</sup> School of Mechanical Engineering, Korea University of Technology and Education,  
Cheonan, South Korea  
{sergey, perseus, ganggai}@kut.ac.kr

**Abstract.** This paper introduces the autonomous vehicle Pharos, which participated in the 2010 Autonomous Vehicle Competition organized by Hyundai-Kia motors. Pharos was developed for high-speed on/off-road unmanned driving avoiding diverse patterns of obstacles. For the high speed traveling up to 60 Km/h, long range terrain perception, real-time path planning and high speed vehicle motion control algorithms are developed. This paper describes the major hardware and software components of our vehicle.

## 1 Introduction

The first autonomous vehicle competition in South Korea organized by Hyundai-Kia Motors took place on November, 2011 [1]. The mission of the competition was unmanned traveling about 4 Km on/off road clearing 7 different patterns of obstacles. Lane keeping and stop within 1 m of crosswalk were also included. Eleven universities in South Korea were participated in the main competition out of 20 initial applicants, and “Pharos”, developed by our team, finished the course in 8 min 52 sec clearing all the missions except crosswalk, which got 5 min penalty, and Pharos took 4<sup>th</sup> place.

Recently there have been many research activities in autonomous vehicle area. Especially, DARPA Grand Challenge [1], [8] and Urban Challenge [10] made a big progress on the area of autonomous vehicle. However, there are still many open issues for high speed unmanned traveling such as long range terrain perception [9], real-time obstacle avoidance and trajectory planning [3], [6], and high speed vehicle motion control [5] et al.

In this paper, autonomous vehicle based on real Sport Utility Vehicle (SUV) is introduced motivated by the competition. The main challenging issue in the development of

---

\* This work was supported partly by the R&D program of the Korea Ministry of Knowledge and Economy (MKE) and the Korea Institute for Advancement of Technology (KIAT). (Project: 3D Perception and Robot Navigation Technology for Unstructured Environments, M002300090).

the vehicle was to build a reliable system, able to traveling high speed up to 60 Km/h through on-road and unstructured off-road while avoiding different types of obstacles. To satisfy these requirements, new methods were developed and extended based on existing methods in the field of autonomous navigation such as long-range obstacle detection and mapping, real-time collision avoidance and trajectory planning, and stable vehicle control on slippery and rugged terrain.

## 2 Hardware Design

Fig. 1 shows the outlook of Pharos at the competition. Pharos is based on a Gasoline-powered Hyundai Santafe CM. Reinforced front bumper has been installed to protect the vehicle from the environmental impact.



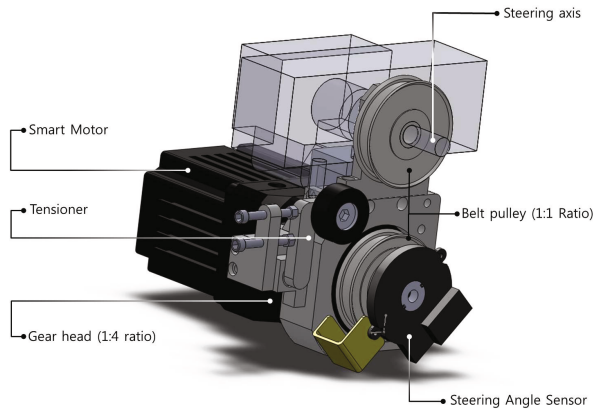
**Fig. 1.** Pharos was standing at the finishing line on the competition track

A geared DC motor is attached to the steering column to allow electronic steering control. Required torque and rotational speed was estimated through the experimental analysis. We found that 4.41 Nm continuous torque with 3000 rpm was required for controlling the steering column. Smart motor from Animatic Co. with 1:4 gear ratio was selected as a steering motor. With this motor, 5.8 Nm continuous torque with 4000 rpm can be achieved at 24V, which is sufficient to control the steering column.

Fig. 2 shows the assembled 3D CAD model of the steering actuation mechanism. one-to-one pulley powered transmission mechanism was used, and steering angle sensor was moved to the motor output axis to measure the absolute steering angle.

To motorize braking system, another geared DC motor is installed near by the braking pedal. Through the initial experiment, we found that 60 mm travel distance with 8 N pressing force is required max. for controlling the breaking system. one-to-five gear ratio was used to give enough force. It also allows fine position control of the breaking pedal.

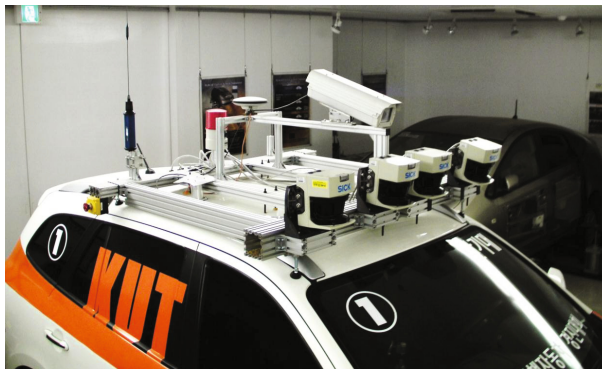




**Fig. 2.** Assembled 3D CAD model of the steering actuation mechanism

Throttle is also modified to be controlled electronically through APS signal modification. APS signal is modified to control the throttle. From 0.8 V to 5 V was given for idling to maximum acceleration.

Vehicle data, such as steering angle, vehicle speed, APS signal and et al. are automatically sensed and communicated to the computer system through Ethernet interface.



**Fig. 3.** View of custom made roof rack with sensors

Fig. 3 shows the custom-made roof rack. Most of the sensors are installed on it. Four SICK laser scanners are installed on the roof rack since it can provide best visibility of the terrain. All scanners are facing forward along the driving direction of the vehicle, but with slightly different angles. A single CMOS camera with housing is also installed on the roof rack for crosswalk detection. GPS receiver, RF modem and a radio antenna for E-stop are also installed on the roof rack. The E-stop system is provided by Hyundai-Kia motors for allowing the chasing vehicle safely stop the vehicle in emergent situation



**Fig. 4.** Controllers and communication system in the trunk of the vehicle

with wireless link. Three manual E-stop buttons are installed also on the roof rack and backside of trunk.

Pharos's controllers and communication system are located inside of trunk as shown in Fig. 4. Six Compact PCI, NI-CompactRIO, Gigabit Ethernet switches and various types of interfaces for physical sensors and actuators are installed on a shock-mounted rack. Custom-made power system with backup batteries are installed underneath of the rack for supplying power to all the electrical components. A six degree-of-freedom IMU is rigidly attached to the vehicle frame underneath the vehicle roof.

The added instrument is required approximately 2.4 KW in addition. Two 12V alternators are installed in addition to supply power, and it provides 24V, 3.4 KW. Therefore, Pharos can even run all the system more than an hour without recharging.

### 3 Software Architecture

#### 3.1 Overview

The main principles that we wanted to follow when developing the system architecture were: reliability of a system, ability to distribute software over several independent computers, easy configuration, simulation and development. To reach these requirements we have chosen an architecture, where each module was an independent program that could be run stand alone in a separate process of an operating system. Communication between modules is done through publish-subscribe mechanism.

There is no central process, which encapsulates modules, as it could reduce reliability of the system, if such a process dies. All communication is done through a lightweight messaging server, which supports connectivity through network protocols. It allowed us to easily distribute modules over several computers, and even gave opportunity to run different modules on different operating systems.

This approach also makes development of the modules simpler, because each module is an independent program and can be run stand alone in an environment, where all

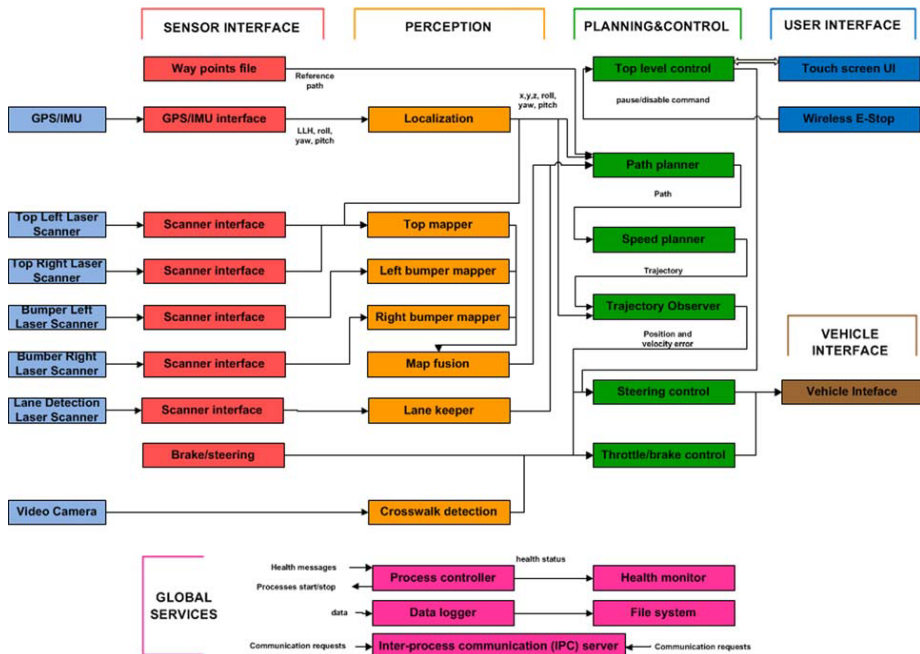


Fig. 5. Overview of the software architecture

inputs and outputs can be stubbed. For example we could substitute actual data from the sensors by data from file for simulation.

In a running integrated system all blocks are put on watchdogs, so they are restarted if fail, or they can be controlled remotely from the user interface or health monitoring system.

### 3.2 System Components

Our system has multiple processing layers, when data flow from sensors through consecutive layers, until it reaches steering and braking-acceleration actuators. These layers are: sensor interface, perception layer, planning layer, and control layer. Each layer consists of several modules as shown in Fig. 5.

Sensor interface layer consists of modules that receive data directly from sensors, using sensor protocols, and then publishes the data in a format used inside our system. They run on frequency determined by corresponding sensors, which are 75Hz for laser scanners and 100 Hz for GPS/IMU.

Perception layer has several independent parts that are: localization for estimating vehicle state, laser mapping for generating obstacle occupancy grid map from laser scanner data, and vision mapping for detecting road markings. Data to the path planner are sent at 10 Hz frequency.

Planning layer consists of path planning module that uses 2D obstacle map to generate traversable path, speed planning layer that plans speed taking into account current vehicle speed, speed limits imposed by competition requirements, and shape of the generated path.

Control layer consists of a Trajectory Observer, which calculates difference between planned path and current vehicle position and sends it to the motion control module, which is run in a separate real-time system. Unlike path planner, which is non-deterministic, Trajectory Observer is run at 100 Hz frequency, monitors vehicle position and stops vehicle if no data from path planner come, which can occur in case of a process failure.

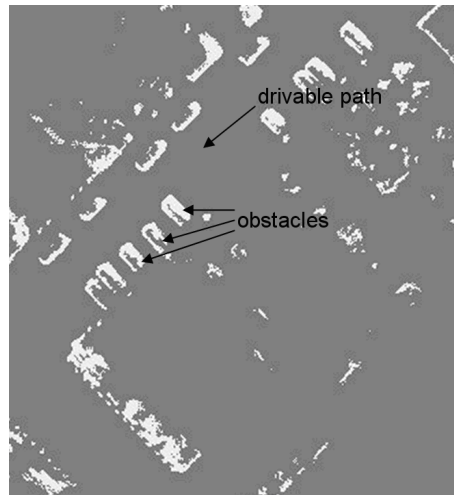
All modules also generate health monitoring messages, which are listened by a health monitoring block.

## 4 Obstacle Detection and Mapping

To make safely avoid obstacles, Pharos must be able to create a precise terrain map with sufficient range of view for undertaking appropriate actions. The map should contains accurate position of the obstacles to maintain the movement of the vehicle even in the narrow path, comparable with the width of the vehicle. Detection range of the obstacles is considered as well since that faster speed of the vehicle results in the longer perceiving distance. To satisfy these main criteria, medium range laser scanners were chosen. It has 75 Hz update rate with accuracy 1cm on the 30 m distance. Laser scanner system of the Pharos consists of three laser scanners, mounted on the roof, tilted downward and two scanners, installed on the front bumper with 45 degree yaw angle with respect to the vehicle to provide wider area of view. Different obstacle classification algorithms are used for scanners installed on the roof and on the bumper. Each laser scanner acquires distance information in its own local coordinate frame. By using estimated position and orientation of the vehicle, data from scanners are transformed into points in global coordinate frame and represent 3D point cloud. Each point in 3D point cloud can be described as  $(X_m^i, Y_m^i, Z_m^i)$  where  $m$  is the index of the individual scanner and  $i$  is the index of each measurement in one scan.

Deterministic algorithm is used for the roof laser scanners because it has shown more reliable performance compared to other algorithms that had been tested. The classification method is based on the occupancy grid map [2]. The map represents 2D map consists of cells, and each cell contains position information and one of the possible conditions: occupied or unknown. The size of the cells is decided considering maximum resolution at the distance 30 m. To classify a cell as obstacle, two nearby points in 3D point cloud must be found and their vertical distance must satisfy condition  $|Z_k^i - Z_l^j| > \epsilon$ , where  $i$  and  $j$  are indices of two closest points in 3D cloud point from two different laser scanners and  $\epsilon$  is the threshold, assumed to being obstacle. If no such points can be found or no value is acquired, this cell is assigned as unknown. Unknown zone considered as drivable. Fig. 6 shows the occupancy grid map, drawn in our campus. Gray color means unknown area or drivable and white color represents obstacles.

The advantage of this algorithm is that the measurements from two different scanners are used to detect obstacles. This excludes possibility to detect fake obstacles due to the



**Fig. 6.** An example of occupancy grid map

error in the orientation estimation of the vehicle. In addition, it is possible to make six combinations when three scanners are used, therefore reliability of the algorithm can be increased. On other hand, algorithm can be possibly less reliable in case of fault in laser scanner hardware. Bumper laser scanners use different way for obstacle classification. Those are mounted on some known height relate to the ground and scans in horizontal plane. Cell is assigned to be an obstacle if points with same  $X$  and  $Y$  location satisfies condition  $\sum_{j=1}^n p_j (Z_j > |h - \delta|) \cdot w > 0.5$ , where  $j$  is the time index for the series of height measurements acquired for same cell,  $w$  is weight of the possibility that point is obstacle and  $\delta$  is vertical distance that determines size of the obstacle. If same point has been classified as obstacle several times, likelihood of the obstacle presence in this location increases and obstacle is detected.

In practice both algorithms have shown reliable performance and allow our vehicle avoiding obstacles on the distance around 30 m ahead with speed up to 60 km/h.

## 5 Road Boundary Estimation

One of the problems that can occur during autonomous driving is reference path drifting, due to the shifting of the base coordinates of the DGPS, which varies depend on the various conditions. To avoid it, we develop algorithm, which helps to determine the position of the vehicle with respect to the road boundary and removes the effect of the drifting. On the asphalt road, where static obstacles usually aren't encountered, vehicle has to keep traveling right side of the road. Therefore our task simply was to define lateral distance between car and one of the road sides. To distinguish actual road from off-road, we use reflection properties of the road. Flat asphalt surfaces have lower reflection value than rough terrain. Based on this reflection, occupancy grid can be built. The cell is assumed as non-drivable if reflection value at that location higher than certain limit. Accumulated average reflection value is used. The reference path assumed

to be parallel to the road side, but usually lateral offset changes over the time and it involves the unstable behavior of the vehicle when unfiltered offset is used to correct car position. To find lateral distance, one-dimensional low-pass Kalman filter is used. The state of the Kalman filter is distance between reference path and the road boundary. The Kalman filter searches for largest offset along discrete search pattern orthogonal to the reference path. The Fig. 7 shows an example of discrete search pattern. In that case the road boundaries change slowly. Sudden increases in reflection, for example when obstacles appear which normally have higher reflection value, result in small affect in the boundary road estimation. Output from Kalman filter defines offset which is used by path planner, which necessary to prevent road boundary crossing and removing influence of the DGPS drift.

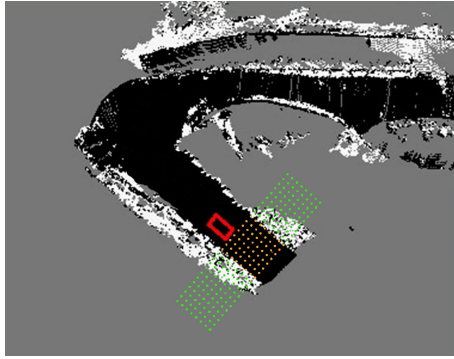


Fig. 7. An example of road boundary estimation

## 6 Path and Speed Planning

### 6.1 Mission

The main mission that car should accomplish is the following of a pre-recorded reference path. Artificial and natural obstacles can be present on a reference path, so vehicle should avoid them, and also edges of a road can be natural obstacle, because of GPS localization drifting.

The reference path is stored in a text file. Before being given as an input to the path planner, the reference path is smoothed, and such parameters as tangential vector and curvature of the path are calculated for each reference point.

### 6.2 Path Planner

For our implementation, we have chosen RRT [7] based approach, as it can drive vehicle through a very narrow path and can take kinematic constraints of a vehicle into account. But computation speed is a challenge for the development of real-time algorithms [4]. We wanted our vehicle to response faster to the environment changes, because the range

of car's perception was limited. And fast path planner response would allow us to increase car's speed.

To cope with those challenges we made some modifications to original RRT algorithm and also developed a new method to increase computing time efficiency of the path planner. It should be mentioned that our path planner is run in a single thread and has re-planning time about 100 ms, which is enough to drive vehicle at speeds up to 60 km/h, given car's perception range.

Original RRT algorithm is modified in several ways, including biased sampling, which is made along lines perpendicular to the reference way points. The state consists of three components, which are:  $x$  and  $y$  coordinates, and a heading angle of a vehicle.

To plan kinematic behavior of a vehicle we assumed that vehicle travels in short arcs. This let us use curvature of a planned path to calculate lateral acceleration and determine maximum speed, given limited lateral acceleration, which is the safe speed. Then we use time component based on this safe speed in our cost function, which measures distance between states in RRT. The main optimization criteria are fast and smooth motion. Fig. 8 shows an example of the path planning inside of an artificially made narrow tunnel. The area beyond the walls on Fig. 8(a) is valid for sampling in original RRT algorithm, but that area is not reachable by a car. Fig. 8(b) shows that the proposed planner allows us to achieve computing time efficiency by not propagating path to the points that are potentially not reachable.

Before planned path is sent as output it is run through a Savitzky-Golay low-pass filter at first to be smoothed, but mainly to calculate tangential vectors and curvatures of the path at every point.

### 6.3 Speed Planner

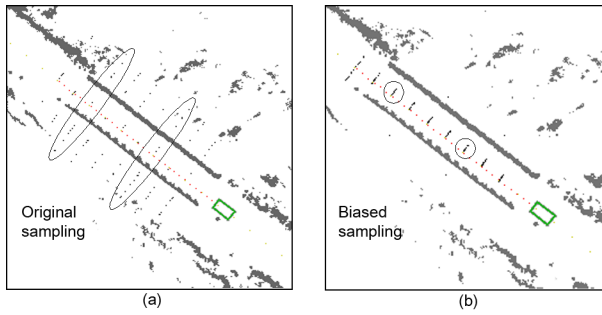
The main task for a speed planner is to follow maximum safe speed, while taking into account speed limit zones, that are recorded together with reference path, and maximum braking deceleration to avoid collisions if there is some sudden change in a safe speed. As it was mentioned before, safe speed is determined by a curvature of a planned path. And, if there is a sudden change in a path curvature, speed planner should react in advance.

### 6.4 Trajectory Observer

Trajectory Observer is a special block, running at a high frequency which measures difference between current state of a vehicle and planned path. Then it sends the data to the control block. There were two reasons to introduce this block. At first it is an interface between the whole system run on a general purpose operating system, which is linux in our case and a real time system. At second it improves safety of a vehicle, as it is deterministic and does not have delays or failures like path planner.

## 7 Real-Time Vehicle Motion Control

Once the errors to the reference trajectory of the vehicle is given from the trajectory observer, the motion controller produces appropriate steering, throttle and brake



**Fig. 8.** Comparison of sampling distribution (black dots) of non-optimized algorithm (a), and optimized algorithm (b). Passing artificially made narrow tunnel

command to achieve the given trajectory. This issue will be described in two parts: steering control and speed control.

## 7.1 Steering Control

Lateral offset, angle difference and curvature of the reference path are given to the steering controller as inputs. The steering controller gives steering commands to the steering motor at a rate of 100 Hz. The basic steering angle control law is very similar to the one in Stanley [8] except look ahead consideration. Simple kinematic based PD feedback controller is used for compensating Lateral offset, which measures the lateral distance of the center of the vehicle's front wheels from the nearest point on the reference trajectory, and angle difference, which is the orientation of the nearest path segment, measured relative to the vehicle's own orientation. Basically, compensating angle difference only allows our vehicle to track the reference trajectory at some level, however without lateral offset consideration it can not compensate steady state error. To cope with the continuous change of the reference trajectory, curvature of the trajectory at an adaptive look ahead point was used as a feed forward control input. Each gains were scheduled to the vehicle velocity, and has proven stable trajectory following on terrain from pavement to off-road, and trajectories with tight curvature.

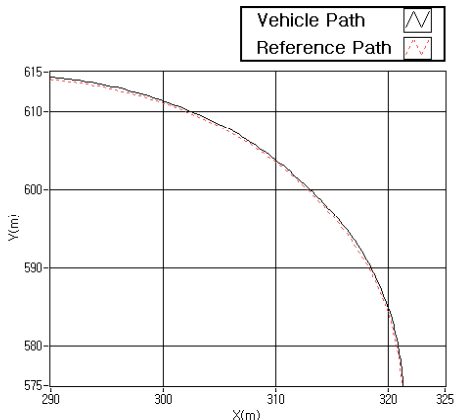
Fig. 9 shows steering performance while Pharos was making a corner of 40 m radius curved road. Steering controller maintained within 30 cm lateral offset around 60 km/h.

## 7.2 Speed Control

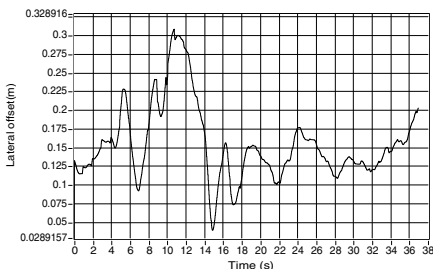
Speed planner and cross walk detection module give speed command to a low-level speed controller. The low-level speed controller translates the speed commands from each modules into actual throttle and brake commands. The minimum of the two recommended speed is always used.

Once the desired speed is determined, low-level speed controller control the brake level and throttle level exclusively. It treats the break and throttle as two opposing single-acting actuators that produce longitudinal force on the vehicle. The controller



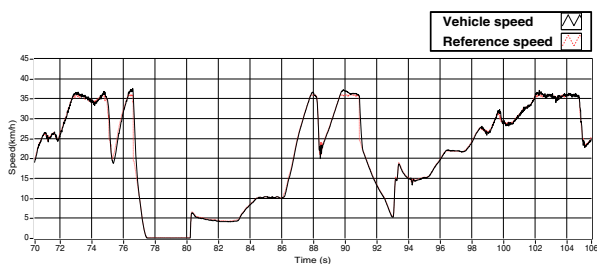


(a) Tracking performance of the steering controller



(b) Lateral offset error of the vehicle

**Fig. 9.** Steering performance of the vehicle on 40m radius curved road



**Fig. 10.** Control performance of the speed controller

computes weighted sum plus weighted integration of the speed error. Based on the sign of the computed value, either brake or throttle level is controlled. By considering dead-zone, the controller is able to avoid the chattering behavior. Fig. 10 shows the control performance of the proposed speed controller. It tracks the desired speed very well.

## 8 Discussion

This paper provides an overview of the autonomous vehicle Pharos, developed to participate in the 2010 Autonomous Vehicle Competition in South Korea.

From a broad perspective of view, Pharos's control software mirrors common methodology in the area of autonomous vehicle. However many of the individual modules have developed to achieve the high speed unmanned driving. Long range terrain perception and obstacle detection algorithm, real-time trajectory planning and obstacle avoidance algorithm, and high speed vehicle motion control method are developed and integrated all together. All the developed algorithms are extensively tested in the field to prove the reliability.

Even though our vehicle could successfully finish the race, there are many issues to improve our vehicle further. Only static environment was considered in perception. Pharos is unable to properly interact with moving objects. Perception range was limited within 30 m, which limited the maximum speed of our vehicle. In some case, to avoid an obstacle it was necessary to move up to 6 m to lateral direction from the reference path. If the perception range is limited, vehicle must reduce the approaching speed to avoid an obstacle which has large lateral offset from the reference path. We expect our methodology can allow us to perceive longer range than 30 m.

**Acknowledgments.** The authors gratefully acknowledge the contribution of Pharos Racing Team for their passion to build our vehicle.

## References

1. Buehler, M., Iagnemma, K., Singh, S.: The 2005 DARPA grand challenge: the great robot race. Springer Tracts in Advanced Robotics, vol. 36. Springer, Berlin (2007)
2. Elfes, A.: Using occupancy grids for mobile robot perception and navigation. *Computer* 22(6), 46–57 (1989)
3. Dolgov, D., Thrun, S., Montemerlo, M., Diebel, J.: Practical Search Techniques in Path Planning for Autonomous Driving. In: in Proceedings of the First International Symposium on Search Techniques in Artificial Intelligence and Robotics (2008)
4. Frazzoli, E., Dahleh, M.A., Feron, E.: Real-time motion planning for agile autonomous vehicles. *AIAA Journal of Guidance and Control* 25(1), 116–129 (2002)
5. Kelly, A.: A partial analysis of the high speed autonomous navigation problem. Project report of Perception for Outdoor Navigation and “Unmanned Ground Vehicle System. Carnegie Mellon University (1994)
6. Kuwata, Y., Fiore, G.A., Teo, J., Frazzoli, E., How, J.P.: Motion planning for urban driving using RRT. In: Proc. IROS, pp. 1681–1686 (2008)
7. LaValle, S.M., Kuffner, J.J.: Randomized kinodynamic planning. *International Journal of Robotics Research* 20(5), 378–400 (2001)
8. Thrun, S., et al.: Stanley: The Robot that Won the DARPA Grand Challenge. *Journal of Field Robotics* 23(9), 661–692 (2006)
9. Urmson, C., et al.: A robust approach to high-speed navigation for unrehearsed desert terrain. *Journal of Field Robotics* 23(8), 427–508 (2006)
10. Urmson, C., et al.: Autonomous driving in urban environments: Boss and the Urban Challenge. *Journal of Field Robotics* 25(8), 425–466 (2008)
11. <http://www.hyundai-ngv.com/techcontest/> (in Korean)

# RoboEarth Action Recipe Execution

Daniel Di Marco<sup>1</sup>, Moritz Tenorth<sup>2</sup>, Kai Häussermann<sup>1</sup>,  
Oliver Zweigle<sup>1</sup>, and Paul Levi<sup>1</sup>

<sup>1</sup> University of Stuttgart, Department Image Understanding, Universitätsstr. 38,  
70563 Stuttgart, Germany

{dimarco,haeussermann,zweigle,levi}@ipvs.uni-stuttgart.de

<sup>2</sup> Technische Universität München, Intelligent Autonomous Systems Group, Karlstr. 45,  
80333 München, Germany  
tenorth@cs.tum.edu

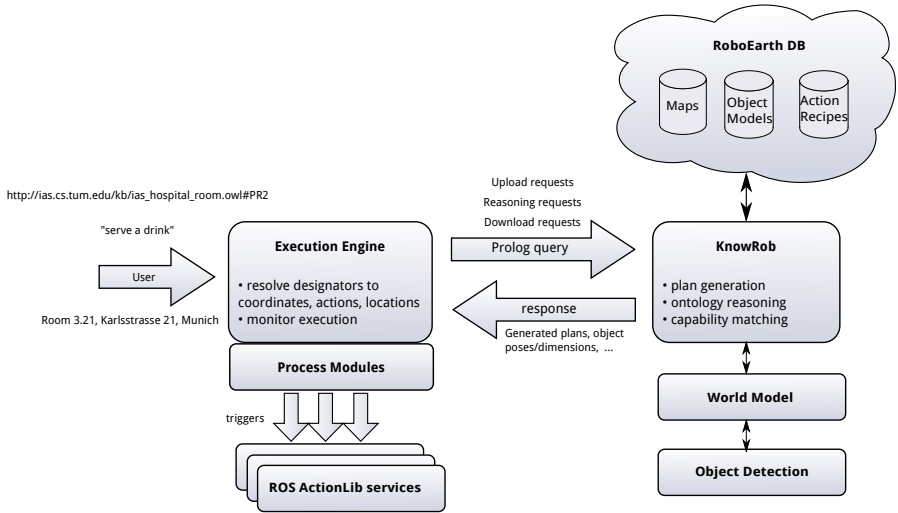
**Abstract.** The ability of reusing existing task execution plans is an important step towards autonomous behavior. Today, the reuse of sophisticated services allowing robots to act autonomously is usually limited to identical robot platforms and to very similar application scenarios. The approach presented in this paper proposes a way to mitigate this limitation by storing and reusing task plans on a globally accessible database. We describe the task execution engine for the RoboEarth project [28] to demonstrate its ability to execute tasks in a flexible and reliable way.

## 1 Introduction

Robot control and execution planning is a challenging task, particularly in complex indoor household environments. Autonomous service robots have to achieve high-level task goals while performing in a reactive and flexible way. To allow a robust and reliable high-level task execution, autonomous robots have to detect unexpected situations, handle exceptions and manage their limited resources by adapting their behavior accordingly to cope with issues of dynamics and uncertainty.

Therefore high-level capabilities, such as knowledge-processing and reasoning, are essential for task planning and decision making processes. Pre-programming a plan for a given robot to solve a task is usually a complex and tedious act and most of the time, plans once written cannot be used again under different premises. Even if a plan is programmed for one robot in a robust manner it is usually impossible to copy this plan on another robot type and achieve the same success. To cope with these typical issues, RoboEarth was established. The prime objective of RoboEarth [26] is to enable robots to reuse previously generated data; i.e. maps of the environment, object models or task execution plans. The most important component in the RoboEarth project thus is a distributed database [21] for storing and processing this data. The data can be generated from participating robots, humans or processor programs running on the database itself. For the interpretation and execution of abstract task descriptions we developed an execution engine which is able to serialize and trigger the execution of all commands, supervise them, cover failure states and handle incoming events.

The focus of this work is to describe the task execution engine of the RoboEarth project and to demonstrate its ability to execute tasks in a flexible and reliable way.



**Fig. 1.** Interaction between RoboEarth components, as seen from the execution engine's point of view

The remainder of the paper is organized as follows: first, a brief overview of the related work in the domain of plan-based robot control and decision-making is given in section 2. The current architecture for the RoboEarth demonstrator implementation is presented briefly in section 3. In the following section 4.1 the notion of action recipes is described in detail. The process for selecting an action recipe is detailed in section 5. Afterwards we describe the chosen language for implementing execution plans along with the necessary modifications in Section 6. Section 7 presents some experimental results. A discussion in Section 8 analyzes and sums up the contributions of this paper.

## 2 Related Work

Within the last few years several applications of autonomous service robots and demonstrations of plan-based control have been developed and successfully evaluated in real world environments. With respect to the implementation of a reliable plan-based high-level controller on autonomous mobile robots the work of [22,21,18] should be mentioned. All mentioned approaches so far have in common that the robot system will not be given the detailed sequence of commands to execute, but a list of goals which have to be achieved instead. During the run, the robot will generate a plan to accomplish these goals, handle unexpected situations, manage failures and thus execute the plan in a robust manner. The approach developed in [11] has the objective to control and manage a group of autonomous mobile robots for transshipment tasks in a decentralized and cooperative way. Further AI-based cooperative systems were proposed by [5,8,7,6].

In the plan-language and -execution domain several sophisticated systems have been developed. One of the first situation-driven interpretations of plans was proposed in RAP [10] to enable reactive planning and execution of *sketchy* plans. A descendant of

RAP is the language Reactive Plan Language (RPL) which was designed by [16] to incorporate a richer set of control constructs and to allow the creation of reactive plans for robotic agents. Further plan execution systems used by autonomous robots are the Reactive Model-based Programming Language (RMPL) [27] and the Executive Support Language (ESL) [12].

The execution system of the RoboEarth project builds upon the CRAM planning language [4], a sophisticated plan language that shares several concepts with RPL, but its focus lies on the plan execution on real physical robots, e.g. providing fast feedback-loops. In [3], the authors demonstrate the system by using a robot which downloads and parses a set of natural language instructions from a web page and translates these into a sketchy CRAM plan.

In contrast to the works mentioned above, this paper focuses on describing an approach to sharing task execution plans among different robot types. The instructions to be downloaded (called *action recipes*) along with object models, maps and other requirements are stored in a well-defined, computer-readable way.

### 3 System Overview

The overall architecture is shown in Figure 1. All data for RoboEarth is stored by a cloud computing database based on Apache Hadoop [17]. The implementation is described in [21]. The data stored in the database includes action recipes, object detection models, navigation maps, etc. Every file of this data is annotated with an OWL description to allow semantic and taxonomic queries. The execution engine takes textual commands and a description of the robot and environment from the user and poses requests for information like plans and object positions to the knowledge processing framework KnowRob. KnowRob is responsible for the knowledge processing (section 4). It can access both the world model (for answering object positions queries) and the global database. To generate a plan, it queries the database and retrieves a matching, symbolic task plan to the execution engine. During execution, the execution engine resolves symbolic plan annotations (*designators*) by querying the KnowRob component. Process modules are basically an interface for basic action primitives, in the given example implemented as ROS actionlib services.

Assume a scenario where a robot is commanded to find a mug in order to provide a human with a drink. In this case it is possible to query the RoboEarth database for all object models that represent an object of type mug and its specializations (e.g. coffee mug). Should this inquiry fail or not give the desired results, it is also possible to query for all models representing an object that can store liquids.

### 4 Representations for Exchanging Knowledge

The knowledge processing framework KnowRob [23] is used as interlingua to exchange knowledge between different components of the RoboEarth system. When the execution engine queries for a recipe, KnowRob searches the RoboEarth database, downloads recipes, checks for missing components (such as environment maps or object models

```

Class: MoveBaseToHandoverPose
EquivalentTo:
  (knowrob:toLocation some
   robotPose-handover1)
SubClassOf:
  roboearth:Translation-LocationChange
...

Individual: ServeADrinkOrder10
Types:
  knowrob:PartialOrdering-Strict
Facts:
  knowrob:occursBeforeInOrdering
    MoveBaseToGraspPose,
  knowrob:occursAfterInOrdering
    GraspBottle

```

**Fig. 2.** Action Recipe example: Subaction and partial ordering specification (Manchester Syntax, [13])

that are required for the task), and triggers their download. During runtime, KnowRob serves as a semantic interface to the object positions provided by the world model.

In the current implementation, the KnowRob knowledge base is running locally on each robot. This comes at the cost of increased processing power and memory requirements on the robot, but has the advantage of the RoboEarth framework still providing basic functionality without having access to the Internet.

## 4.1 Action Recipes

Action recipes describe in a hardware-agnostic way how tasks are to be executed by a robot and are thus a kind of generalized task specification. Recipes are stored in description logic in a language built upon the W3C-standardized Web Ontology Language [25] and the KnowRob knowledge base [23]. The language itself is described in [24]. An excerpt from an exemplary action recipe is given in Figure 2.

This OWL-based language is optimized for automated reasoning about the requirements that need to be fulfilled in order to execute the action on a given robot platform, and for the description of actions, objects, environment models and robot capabilities in a single, coherent format. The following two sections will introduce some of the key concepts of this language. Before execution, the recipes are translated into the CREAM Plan Language CPL, which supports better execution monitoring, failure handling, and grounding of abstract object descriptions. This language will be described in Section 6.

Figure 2 gives examples of action specifications in RoboEarth action recipes, describing the action to move to a position for handing an object to a patient (*MoveBaseToHandoverPose*) as a specialization of an intentional location change (*Translation-LocationChange*). Task specifications are divided in sub-tasks that are partially ordered, thus allowing descriptions of both sequential and parallel task execution order.

```

Individual: Bottle1
Types: knowrob:DrinkingBottle
Individual: Bed1
Types: knowrob:Bed-PieceOfFurniture

Individual: handPose-handover1
Types: knowrob:Point3D
Facts: knowrob:aboveOf Bed1

```

**Fig. 3.** Description of two objects and one location for the example use case

## 4.2 Objects and Locations

In order to fully specify a task, it is not only necessary to describe the actions to execute, but also the objects and locations that the actions will interact with.

Pictured in Figure 3 is the corresponding description for some of the objects and locations used in the previously given example plan. *DrinkingBottle*, *Bed-PieceOfFurniture* and *Point2D* are the names for OWL classes defined in the RoboEarth ontology.

## 5 Reasoning about Robot Capabilities

To show the essential need of capability matching, we start with a typical scenario. Assume the RoboEarth database stores task descriptions for very different robot platforms. Assume further that some of the stored recipes were originally intended for e.g. Willow Garage's PR2 service robot and others were provided for a Roomba vacuum cleaner robot (i.e. a robot without an arm and gripper). Obviously, in this case there are recipes in the database that the Roomba robot cannot use, no matter how generic and platform agnostic these are formulated.

Thus two key features are required:

1. a way for describing the capabilities of different robot hardware
2. a method for selecting recipes from the database which are usable for a robot platform

These two points will be addressed in the following subsections.

### 5.1 Specifying Capability Requirements of an Actions

The Unified Robot Description Format (URDF) [11] has been adopted by the Robot Operating System [19] for describing a robot platform by its joints and links. This format is used within the ROS framework for visualizing robots and for configuring motion planners.

To decide whether a given recipe can be executed by a given robot, enumerating the joints and links is not enough [15]. Additionally, a semantic description of sensors and actuators is required, which can be checked with requirement descriptions in the action recipes.

Recently, a robot capability language incorporating sensor and actuator semantics (called SRDL) has been proposed by [15]. The OWL classes it uses for designating

sensors and actuators are extensions of the KnowRob ontology, and are thus easily integrable into the RoboEarth system. We chose the successor of the language, called SRDL2, for integration into RoboEarth [24].

## 5.2 Selecting Action Recipes

To decide whether a given robot is capable of executing a recipe, it is necessary to annotate said recipe with capability requirements. Thanks to formulating recipes in terms of classes in an OWL ontology, there are multiple ways to do that. For instance, in the KnowRob ontology there is an OWL class called *Reaching*, whose parent is *Voluntary-BodyMovement* and which has the children (among others) *ReachForObjectPose* and *HandoverObject*. Both children inherit the dependency on a *ReachingAbility* from their parent class, specified as OWL restriction *hasCapabilityDependency some Reaching-Capability* for the class *Reaching*.

With this adaption, for every recipe containing a sub-action that is a descendant of *Reaching* (for example *HandoverObject*), we can infer by OWL reasoning that a robot needs to provide the *ReachingCapability* capability to execute the recipe.

## 6 CRAM Plan Language

The CRAM planning language (CPL) [4] is used as the base language for executing RoboEarth. CPL extends on concepts from McDermott's "Reactive Plan Language" [16]. It provides amongst other things several control structures for executing and synchronizing tasks in parallel and failure handling. As it is based on Common Lisp, the expressiveness of a proven, Turing-complete programming language is available for implementing plans. Together with ROSLisp, the Lisp client library for the Robot Operating System, a whole range of tested robotics implementations are readily available for use in an execution plan. An exemplary plan describing the task of fetching a bottle to a person in a hospital bed is shown in Figure 4.

Objects, locations and actions are described by *designators*. Designators are basically symbolic descriptions that are resolved at the latest possible instant during task execution. For example, *robot-pose-handover1* in Figure 4 is a location designator that gets resolved to an actual robot base position such that the robot can reach to *hand-pose-handover1*, while the latter designator gets resolved to a position above the bed.

Actions to be executed in CPL are specified by describing corresponding goals. These goals in turn can be specified by a set of other goals or by calling a process module. Process modules implement hardware-specific behavior, for example for navigating to a given pose or for controlling the manipulators. They can thus be seen as a classic hardware abstraction layer, abstracting manipulators as well as sensors.

### 6.1 Designator Resolution

There are three different designator types in the current implementation:

1. Object designators describe objects that the robot has to interact with. In the example plan in Figure 4 both the bed and the bottle are designators of this kind.



```
(def-top-level-plan serve-a-drink ()
  (with-designators
    ((bed1 (object '((name bed1)
                    (type bed_piece-of-furniture))))
     (location-bed1 (location
                    '((of ,bed1))))
     (hand-pose-handover1 (location
                          '((on ,location-bed1))))
     (bottle1 (object '((name bottle1)
                       (type drinking-bottle))))
     (robot-pose-handover1 (location
                           '((to reach)
                             (loc ,hand-pose-handover1))))
    (achieve '(object-in-hand ,bottle1 :right))
    (achieve '(loc Robot ,robot-pose-handover1))
    (achieve '(object-handed-over ,bottle1
                                   ,hand-pose-handover1))))
```

**Fig. 4.** CRAM example plan for serving a drink to a patient

These kinds of designators are resolved to objects defined in the semantic map of the environment by evaluating the given description.

The bed object in the given example is resolved by the object type; i.e. all objects of the type *bed\_piece-of-furniture* found in the semantic map of the environment are candidates for resolution. It is worthwhile to mention that specializations of object types are also considered: E.g. assume that the above plan is to be run in a hospital room with a semantic map containing an object described as *hospital\_bed\_piece-of-furniture*. Presuming this object type is defined as a subclass of *bed\_piece-of-furniture*, the bed object designator gets still bound to the hospital bed object.

2. Location designators can be bound to objects to reference the pose of objects, as is the case with in the example in Figure 4. They can also describe poses relative to other locations and thus provide a basic form of geometric reasoning, as is the case with *hand-pose-handover1*.
3. Action designators provide symbolic annotations to a movement or perception action. They get resolved by the platform-dependent process modules responsible for the corresponding manipulator. In order to keep action designator descriptions consistent across different hardware platforms, we plan to extend the RoboEarth ontology as a basis for action designators.

## 6.2 Translating Action Recipes into CRAM Plans

The language for formulating action recipes described in [24] is designed to be a language for specifying and sharing hardware-independent concepts and enabling automated reasoning on these concepts, i.e. it is a language to describe the semantics of a task. Therefore, before execution it is necessary to translate these descriptions into a plan description language, containing calls to manipulator commands, sensor interpretation commands, etc. Older prototypical implementations of an execution engine for the RoboEarth project are translating recipes into a finite state machine<sup>1</sup>.

<sup>1</sup> A video of a public demonstration where one implementation of this kind was used is available at <http://youtu.be/r5ZRxb0pSQ>

As the sub-tasks in an action recipe are partially ordered, an action recipe can be represented as a directed, acyclic graph. To create a plan from this graph, our implementation first applies a topological sorting (using an implementation of the algorithm proposed by [14]). Our implementation thus currently does not account for parallel execution of sub-tasks.

Having found a valid topological ordering, we then proceed to generate a list of designators from the task descriptions in the recipe. The creation of designators for objects and locations in the recipe is straight forward: Object designators are created from the object type and name as specified in the recipe; e.g. the object *Bed1* from Figure 3 is translated to the designator *bed1 (object ((name bed1) (type bed\_piece-of-furniture)))*. Similarly, location descriptions like *handPose-handover1* are translated to location designators as *hand-pose-handover1 (location ((above-of, location-bed1)))*

For translating tasks to goals, we have defined a set of mappings. E.g. the *MoveBase-ToHandoverPose* from Figure 2 is mapped to the goal (*loc Robot pose*), whereas *pose* is the location designator for the target pose. In the future, we plan to integrate these mappings into the action recipe language, to increase the applicability of the approach.

## 7 Experiments

The work described in this paper was used for the demonstrator for the third RoboEarth workshop at TU Munich in February 2012 on a PR2 and the TUE's AMIGO robot platform [9]. Both robots had the same task similar to the examples given above, i.e. to fetch a bottle from inside a cabinet to a hospital bed, but were situated in different rooms in different locations. Two recipes were used, one describing the task of grasping an object and one describing fetching an object to a hospital bed whereas the former was used by the latter. There were several resources that had to be downloaded prior to task execution; i.e. maps describing the environments and object detection models. Additionally, the robots were downloading an articulation model [20] to open the cabinet door or creating one, if none was available.

Both robots were able to successfully retrieve the required resources from the database, thus synthesize the plan and complete the task. We noticed that due to the static nature of the OWL to CRAM translation, we had to take care that the process modules for both platforms are equivalent in their effects. A video of the process can be found at <http://youtu.be/o3oi7PZkFII>

## 8 Conclusions and Future Work

In this work, a plan-based execution engine for the RoboEarth-Project has been presented. The primary focus of this work was the tight integration of the concept in the whole RoboEarth framework. The key aspect of RoboEarth is the use of action recipes, which act as symbolic, high-level task descriptions and which are described in a description language in a platform independent way to allow a platform independent sharing and reasoning.

In the current implementation, the plans describing how to achieve sub-goals (e.g. (*achieve (object-in-hand ...)*) in Figure 4) for picking up objects) are implemented on

the robot itself. A more generic approach would be to have these sub-plans generated automatically from action recipes available in the database, as we do it with the top level plan (described in section 6.2).

Another aspect that we will investigate further is how to move the knowledge processing onto the database servers. This would enable robots with less computing power to retrieve and execute action recipes, if a reliable Internet connection is given.

**Acknowledgments.** The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement number 248942 RoboEarth.

## References

1. Alami, R., Fleury, S., Herrb, M., Ingrand, F., Robert, F.: Multi-robot cooperation in the martha project. *IEEE Robotics & Automation Magazine* 5(1), 36–47 (1998)
2. Beetz, M., Arbuckle, T., Belker, T., Cremers, A.B., Schulz, D., Bennewitz, M., Burgard, W., Hahnel, D., Fox, D., Grosskreutz, H.: Integrated, plan-based control of autonomous robot in human environments. *IEEE Intelligent Systems* 16(5), 56–65 (2001)
3. Beetz, M., Klank, U., Kresse, I., Maldonado, A., Mösenlechner, L., Pangercic, D., Rühr, T., Tenorth, M.: Robotic roommates making pancakes. In: 2011 11th IEEE-RAS International Conference on Humanoid Robots (Humanoids), pp. 529–536 (2011) doi:10.1109/Humanoids.2011.6100855
4. Beetz, M., Mösenlechner, L., Tenorth, M.: CRAM - A Cognitive Robot Abstract Machine for everyday manipulation in human environments. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1012–1017 (2010)
5. Boutilier, C., Brafman, R.: Partial-order planning with concurrent interacting actions. Arxiv preprint arXiv:1106.0249 (2011)
6. Brumitt, B., Stentz, A.: Grammps: A generalized mission planner for multiple mobile robots in unstructured environments. In: Proceedings of 1998 IEEE International Conference on Robotics and Automation, vol. 2, pp. 1564–1571. IEEE (1998)
7. Clement, B., Durfee, E.: Top-down search for coordinating the hierarchical plans of multiple agents. In: Proceedings of the Third Annual Conference on Autonomous Agents, pp. 252–259. ACM (1999)
8. Collins, J., Tsvetovat, M., Mobasher, B., Gini, M.: Magnet: A multi-agent contracting system for plan execution. In: Proc. of SIGMAN, pp. 63–68 (1998)
9. Control Systems Technology Group, Eindhoven University of Technology. AMIGO specifications, <http://www.roboticopenplatform.org/wiki/AMIGO> (accessed December 19, 2011)
10. Firby, R.: An investigation into reactive planning in complex domains. In: Proceedings of the Sixth National Conference on Artificial Intelligence, vol. 1, pp. 202–206 (1987)
11. Garage, W.: Xml robot description format (urdf), <http://www.ros.org/wiki/urdf/XML> (accessed December 1, 2011)
12. Gat, E.: Esl: A language for supporting robust plan execution in embedded autonomous agents. In: Proceedings of IEEE Aerospace Conference, vol. 1, pp. 319–324. IEEE (1997)
13. Horridge, M., Patel-Schneider, P.: Manchester syntax for owl 1.1. In: OWL: Experiences and Directions, OWLED (2008)
14. Kahn, A.B.: Topological sorting of large networks. *Commun. ACM* 5, 558–562 (1962)

15. Kunze, L., Roehm, T., Beetz, M.: Towards semantic robot description languages. In: IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China (2011)
16. McDermott, D.: A reactive plan language. Tech. rep., Citeseer (1991)
17. Murthy, A.C., Douglas, C., Cutting, D., Das, D., Borthakur, D., Collins, E., Soztutar, E., Kuang, H., Homan, J., Konar, M., Daley, N., O'Malley, O., Hunt, P., Angadi, R., Agarwal, S., Shvachko, K., Stack, M., Sze, T.W.N., Lipcon, T., White, T., Shao, Z.: Apache Hadoop, a framework for reliable, scalable and distributed computing, <http://hadoop.apache.org>
18. Muscettola, N., Dorais, G., Fry, C., Levinson, R., Plaunt, C.: Idea: Planning at the core of autonomous reactive agents. In: Proceedings of the Workshops at the AIPS-2002 Conference, Toulouse, France (2002)
19. Quigley, M., Gerkey, B., Conley, K., Faust, J., Foote, T., Leibs, J., Berger, E., Wheeler, R., Ng, A.: Ros: an open-source robot operating system. In: ICRA Workshop on Open Source Software (2009)
20. Rühr, T., Sturm, J., Pangercic, D., Cremers, D., Beetz, M.: A generalized framework for opening doors and drawers in kitchen environments. In: IEEE International Conference on Robotics and Automation (ICRA), St. Paul, MN, USA (to appear, 2012)
21. Schieble, B., Häussermann, K., Zweigle, O.: Deliverable D6.1: Complete specification of the RoboEarth platform. Tech. rep. (December 1, 2010), <http://www.robearth.org/wp-content/uploads/2011/03/D61.pdf>
22. Simmons, R., Goodwin, R., Haigh, K.Z., Koenig, S., O'Sullivan, J.: A layered architecture for office delivery robots. In: Proceedings of the First International Conference on Autonomous Agents, pp. 245–252. ACM (1997)
23. Tenorth, M., Beetz, M.: KnowRob - knowledge processing for autonomous personal robots. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2009, pp. 4261–4266. IEEE (2009)
24. Tenorth, M., Perzylo, A., Lafrenz, R., Beetz, M.: The RoboEarth language: Representing and Exchanging Knowledge about Actions, Objects, and Environments. In: IEEE International Conference on Robotics and Automation (ICRA), Saint Paul, USA (accepted for publication, 2012)
25. W3C OWL Working Group, OWL 2 Web OntologyLanguage Document Overview. W3C recommendation, W3C (2009) (accessed December 1, 2011)
26. Waibel, M., Beetz, M., Civera, J., D'Andrea, R., Elfring, J., Galvez-Lopez, D., Haussermann, K., Janssen, R., Montiel, J., Perzylo, A., et al.: RoboEarth. IEEE Robotics & Automation Magazine 18(2), 69–82 (2011)
27. Williams, B., Ingham, M., Chung, S., Elliott, P.: Model-based programming of intelligent embedded systems and robotic space explorers. Proceedings of the IEEE 91(1), 212–237 (2003)
28. Zweigle, O., van de Molengraft, R., D'Andrea, R., Häussermann, K.: RoboEarth: connecting robots worldwide. In: Proceedings of the International Conference on Interaction Sciences: Information Technology, Culture and Human, pp. 184–191. ACM (2009)

# Exchanging Action-Related Information among Autonomous Robots

Moritz Tenorth and Michael Beetz

Intelligent Autonomous Systems Group  
Department of Informatics, Technische Universität München  
{tenorth,beetz}@cs.tum.edu

**Abstract.** In this paper, we describe representations and inference techniques that are used in the RoboEarth system for the web-based exchange of information between robots. We present novel representations for environment maps that combine expressive semantic environment models with techniques for selecting suitable maps from the web-based RoboEarth knowledge base. We further propose techniques for improving class-level object models with additional information as needed for distributed learning of object properties. In an integrated experiment, we show that the system enables robots to perform mobile manipulation tasks including the retrieval of suitable environment maps and the estimation and exchange of object property information.

## 1 Introduction

The ROBOEARTH system [1] is designed as a web community by robots for robots: Similar to community-driven web pages for humans, such as Wikipedia, which enable humans to exchange knowledge among each other, ROBOEARTH allows knowledge exchange between robots. When one robot has learned how to perform a task or has created an object model, it can upload this information to ROBOEARTH. Other robots can later download it and do not have to learn everything on their own again.

Since the platform is intended to serve for exchanging various kinds of information among heterogeneous robots with different capabilities, the development of appropriate techniques for finding relevant information becomes a challenging problem. When humans search for information on the Web, they can easily distinguish relevant from irrelevant information (e.g. search results caused by ambiguous meanings of the search terms) and filter out information they cannot make use of (e.g. text written in a language they do not understand). Realizing these capabilities on a robot requires powerful knowledge representation and reasoning techniques.

A central part of the ROBOEARTH project is thus the development of an expressive, formal representation language to encode the exchanged information. It helps the robot answer questions such as: Which descriptions exist for the task to be performed? Which of them can be used given the robot's hardware and software configuration? Are all required capabilities available? Which object models

are needed to perform the task? Are there maps that describe the environment the robot is to operate in, which kinds of information do they provide?

Autonomously taking these decisions requires access to meta-information that describes which kind of knowledge is encoded in which format, including for instance which coordinate systems are used for spatial information or which capabilities are needed to perform a task. This requires more comprehensive representations than commonly used for describing actions, objects, or environment structures in robotics. Related approaches usually focus on the representation of the information content, e.g. the spatial information in a map, but do not describe meta-information, e.g. *which* environment is actually described in that map. Hierarchical Task Networks (HTN [2]) and related plan languages are similar to the action representation used in ROBOEARTH but focus on the description of the task itself, i.e. its sub-actions, goals, and ordering constraints. XABSL [3], mainly used in the RoboCup soccer context, describes actions in terms of hierarchical finite state machines. AutomationML [4] is a standard for describing task information and spatial configurations, mainly used in industrial applications. The FIPA [5] standard primarily deals with the definition of communication standards for software agents. Object description formats like the proprietary DXF [6] or the open Collada [7] standard describe objects using meshes and textures, but without further specifying semantic properties.

In [8], we gave a general overview of the representation language used in ROBOEARTH. In this paper, we will first describe the work-flow for downloading information from ROBOEARTH to show how the reasoning processes are integrated into the overall system. Then, we will focus on two specific reasoning problems, namely the representation and discovery of map information in ROBOEARTH, and the retrieval, improvement and sharing of object models.

The main contributions of this paper are the following: We present novel representations for environment maps that combine the expressive semantic environment models introduced in [9] with explicit descriptions of the environment that is described in the map, and with techniques for selecting suitable maps from a web-based database. We further present novel techniques for improving class-level object models with additional information as needed for distributed learning of object properties.

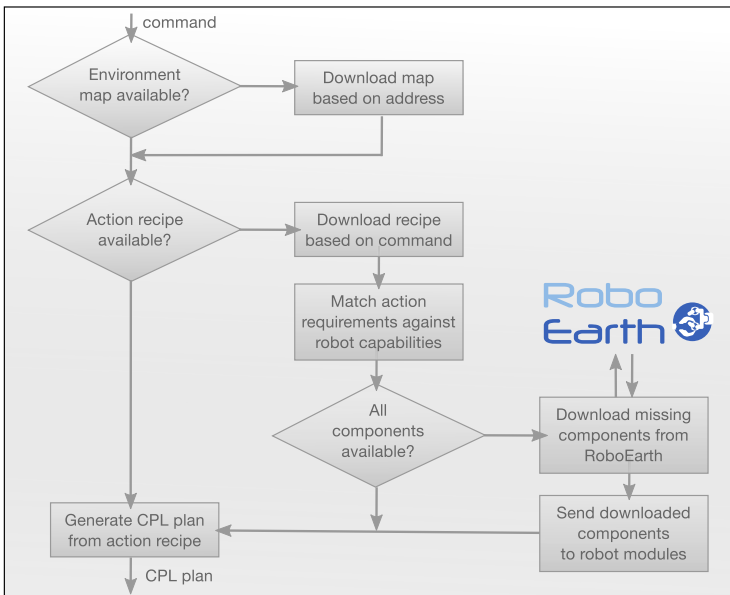
## 2 Retrieving Information Needed for a Task

A common scenario in ROBOEARTH is that a robot is given a task that it does not have a plan for. This means the robot needs to obtain an action plan and identify and retrieve missing components such that it, in the end, has all information it needs to successfully accomplish the task.

We assume that the robot has initial knowledge about the environment it is operating in, for example the address of the room (see Section 3.3), and a semantic model of itself, including its hardware and software components and abstract

capabilities. This self-model needs to be created only once for each kind of robot and is described using the Semantic Robot Description Language (SRDL [10]). When the robot acquires additional capabilities, e.g. by downloading an object model from ROBOEARTH, it adds them to its self-model so that they are available for future queries.

Figure 1 outlines the main inference steps involved in the retrieval of information. These inferences are performed before the robot starts to execute the task and ensure the availability of all components that are required for this task. First, the robot checks whether it has a map of the environment at hand and, if not, downloads one from ROBOEARTH to obtain information about free space and obstacles as well as about objects in the environment. The next step is to search for task descriptions, called “action recipes”, that describe on the one hand which actions need to be performed, and, on the other hand, list dependencies the task has in terms of components and capabilities. The robot matches this specification against its self-model and iterates over all missing components to check whether they can be obtained from ROBOEARTH. If all missing components can be provided, the system converts the action recipe into an executable action plan in the CRAM plan language (CPL, [11]) and sends it to the CRAM executive.

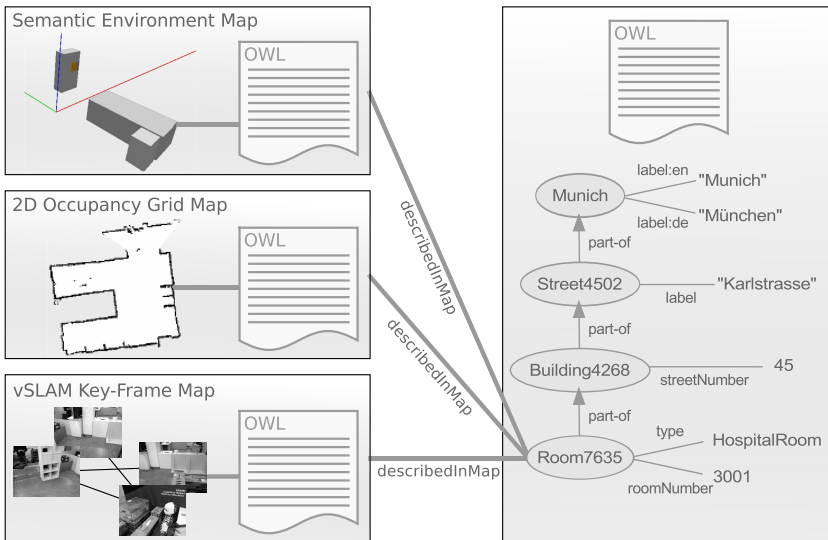


**Fig. 1.** Flowchart describing the process for downloading information from ROBOEARTH. Based on the command to perform a task, the robot determines which information it needs and downloads the required descriptions from the ROBOEARTH knowledge base

The representation language [8] represents knowledge using the Web Ontology Language (OWL) as an extension of the KNOWROB knowledge base [12]. The ontologies and software modules are available online as open source software.<sup>1</sup>

### 3 Representation of Environment Information

Figure 2 visualizes different kinds of environment maps that can be described and exchanged via ROBOEARTH. Their descriptions consist of two parts: the representation of the map's content, described in Section 3.1, and of the environment that is described in this map, which will be introduced in Section 3.3.



**Fig. 2.** Representations of different kinds of environment maps. Each map consists of an OWL description of its type and properties and optionally a binary file. In the case of semantic maps, the OWL description also describes the objects that are part of the map. The map is linked to a description of the environment it describes via the *describedInMap* relation.

#### 3.1 Types of Environment Maps

For all tasks that include navigation or interaction with objects, a robot needs an environment model to plan its actions. Depending on the action to be performed, different kinds of maps are needed: Occupancy grid maps discriminate between free space and obstacles for localization, navigation and obstacle avoidance. Semantic environment maps contain localized object instances and can be used for grounding abstract object descriptions contained in an action recipe.

<sup>1</sup> Online: <http://ros.org/wiki/knowrob>, [http://ros.org/wiki/re\\_ontology](http://ros.org/wiki/re_ontology)



Maps are described by an OWL specification of its type and properties that can optionally be linked to binary files. After download, the OWL description is parsed, which creates an instance of the respective type of map in the knowledge base such that the robot 'knows' that the map is available. Depending on the type of the map, the system can decide how to proceed after download: For example, a serialized occupancy grid is sent to the localization and navigation routines.

### 3.2 Spatio-temporal Object Pose Representation

Semantic environment maps consist of a set of typed, localized object instances. Based on the type, one can retrieve semantic information about the objects that is described at the class level, for example descriptions of physical parts, articulation properties, or CAD models to be used for recognition, spatial reasoning and visualization. Section 4 describes in detail how these properties are described for object classes.

To account for the dynamics in the environment and in order to be able to describe changes in the world, introduced by the robot itself or external effects, the representation of object instances can store and reason about changes in object poses. Multiple detections at different poses can be attached to an object instance; each of them is annotated with a time point and the perception method that has been used. This spatio-temporal object representation can be generated automatically from perceptual information and provides a flexible way of integrating novel information about objects into the knowledge base. It is described in more detail in 8.

### 3.3 Discovery of Suitable Maps for the Robot's Environment

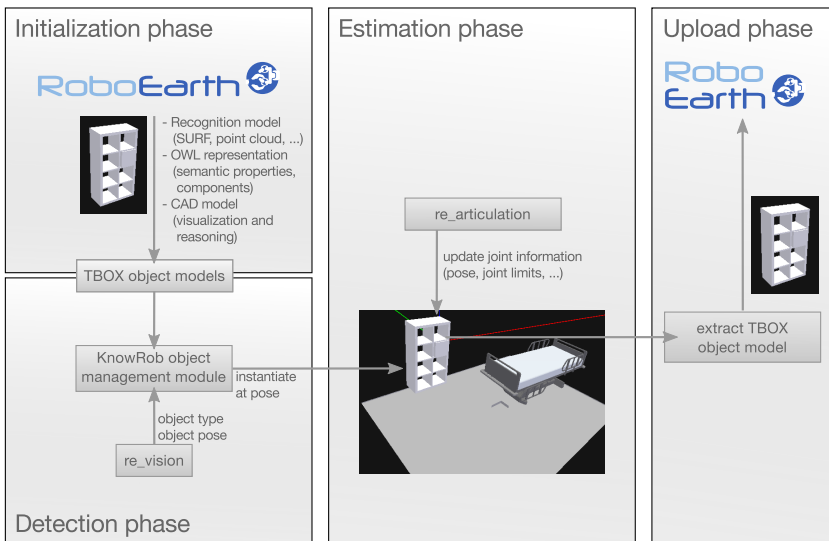
In order to exchange maps with other agents, a robot needs to be able to describe which environment the maps belong to. This information is very important in the ROBOEARTH setup as it allows to specify which maps are desired when sending a request to the knowledge base. Since environments can be described in many different ways, the representation should support a spatial hierarchy (city – street – building), a semantic hierarchy (room – kitchen), and different labels for the same room.

We have developed a flexible approach that is similar to the human way of describing an address: Maps are annotated with data like the city, street, building, floor, room number or room type they describe, as shown in the right part of Figure 2. These components are linked by a transitive part-of relation. This allows to query for combinations of these levels, e.g. to search for all maps of a kitchen in Karlstrasse in Munich, or for all rooms on the third floor of Karlstrasse 45. It further allows to combine labels, such as room numbers, with types of rooms (private homes usually do not have room numbers) and to attach multiple labels to the same physical entity (Karlsplatz and Stachus are two names for the same square in Munich).

The representation can be combined with other techniques for searching for environment maps: Searching by GPS coordinate would be useful for field robotics, while there is normally no GPS signal available indoors. Alternatively, one can also assume that the robot is continuously localized (quantitatively or qualitatively) in a hierarchy of metric and topological maps, and that it can detect when it runs out of one map. In this case, it can search for an extension map describing the new area. This method, however, requires modification of the lower-level map management infrastructure.

## 4 Retrieving and Improving Object Models

While the information a robot perceives is at first environment-specific, there are often generic pieces of information that can be extracted from it. For example, if the robot estimates the articulation properties of an object [13], this information is a priori specific to the object instance the robot was interacting with. If the object, however, is an instance of a common type, it may make sense to extract the generic information and share it with other robots. For this reason, we have developed methods to generate class-level object models, taking a specific object instance as an example.



**Fig. 3.** Schematic overview of the object model up- and download. Robots can retrieve an object model from ROBOEARTH, instantiate the model if the object has been perceived in the environment, estimate additional properties, and upload a model generated from this object instance to share the new information with other robots.

Figure 3 shows the life cycle of an object model: During the initialization phase, shown in Figure 1, the model is downloaded from ROBOEARTH. If the robot detects the object in the environment, the abstract class-level model is instantiated at the respective position, recursively translating all object-intrinsic coordinates into global map coordinates. During interaction, the robot estimates object properties like the positions and types of joints and adds them to the object instance. After finishing the task, it checks which pieces of information have been updated and for which sharing makes sense.

For these objects, it extracts a class-level (TBOX) model and uploads it to the ROBOEARTH knowledge base. The ROBOEARTH language supports the explicit specification of coordinate frames (e.g. map-global, relative to other objects or other components of the same objects, or qualified using frames in the ROS tf system<sup>2</sup>), and provides methods for the conversion between coordinates in different frames. By recursively converting the coordinates and translating instance-level descriptions into class restrictions, the system can create an object model that includes the new information.

## 5 Experiments

In a recent experiment, we have investigated how these methods can enable robots to perform a drink serving task in a previously unknown environment. The experiment has been done in two locations with two different robot platforms, a PR2 and the Amigo robot. Both robots were only equipped with the command and the address of the environment they were operating in and successfully downloaded all information required for the task. Compared to prior experiments<sup>3</sup>, the robots thus had to operate with less initial information and perform more inference to find, select and apply information from the ROBOEARTH knowledge base.

The upper part of Figure 4 shows the environment maps that have been downloaded from ROBOEARTH. ROBOEARTH offers a SerQL<sup>4</sup> query interface<sup>3</sup>. The following query has been used to download the map information:

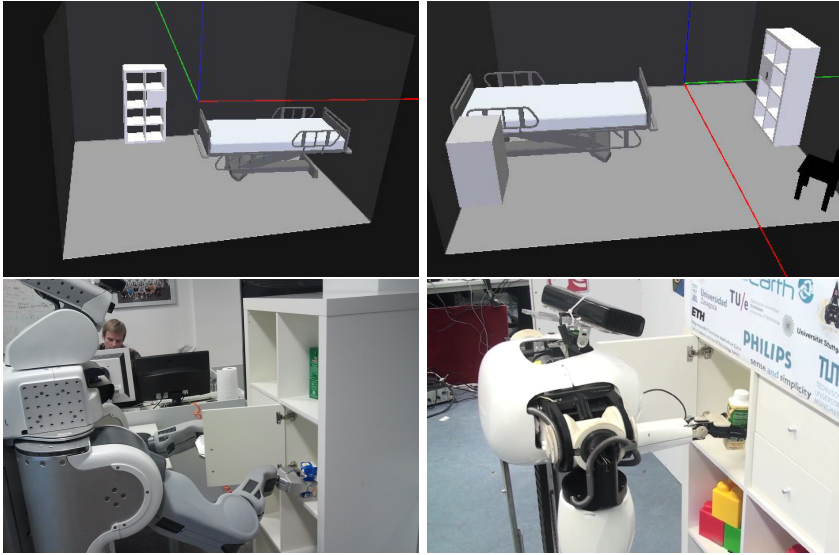
```
select source from context source {R}
  kr:describedInMap {S} ;
  kr:roomNumber {N}
  where N like "3001"
  using namespace
  kr=<http://ias.cs.tum.edu/kb/knowrob.owl#>;
```

Based on this map, the robots (Figure 4 bottom) could navigate to the appropriate positions and locate the objects required for the task. The action recipe to be used was selected using the following query:

```
select source from context source {A}
  rdfs:label {"serve_a_drink"^^xsd:string}
  using namespace
  rdfs=<http://www.w3.org/2000/01/rdf-schema#>
```

<sup>2</sup> <http://ros.org/wiki/tf>

<sup>3</sup> <http://api.roboearth.org>



**Fig. 4.** Top: Semantic environment maps of the two hospital rooms, downloaded from ROBOEARTH based on the address and room number. Bottom: PR2 and Amigo robots opening the cabinet and picking up the drink to be served.

The action recipe was then matched against the robots' capabilities with the result that all required capabilities were available, but some recognition models for some of the objects mentioned in the task were missing (namely the bottle and the bed), which have been downloaded, including the CAD models shown in Figure 4. The following CPL plan was then generated from the action recipe:

```
(def-top-level-plan serve-a-drink ()
  (with-designators (
    (bottle1 (object '((name bottle1)
                      (type drinking-bottle)))
    (bed1 (object '((name bed1)
                   (type bed-piece-of-furniture))))
    (hand-pose-handover1 (location '((on ,bed1)))
    (robot-pose-handover1 (location '((to reach)
                                     (side :right)
                                     (loc ,hand-pose-handover1)))
    (arms-at101 (action '((type trajectory)
                        (pose ,hand-pose-handover1)
                        (side :right))))
    (unhand-action102 (action '((type open-gripper)
                               (side :right))))
  )

  (achieve '(object-in-hand ,bottle1 :right))
  (at-location (robot-pose-handover1))
  (achieve '(arms-at ,arms-at101))
  (achieve '(arms-at ,unhand-action102))))
```

This experiment shows that the representation and reasoning mechanisms in ROBOEARTH can be used to represent and select information about human

environments and mobile manipulation tasks. They have successfully been used by two heterogeneous robots in different environments who both performed the same mobile manipulation task.

## 6 Discussion and Conclusions

In this paper, we presented novel methods for exchanging information between robots with a focus on two applications: the exchange of environment maps and of object models. Regarding maps, we discussed the representation of the different kinds of maps and the relation between the maps themselves and the environments they describe. These techniques allow easy discovery of information in the database using only information that can realistically be assumed to be available on the robot. Object models are described by a combination of a semantic description in OWL with (binary) recognition models and support the representation of a component hierarchy as well as articulation properties. We showed that object class descriptions can be downloaded and be instantiated once the corresponding object has been detected in the environment. Based on these instances, the robot can estimate additional information, attach this information to the object, and share the newly acquired information with other robots. The presented methods significantly raise the level of semantics and are a big step towards robots that autonomously exchange information. Challenges to be addressed in the future include the question of which pieces of information are actually worth being exchanged.

**Acknowledgments.** This work is supported in part by the EU FP7 Project *RoboEarth* (grant number 248942).

## References

1. Waibel, M., Beetz, M., D'Andrea, R., Janssen, R., Tenorth, M., Civera, J., Elfiring, J., Gálvez-López, D., Häussermann, K., Montiel, J., Perzylo, A., Schießle, B., Zweigle, O., van de Molengraft, R.: *RoboEarth - A World Wide Web for Robots*. *Robotics & Automation Magazine* 18(2), 69–82 (2011)
2. Erol, K., Hendler, J., Nau, D.: *Htn planning: Complexity and expressivity*. In: *Proceedings of the National Conference on Artificial Intelligence*, pp. 1123–1123. John Wiley & Sons LTD. (1994)
3. Loetzsch, M., Risler, M., Jüngel, M.: *XABSL-a pragmatic approach to behavior engineering*. In: *Proceedings of IEEE/RSJ International Conference of Intelligent Robots and Systems (IROS)*, Citeseer, pp. 5124–5129 (2006)
4. Drath, R., Luder, A., Peschke, J., Hundt, L.: *AutomationML-the glue for seamless automation engineering*. In: *IEEE International Conference on Emerging Technologies and Factory Automation, ETFA 2008*, pp. 616–623. IEEE (2008)
5. O'Brien, P., Nicol, R.: *FIPA – towards a standard for software agents*. *BT Technology Journal* 16(3), 51–59 (1998)
6. Rudolph, D., Stürznicke, T., Weissenberger, L.: *Der DXF-Standard*, Rossipaul (1993)

7. Arnaud, R., Barnes, M.: COLLADA: sailing the gulf of 3D digital content creation. AK Peters, Ltd. (2006)
8. Tenorth, M., Perzylo, A.C., Lafrenz, R., Beetz, M.: The roboearth language: Representing and exchanging knowledge about actions, objects, and environments. In: IEEE International Conference on Robotics and Automation (ICRA), St. Paul, MN, USA, May 14–18 (accepted for publication, 2012)
9. Tenorth, M., Kunze, L., Jain, D., Beetz, M.: KNOWROB-MAP – Knowledge-Linked Semantic Object Maps. In: 10th IEEE-RAS International Conference on Humanoid Robots, Nashville, TN, USA, December 6-8, pp. 430–435 (2010)
10. Kunze, L., Roehm, T., Beetz, M.: Towards semantic robot description languages. In: IEEE International Conference on Robotics and Automation (ICRA), Shanghai, China, May 9–13 (2011)
11. Beetz, M., Mösenlechner, L., Tenorth, M.: CRAM – A Cognitive Robot Abstract Machine for Everyday Manipulation in Human Environments. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan, October 18-22 (2010)
12. Tenorth, M., Beetz, M.: KnowRob – Knowledge Processing for Autonomous Personal Robots. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 4261–4266 (2009)
13. Sturm, J., Stachniss, C., Burgard, W.: Learning kinematic models for articulated objects. *Journal on Artificial Intelligence Research, JAIR* (2011)
14. Broekstra, J., Kampman, A.: Serql: A second generation rdf query language. In: Proc. SWAD-Europe Workshop on Semantic Web Storage and Retrieval, pp. 13–14 (2003)

# New Motor Primitives through Graph Search, Interpolation and Generalization

Miha Deniša and Aleš Ude

Jožef Stefan Institute, Jamova 39, Ljubljana, Slovenia  
{miha.denisa,ales.ude}@ijs.si

**Abstract.** This paper presents our proposed approach for discovering new motor primitives in a database of demonstrated example movements. Example trajectory data is usually obtained from human demonstration or by kinesthetic guiding. It is then organized and clustered into a binary tree with transition graphs at every level. Each level presents a different granularity of example data. We show that new movements (or their parts) can be discovered through graph search, by exploiting the interdependencies between the movements encoded by the graph. By combining results with optimized interpolation new series of primitives can be found. A complete representation of new, not directly demonstrated, movements can be constructed by using new series of movements with statistical generalization techniques.

## 1 Introduction

Programming by demonstration, also called imitation learning, is a widely used approach for the acquisition of new behaviours [13]. Data suitable for learning by imitation is usually acquired from human demonstrations using optical marker systems or standard computer vision techniques, or by kinesthetic guiding. Due to the variability of problems that an autonomous robot might encounter in natural environments, it is not feasible to obtain all necessary movements from the demonstrated behaviors. To reduce the burden of teaching, a database of example motions must allow for new, not directly demonstrated motions to be discovered and new behaviors to be generated. In this paper we propose an approach that uses the concept of motion graphs and statistical generalization to create new movement primitives.

To organize the available example movements in a graph-like structure, we follow (with modifications) the approach of Yamane et al. [17]. While he proposed to cluster the state vectors from the example trajectories using principal component analysis (PCA) and minimum-error thresholding technique [5], we cluster our data with a k-mean algorithm [10]. Note that unlike Yamane et al., whose interest was primarily in full-body movements, we focus on arm trajectories and manipulation tasks, which often require fine precision. The results of clustering are used to construct a binary tree, which represents the captured data at different granularities. Every level of the binary tree is associated with a transition graph that describes transitions between the nodes at that level. The

nodes contain state vectors from all example trajectories. In contrast to Yamane et al. we do not assume connections between desired start and end nodes at the desired level. We employ a graph search algorithm and optimized interpolation to find best suited paths from desired start and end state vectors at the desired level of granularity. Statistical methods [16] are then utilized to generalize the newly discovered trajectories and to create a complete representation for a newly discovered motor primitive.

The rest of the paper is organized as follows. In Section [1.1] we describe previous research related to this paper. Section [2] presents the process of constructing the database. The following section deals with the discovery and generalization of motor primitives using the transition graphs and optimized interpolation. Experimental evaluation of our approach is presented in Section [4], while the conclusion and summary are given in Section [5]. At the end a brief review of dynamic movement primitive representation is given in the appendix.

## 1.1 Related Work

A lot of research on imitation is focused on learning from a single demonstration, like for example in the case of dynamic movement primitives [4], or learning from multiple demonstrations, where each demonstration is a variant of the same type of movement, like in [2] when learning table tennis strokes. Other research includes segmentation and sequencing of motor primitives [7], storing movement primitives as basis functions [9] and the generation of mixtures of motor primitives [11]. However, compared to the work in computer graphics community, who has always assumed that a large database of diverse motion capture data is available for the generation of computer animations, the number of example movements considered in imitation research has usually been more limited and the example trajectories has been less diverse. The work of Kulić et al. [7,8] is a notable exception. On the other hand, the computer graphics community have shown that by exploiting the structured nature of motion capture data in large databases, which is made evident in motion graphs, smooth transitions between interconnected body movements can be found [13]. We therefore base the discovery of new movement primitives on a graph search process, which enables us to find subtler transitions between different example movements in the database.

Motion graphs have been proposed and investigated by the computer graphics community to encapsulate connections in the available motion capture data [6]. Kovar et al. [6] applied them to generate different styles of locomotion along arbitrary paths. A graph search algorithm was employed to find nodes that represent possible transitions between parts of the captured movements. Safonova and Hodgins [14] combined motion graphs and interpolation techniques to increase the number of paths through the graph. While the motion graph literature is vast, our work is most related to the approach of Yamane et al. [17], who used a binary tree to organize the data and the resulting transition graphs to generate human body locomotion on a desired path. A binary tree structure was also used by Sidenbladh et al. [15] for an efficient sampling of human poses.



## 2 Building the Database

In our approach, a database of example trajectories is organized using an approach similar to the one proposed by Yamane et al. [17]. It will be pointed out where our approach differs. In the developed system, example trajectories, which need to be organized into a graph-like data structure, can be given either in task or in joint space. For the purpose of building the database, we concatenate the acquired trajectories in a sample motion matrix:

$$\mathbf{X} = [\mathbf{x}_1 \ \mathbf{x}_2 \ \cdots \ \mathbf{x}_{N_f}], \quad (1)$$

where  $\mathbf{x}_i$  denotes the state vectors sampled at a given discrete time interval and  $N_f$  is the total number of all postures belonging to the example trajectories incorporated into the database. State vectors for the end-effector trajectories specified in Cartesian space are given as

$$\mathbf{x}_i = [p_{xi} \ \dot{p}_{xi} \ p_{yi} \ \dot{p}_{yi} \ p_{zi} \ \dot{p}_{zi}]^T, \quad (2)$$

where  $p_{ji}$  and  $\dot{p}_{ji}$ ,  $j = x, y, z$ , denote the position and velocity at time  $t_i$ . If the trajectories are given in joint space, then state vectors are defined by

$$\mathbf{x}_i = [q_{1i} \ \dot{q}_{1i} \ q_{2i} \ \dot{q}_{2i} \ \cdots \ q_{di} \ \dot{q}_{di}]^T, \quad (3)$$

where  $j$ -th joint angle and its velocity at time  $t_i$  are denoted by  $q_{ji}$  and  $\dot{q}_{ji}$ , and  $d$  is the number of the robot degrees of freedom (DOF).

Once example trajectories are stored and arranged in a sample matrix, they can be utilized to build a binary tree. The complete sample data matrix  $\mathbf{X}$  represents the root node, which we split using the k-mean algorithm and thus gain its two child nodes. Those nodes are then split again to gain nodes at the next depth of our binary tree. Yamane et al. [17] based their criterion for when to stop splitting the tree nodes solely on the number of state vectors that remain in the two following child nodes. Instead, we use a criterion based on the variability of data contained in the node. We define the mean distance  $d_k$  of node  $k$  as

$$d_k = \frac{\sum_{i=1}^{n_k} d(\mathbf{x}_{ki}, \mathbf{c}_k)}{n_k}, \quad (4)$$

where  $n_k$  denotes the number of state vectors clustered at node  $k$ . For  $d(\mathbf{x}_{ki}, \mathbf{c}_k)$  euclidean distance between state vectors  $\mathbf{x}_k$  associated with this node and the nodes centroid  $\mathbf{c}_k$ , gained with k-mean algorithm, is used. If  $d_k$  is lower than a predefined threshold, then state vectors contained in the node are similar. By not splitting the nodes in this case, we avoid new nodes that would make the binary tree unnecessarily deep. On the other hand, we split the nodes even if they include a small number of state vectors if these state vectors are sufficiently diverse. In this way we gain on precision of our representation while preventing the binary tree and the resulting transition graphs from becoming unnecessarily large. With these criteria we cluster the data into nodes until we do not have any nodes left to split. We extend every branch to the last level by copying the

leaf nodes and so ensuring that all state vectors are represented at all levels of the binary tree.

For further processing it is not necessary to store all state vectors at each node of the binary tree. Storing the mean of all state vectors associated with the node is sufficient. Only if the node contains exactly one final configuration on a trajectory, we store this final configuration instead of the mean of all state vectors. In this way we ensure that movements generated by graph search end in the same end points as the original movements. At each depth of the binary tree, we build a transition graph that represents all possible transitions between the nodes at the current depth. The edge weights in the transition graph represent the probability of transition from one node to another. Transition probability from node  $k$  to node  $l$  is estimated by

$$p_{kl} = \frac{t_{kl}}{n_k} \quad (5)$$

where  $t_{k,l}$  denotes the number of transitions observed in all trajectories of the original data, i. e. the number of all state vectors clustered in node  $k$  that have a successor in node  $l$ . Given the binary tree, average state vectors (or end points) at nodes of the binary tree, and transition graphs for every depth, we can now search for new movements in the transition graphs.

### 3 Discovering New Movement Primitives

We start the process of discovering new discrete movement primitives, i. e. point-to-point movements such as reaching and grasping, by selecting the desired start and end points on two different trajectories. If the desired start and end point do not belong to one of the original example movements, we first establish the closest start and end node in the transition graph. At this point we also need to decide for a level in the binary tree, which determines the fidelity of reproduction compared to the original trajectories. We then try to find a path between these two nodes at the chosen level by using the A\* algorithm. If the desired start and end nodes belong to the same connected component in the transition graph at this layer, the shortest path is found. If they do not, we can use the binary tree like structure of our database to find a level at which the connection exists. However, this higher level does not have the desired granularity. To achieve the proper granularity we need to move down the layers to the desired depth (see Fig. II). We do that by finding the node which division caused the break in the connection. This is the node that has only one child accessible from the start node at the next layer. We then store both parts of the path, i. e. the path connecting start node to that child and the path connecting the second child to the desired end node. If another break occurs while we move down the layers, we repeat this process and store the associated path parts. Once we return to the desired level, we have all parts of the path we are searching for.

Once the shortest path or its parts have been found, we obtain a series of nodes, i. e. mean values of state vectors. New example trajectories (or its parts)

are defined by these state vectors, which need to be enhanced by the time evolution of the trajectory. For this purpose we define

$$\tau_{i,j} = \frac{\tau_i}{2} + \frac{\tau_j}{2}, \tag{6}$$

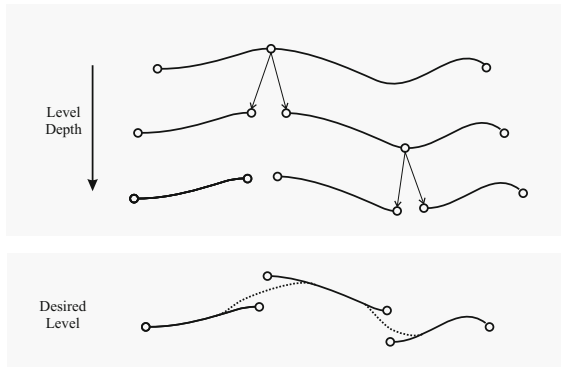
where  $\tau_{i,j}$  denotes the time between two successive nodes and  $\tau_i$  is the duration of a single node, which is defined by

$$\tau_i = \frac{n_i}{p_i} \Delta t, \tag{7}$$

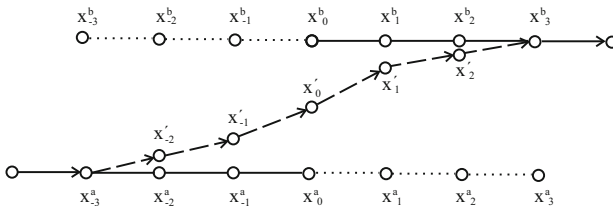
where  $1/\Delta t$  is the sampling frequency of example movements. In Eq. (7),  $n_i$  denotes the number of state vectors clustered in node  $i$  and  $p_i$  the number of trajectories passing through node  $i$ .

If our process found non-connected path parts at the desired level, we use optimized interpolation to generate a complete movement. We interpolate a certain number of state vectors belonging to the nodes around the two break nodes (see Fig. 2)

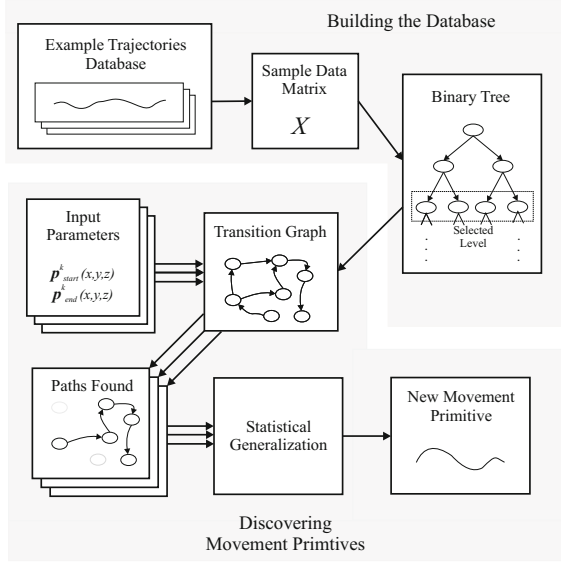
$$\mathbf{x}'_i = \lambda_i \mathbf{x}_i^a + (1 - \lambda_i) \mathbf{x}_i^b. \tag{8}$$



**Fig. 1.** Finding non-connected path parts at the desired level. Nodes are denoted by circles, while the connections are shown with lines. Dashed lines denote interpolated transitions.



**Fig. 2.** Transition by interpolation. Nodes are represented by circles and connections by lines. Solid line denotes found connections i.e. parts of the path at the desired level. Dashed lines connect interpolated nodes i.e. state vectors. Nodes that are needed for interpolation but are not on our path parts are connected by dotted lines.



**Fig. 3.** Process of discovering new movement primitives. Block diagram represents our process of building the database and then using it to generate new movement primitives that were not present in the example database.

The number of nodes we choose to interpolate is a compromise between the smoothness of the transition and the preservation of original paths. Non-linear local optimization algorithm BOBYQA [12] is used to find weights  $\lambda$  and time durations of individual nodes in such a way as to minimize the jerk of the transition trajectory.

After specifying a number of such start and end points, we end up with a set of new example trajectories  $\mathbf{M}_k$ . They are encoded by state vectors associated with the graph nodes and the start and end configurations on the trajectory:

$$\mathbf{M}_k = \{ \{ \mathbf{x}_i^k, t_i^k \}_{i=1}^{n_k}, \mathbf{p}_{start}^k, \mathbf{p}_{end}^k \}. \quad (9)$$

The start and end configurations are always specified in task space and are the same as the first and last state vectors on the path if examples are given in task space, whereas they are related to these configurations via forward kinematics if the example trajectories (and consequently state vectors) are given in joint space. Even if a large database of example movements is available, it is highly unlikely – at least in the case of manipulation tasks – that an exact desired movement can be found via direct graph search. To accomplish a task such as reaching towards an object, we need to generalize the discovered example movements to new reaching configurations. This can be accomplished by means of statistical generalization, where a movement suitable for the current object configuration is synthesized from a number of example trajectories. The complete discovery process is illustrated in Fig. 3.

For execution of a discrete movement on a robot we represent the trajectory of every degree of freedom by a dynamic movement primitive (DMP) [4]. Given the DMP representation (see appendix for a brief review of DMP representation) and trajectory data [9], Ude et al. [16] showed how to generalize the example trajectories to any combination of start and end configurations  $\{\mathbf{p}_{start}, \mathbf{p}_{end}\}$  within the trajectory training space. Essentially, they proposed a methodology to learn a function

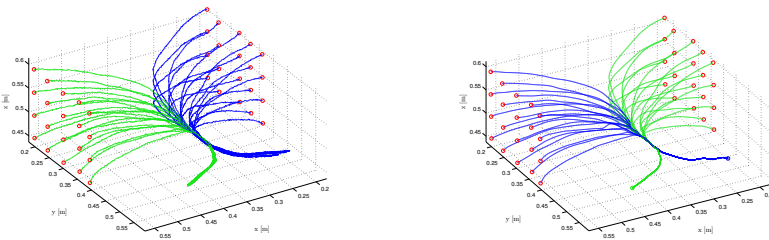
$$\mathbf{F}(\{\mathbf{M}_k\}) : [\mathbf{p}_{start}^T, \mathbf{p}_{end}^T]^T \mapsto [\mathbf{w}^T, \mathbf{g}^T, \tau]^T, \quad (10)$$

where  $\mathbf{w}$ ,  $\mathbf{g}$  and  $\tau$  are the parameters that define a DMP that starts in  $\mathbf{p}_{start}$  and ends in  $\mathbf{p}_{end}$ . Function  $\mathbf{F}$  is learned by a combination of locally weighted regression and Gaussian process regression. We omit the details. What is important for us here is that for this method to work, the discovered trajectories  $\mathbf{M}_k$  need to transition smoothly between each other as a function of start and end configuration  $\mathbf{p}_{start}$  and  $\mathbf{p}_{end}$ . It is the task of the graph search process to find paths through the transition graph that fulfill this condition.

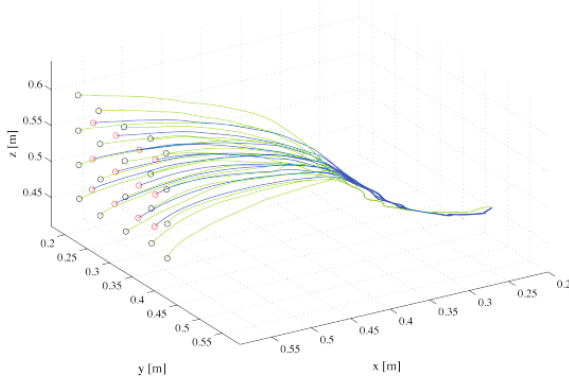
## 4 Evaluation

Our process was evaluated with two experiments. First experiment was done in a simulation with a set of constructed movements with position end-effector trajectories. The second used sets of movements captured through kinesthetic guidance of a robot arm.

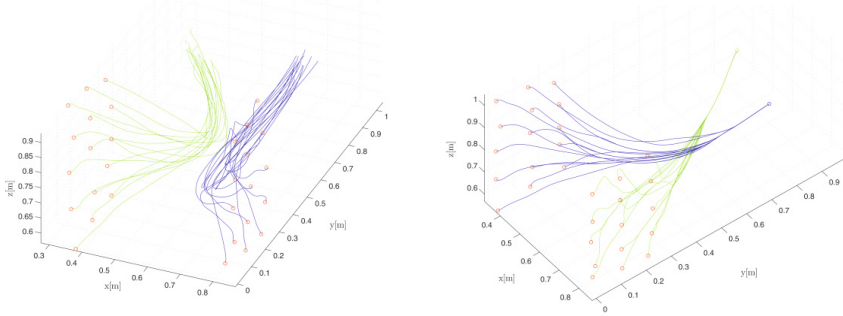
In a simulation experiment we constructed a database using two sets of 20 reaching movements. In both sets all movements have approximately the same starting point, but different ending points. Ending points cover a plane at roughly 5 cm intervals. The end-effector trajectories in task space can be seen in Fig. 4.



**Fig. 4.** Two sets of reaching movements used to construct the database. Every set of reaching movement has roughly the same starting point but 20 different end points, as shown on the left figure. Trajectories from one set are in green, from the other in blue. For better visualization, the end points are marked with circles. Two new sets of reaching trajectories that were discovered by graph search, but were not present in the original example database. They are shown on the right figure. One set of movements is shown in green, the other in blue. End points are again marked in red.

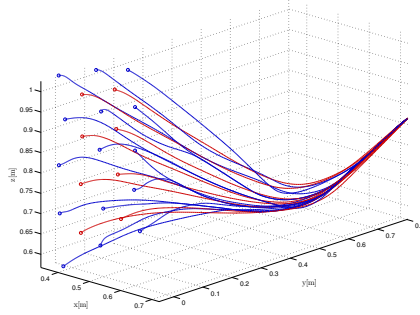


**Fig. 5.** Generalization of new reaching movements. The newly found trajectories, which were discovered by graph search, are shown in green with end points shown as black circles. Trajectories towards in-between reaching configurations (red circles) computed by statistical generalization (see Eq. (10)) are shown in blue. Note that the generalized trajectories preserve the shape of the initial reaching movements. A new movement primitive has been discovered and learned.



**Fig. 6.** Two sets of reaching movements, which were acquired by kinesthetic guiding of the Kuka Lightweight Robot arm. Every set of reaching movement has roughly the same starting point but 15 different end points, as shown on the left figure. Trajectories from one set are in green, from the other in blue. Two new sets of reaching trajectories that were discovered by our procedure, but were not present in the original example database. They are shown in the right figure. One set of movements is shown in green, the other in blue. End points are again marked in red.

From these two sets of movements (altogether 40 trajectories) we constructed a database as described in Section 2. The resulting binary tree had 20 levels and 1862 nodes at the deepest level. Using the transition graph associated with the deepest level of the binary tree, we generated new movements by selecting a starting point in the first set of reaching movements and the ending point in the second set. Due to the structure of the training data, there always existed a path between these two points. Our graph search procedure was able to generate



**Fig. 7.** Generalization of new reaching movements. The newly found trajectories, which were discovered by our procedure, are shown in blue with end points shown as blue circles. Trajectories towards in-between reaching configurations (red circles) computed by statistical generalization (see Eq. (10)) are shown in red. Note that the generalized trajectories preserve the shape of the new reaching movements, which are themselves obtained by concatenating the original trajectories acquired by kinesthetic guiding.

two new sets of movement, where each of the new movements starts in one set of reaching movements and ends in the second set (see Fig. 4). In this way we could generate two new sets of reaching trajectories that were not part of the initial database. It is especially important that the newly generated movements smoothly transition from one to another because this is a precondition for successful generalization. The results of the generalization process are shown in Fig. 5. Only the end points were used as input parameters for generalization because the starting points were the same over the complete set of example movements. Note that the generalized trajectories have a similar shape as the ones discovered by graph search. In our second experiment we constructed a database using two sets of 15 reaching movements acquired by kinesthetic guiding on anthropomorphic Kuka Light Weight Robot arm. Again all movements have approximately the same starting point, but different ending points in each set. Ending points cover a plane at roughly 10 cm intervals. The end-effector trajectories in task space can be seen in Fig. 6.

From these two sets of movements (altogether 30 trajectories) we constructed a database as described in Section 2. The resulting binary tree had 17 levels and 2954 nodes at the deepest level. As in previous experiment, we generated new movements by selecting a starting point in the first set of reaching movements and the ending point in the second set. Since we chose the deepest level of the binary tree, new movements could not be generated only through path search in transition graph as the two sets did not share any nodes at that level. That can also be observed in Fig. 6, where it is seen that our sets do not have any common parts. Despite of that, two new sets of movement were generated by our procedure (see Fig. 6). As in the first experiment every new movement starts in one set of reaching movements and ends in the second set. Although the initial sets of movements were not as carefully constructed as in the previous experiment,

**Table 1.** Properties of the constructed database

	Database 1	Database 2
No. of example trajectories	40	30
No. of state vectors	23536	8877
No. of levels	20	17
No. of nodes at the deepest level	1862	2954

newly generated movements smoothly transition from one to another, which is a precondition for a successful generalization. The results of the generalization process are shown in Fig. 7. Note the similar shape of the generalized trajectories in regards to those discovered by graph search. At the end we executed new sets of smooth movements on our LWR Kuka robot arm.

## 5 Conclusion

While more experiments need to be conducted to fully evaluate the proposed approach, the first experiments presented here demonstrate that new movement primitives can be discovered and learned from a database of training movements using a combination of graph search, optimized interpolation and statistical generalization. Unlike many other approaches in imitation learning, we assume that the training database consists of different types of movements, which we combine to new movements by exploiting the interconnections between them. This approach does not need a direct connection between desired points at the selected level of granularity to generate new sets of example movements that can be used for generalization. In this way a complete representation for the new movement primitive can be calculated.

**Acknowledgement.** The research leading to these results has received funding from the European Communitys Seventh Framework Programme FP7/20072013 (Specific Programme Cooperation, Theme 3, Information and Communication Technologies) under grant agreement no. 270273, Xperience.

## Appendix - Dynamic Movement Primitives

DMPs are defined by the following nonlinear system of differential equations

$$\tau\dot{v} = K(g - y) - Dv + f(x), \quad (11)$$

$$\tau\dot{y} = v. \quad (12)$$



The linear part of Eq. (11) – (12) ensures that  $y$  converges to the desired final configuration, denoted as  $g$ . The nonlinear part  $f(x)$  modifies the shape of a movement and is defined by a linear combination of radial basis functions

$$f(x) = \frac{\sum_{i=1}^N w_i \psi_i}{\sum_{i=1}^N \psi_i} \quad (13)$$

$$\psi_i = \exp(-h_i(x - c_i)^2), \quad (14)$$

where  $\psi_i$  defines the basis functions, with centres at  $c_i$  and widths  $h_i > 0$ . As seen in Eq. (13),  $f(x)$  is not directly time dependent. Instead, phase variable  $x$  defined in Eq. (15), with initial value  $x(0) = 1$ , is used to make the dependency more implicit:

$$\tau \dot{x} = -\alpha_x x \quad (15)$$

The phase is common across all DOF. By specifying the time evolution through phase, it becomes easier to stop the clock in case of external perturbations which cause the robot to deviate from the desired trajectory. It can be shown that – given the properly defined constants  $K$ ,  $D$ ,  $\tau$ ,  $\alpha_x > 0$  – the above system is guaranteed to converge to the desired final configuration  $g$ .

## References

1. Breazeal, C., Scassellati, B.: Robots that imitate humans. *Trends Cognitive Sci.* 6(11), 481–487 (2002)
2. Calinon, S., D’halluin, F., Sauser, E.L., Caldwell, D.G., Billard, A.G.: Learning and reproduction of gestures by imitation: An approach based on Hidden Markov Model and Gaussian Mixture Regression. *IEEE Robot. Autom. Mag.* 17(2), 44–54 (2010)
3. Dillmann, R.: Teaching and learning of robot tasks via observation of human performance. *Robotics and Autonomous Systems* 47(2-3), 109–116 (2004)
4. Ijspeert, A.J., Nakanishi, J., Schaal, S.: Movement imitation with nonlinear dynamical systems in humanoid robots. In: *Proc. IEEE Int. Conf. Robot. Autom.*, Washington, DC, pp. 1398–1403 (2002)
5. Kittler, J., Illingworth, J.: Minimum error thresholding. *Pattern Recognition* 19(1), 41–47 (1986)
6. Kovar, L., Gleicher, M., Pighin, F.: Motion graphs. *ACM Transactions on Graphics* 21(3), 473–482 (2002)
7. Kulić, D., Takano, W., Nakamura, Y.: Online segmentation and clustering from continuous observation of whole body motions. *IEEE Trans. Robot.* 25(5), 1158–1166 (2009)
8. Kulić, D., Takano, W., Nakamura, Y.: Incremental learning, clustering and hierarchy formation of whole body motion patterns using adaptive hidden Markov chains. *Int. J. Robot. Res.* 27(7), 761–784 (2008)
9. Lim, B., Ra, S., Park, F.C.: Movement Primitives, Principal Component Analysis, and the Efficient Generation of Natural Motions. In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation, ICRA 2005*, pp. 4630–4635 (2005)

10. MacQueen, J.B.: Some methods for classification and analysis of multivariate observation. In: Proc. of the 5th Berkley Symposium on Mathematical Statistics and Probability, pp. 281–297 (1967)
11. Peters, J., Mülling, K., Kober, J., Nguyen-Tuong, D., Krömer, O.: Towards Motor Skill Learning for Robotics. In: Pradalier, C., Siegwart, R., Hirzinger, G. (eds.) Robotics Research. Springer Tracts in Advanced Robotics, vol. 70, pp. 469–482. Springer, Heidelberg (2011)
12. Powell, M.J.D.: The BOBYQA algorithm for bound constrained optimization without derivatives, Department of Applied Mathematics and Theoretical Physics, Cambridge England, technical report NA2009/06 (2009)
13. Rose, C., Cohen, M.F., Bodenheimer, B.: Verbs and adverbs: multidimensional motion interpolation. *IEEE Computer Graphics and Applications* 18(5), 32–40 (1998)
14. Safonova, A., Hodgins, J.K.: Construction and optimal search of interpolated motion graphs. *ACM Transactions on Graphics* 26(3) (2007)
15. Sidenbladh, H., Black, M.J., Sigal, L.: Implicit Probabilistic Models of Human Motion for Synthesis and Tracking. In: Heyden, A., Sparr, G., Nielsen, M., Johansen, P. (eds.) ECCV 2002, Part I. LNCS, vol. 2350, pp. 784–800. Springer, Heidelberg (2002)
16. Ude, A., Gams, A., Asfour, T., Morimoto, J.: Task-specific generalization of discrete and periodic dynamic movement primitives. *IEEE Trans. Robotics and Autom.* (5), 800–815 (2010)
17. Yamane, K., Yamaguchi, Y., Nakamura, Y.: Human motion database with a binary tree and node transition graphs. In: Presented at the Robot., Sci. Syst. Conf., Seattle, Washington (2009)

# Model Based Visual Servoing Tasks with an Autonomous Humanoid Robot\*

Amine Abou Moughlbay<sup>1</sup>, Enric Cervera<sup>2</sup>, and Philippe Martinet<sup>3</sup>

<sup>1</sup> Institut de Recherche en Communications et Cybernétique de Nantes,  
Ecole Centrale de Nantes, 1 rue de la noë, 44321 Nantes - France,  
Amine.Abou-Moughlbay@ec-nantes.fr

<sup>2</sup> Robotic Intelligence Lab, Jaume-I University - Spain  
Ecervera@icc.uji.es

<sup>3</sup> IRCCyN, Ecole Centrale de Nantes and Institut Pascal in Clermont Ferrand - France  
Philippe.Martinet@ircryn.ec-nantes.fr

**Abstract.** Many model based techniques have been proposed in the literature for applying domestic service tasks on humanoid robots, such as teleoperation, learning from demonstration and imitation. However sensor based robot control overcomes many of the difficulties of uncertain models and unknown environments which limit the domain of application of the previous methods. Furthermore, for service and manipulation tasks, it is more suitable to study the interaction between the robot and its environment at the contact point using the sensor based control, rather than specifying the joint positions and velocities required to achieve them.

In this work we present an integration of real-time visual servoing techniques on a humanoid robot in closed loop, to perform self-localization and different manipulation tasks. Indeed, real-time model based tracking techniques are used to apply 3D visual servoing tasks on the Nao humanoid robot. The elementary tasks which are used by the robot to perform a concrete service scenario are detailed with their corresponding control laws. Finally, we present the experimental results of the following tasks: self-localization of the robot while walking, head servoing for the visibility task, detection, tracking and manipulation of environment's objects.

## 1 Introduction

To perform manipulation tasks, the robot interacts with its environment through contact points, such as applying forces and moments on objects. By controlling the position and velocity of these points, as well as the forces acting on them, the robot performs the desired tasks. Furthermore, in complicated and uncertain environments, end-effector motion may be subject to online modifications in order to accommodate unexpected events or to respond to sensor inputs. Thus programming of service and manipulation tasks is

---

\* This research has been partially funded by the French National Agency of Research under the reference ANR-10-SEGI-002, and partially by Ministerio de Ciencia e Innovación (DPI2011-27846), Generalitat Valenciana (PROMETEO/2009/052) and Fundació Caixa-Castelló-Bancaixa (P1-1B2011-54).

most conveniently accomplished by directly specifying data at the contact points (using operational space control), rather than specifying the joint positions and velocities required to achieve them (using joint space control) [1].

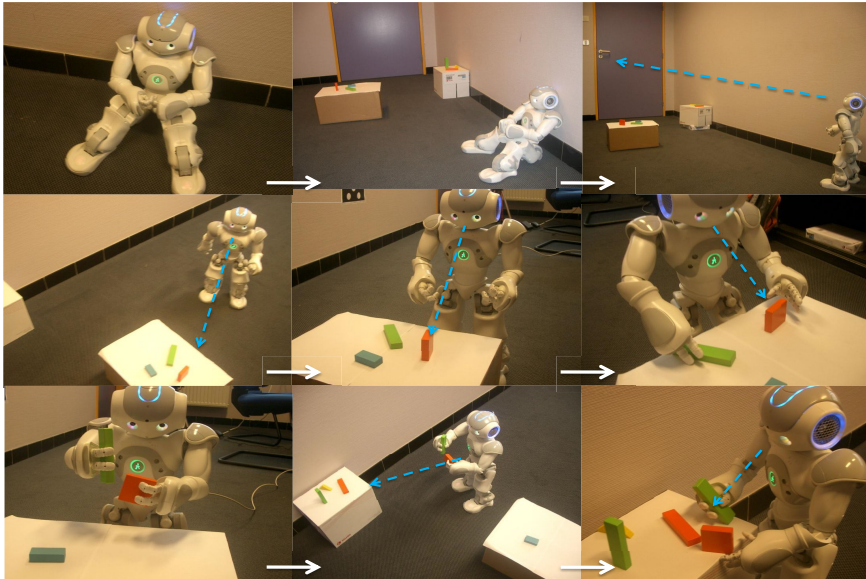
Many approaches have been used to apply service tasks: several works have been carried out in the area of teleoperation, by controlling a robotic system to perform tasks at a distance using a multi-modal human-system interface which provides sensory feedback to the operator and allows him to interact with the remote environment by mapping his actions. It was applied to bi-manual manipulation and walking [2], haptic interface for mobile teleoperators [3] and recently on a teleoperation system with haptic device for micro-manipulation [4].

Furthermore, many practical learning control systems are used to control complex robots involving multiple feedback sensors and multiple command variables during both repetitive and nonrepetitive operations [5]. The issue of teaching a robot to manipulate everyday objects through human demonstration has been studied by [6] who proposed a method that enables a robot to decompose a demonstrated task into sequential manipulation primitives, series of sequential rotations and translations [7]. Other earlier works used also the Model Based Control strategies, executed by automatically generating a control sequence that moves the robot to the states specified by the program to develop executives that emphasize model based approaches and deep integration of automated planning [8].

On the other hand, self-localization of service robots is one of the fundamental problems in robotics, as in many applications a robot needs to know its location in order to perform its tasks. Most of the indoor localization algorithms use particle based filters or Kalman type filters to solve the problem of noisy sensors and controls. Particle filters inherently help solve the problem of ambiguous landmarks, whereas Kalman filters must track multiple hypotheses to work in ambiguous environments [9]. In some divisions of RoboCup, algorithms are very well established, given the rich sensor data provided by laser scanners, omni-directional cameras etc. However, in more general cases, there are substantial sensor limitations particularly with the rapid motion of the camera, and the need for active perception [10]. In addition to odometry sensors, some recent alternatives use also the ambient magnetic field to control the heading of a robot in case of one-dimensional localization problem [11].

As previously presented, most actuated systems use sensors to obtain information about their environment. These can be a camera, ranging devices, or temperature and force sensors. Among all these feedbacks, the visual information provides the most important and instant cues for perception of the interaction with the working environment. Compared to already mentioned methods, visual servoing provides very efficient solutions to control robot motions. It supplies high positioning accuracy, good robustness to sensor noise and calibration uncertainties, and reactivity to environment changes [12].

In this work, only Sensor Based Control formalism is used to perform the desired tasks. More especially, 3D visual feedback data and Model Based Tracking (MBT) techniques are used to execute, in real-time and closed loop, many tasks on the humanoid mobile robot Nao in a semi structured environment. The robot's camera is calibrated, and a rough geometric model of the objects is available (doors, tables, pieces to grasp ...). The envisioned scenario consists of a Nao robot which can carry out service tasks,



**Fig. 1.** Nao picks up the orange and green pieces and deposits them on the table near the door

moving around the room and manipulating objects. One of the missions that can be requested from such robot would be to “Pick up the orange and green pieces and deposit them on the table nearby the door” (see Fig. 1). But to execute this mission, many problems should be addressed and resolved:

- How can the robot localize itself with respect to its environment?
- How to decompose the robot’s mission into elementary tasks?
- What information should be given to the robot to execute each task?
- What are the more useful techniques the robot can use to perform these tasks?

In the next section, we present the architecture of the Nao robot and the Model Based Tracking technique. In section 3, we define the control laws of the common elementary tasks employed during the application of the desired tasks. Implemented tasks are presented in section 4 and the experimental results in section 5. The first part of this latter includes the localization of the robot in its environment when walking. The second one includes the simultaneous execution of object tracking, head control and manipulation by visual servoing. The final section draws some conclusions and outlines future works.

## 2 System Architecture

### 2.1 Nao Architecture

Nao Robot [13], developed by Aldebaran robotics, is a (57 cm, 5.2 kg) biped robot with 25 Degrees of Freedom (DOF). It has 3-fingered robotic hands used for grasping and

holding small objects (it can carry up to 300 g using both hands). Nao is equipped with: two ultrasound devices situated in the chest that provide space information in a range of 1 meter, two cameras situated on the top and bottom of the head, two bumpers (contact sensors on the robot's feet), a gyrometer and an accelerometer (to determine whether the robot is in a stable or unstable position).

## 2.2 Visual Servoing and Tracking Techniques

A large variety of positioning or target tracking tasks can be implemented by controlling from one to all DOF of the system. For whatever the sensor configuration, which can vary from one camera mounted on the robot end-effector to several free-standing cameras, a set of visual features  $\mathbf{s}$  has to be designed from the visual measurements obtained from the system configuration  $\mathbf{x}(t)$ , allowing control of the desired DOF.

A control law is thus designed so that these features  $\mathbf{s}$  reach a desired value  $\mathbf{s}^*$ , defining a correct realization of the task. Indeed, if the camera velocity  $\mathbf{V}_c$  is considered as input of the robot controller, the control law which performs the desired exponential decoupled decrease of the error  $\mathbf{e} = (\mathbf{s} - \mathbf{s}^*)$  is given by:

$$\mathbf{V}_c = -\lambda \mathbf{L}_s^+ (\mathbf{s} - \mathbf{s}^*) \quad (1)$$

where  $\lambda$  is the classical proportional gain that has to be tuned to minimize the time to convergence of the task, and  $\mathbf{L}_s^+$  is the Moore-Penrose pseudo-inverse of the interaction matrix  $\mathbf{L}_s$  (for more details refer to visual servo control book [14]).

Many tracking tools have been implemented in several visual servoing toolboxes [15]. On the Visual Servoing Platform (ViSP) [16], we find a dot tracker, a moving edges tracker, and a 3D model based tracker. The last one [17] tracks a 3D model thanks to the moving edges method, using a virtual visual servoing technique. It requires a 3D model and needs to compute the initial pose which is used to project the model on the image. The tracking method assumes that the pose corresponding to the previous image is known, the new lines are tracked, and the goal is to move the pose to match the object in the new image with the projection of the model.

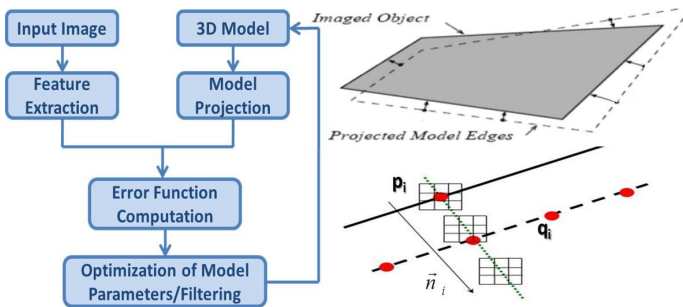


Fig. 2. Model Based tracking system (left) using the Moving edge detection technique (right)

The following error function ( $err$ ) between image features  $p_i$  and model projection  $q_i$  is thus minimized along the normal direction  $n$  (see Fig. 2):

$$err = \sum_i \Delta(p_i, q_i) = \sum_i |(q_i - p_i) \cdot (n_i)| \quad (2)$$

### 3 Control Law and Tasks Identification

To execute the desired tasks, we should first define several coordinate frames on the robot's body and environment's items as represented in Fig. 3. And using the inverse kinematic model of the robot and the 3D visual servoing technique (1) detailed above, the general control law which is used to define a generic task can be written:

$$\dot{\mathbf{q}} = -\lambda (\mathbf{L}_s \mathbf{J})^+ (\mathbf{s} - \mathbf{s}^*) \quad (3)$$

where  $\dot{\mathbf{q}}$  is the robot's joint velocity and  $\mathbf{J}$  is the kinematic matrix which relates camera velocity with the robot's joint velocity ( $\mathbf{V}_c = \mathbf{J} \dot{\mathbf{q}}$ ).

In the rest of this section, the envisaged scenario is decomposed into simpler generic tasks to reduce the complexity of the problem to be solved. Thus, the desired scenario is executed by answering the following questions:

(a) **Where am I ? → Self-Localization Task**

Depending on the desired mission to be executed, the robot searches for the corresponding set of models in its environment to be localized with respect to them (such as table, door, light switch, corner, ...).

After detecting the items in the field of view of the robot's camera, and choosing the nearest one to the object to manipulate ( $\mathcal{F}_e$ ), the robot calculates, using the MBT technique, its position/orientation with respect to the item in form of a homogeneous transformation matrix ( ${}^n\mathbf{M}_e$ ).

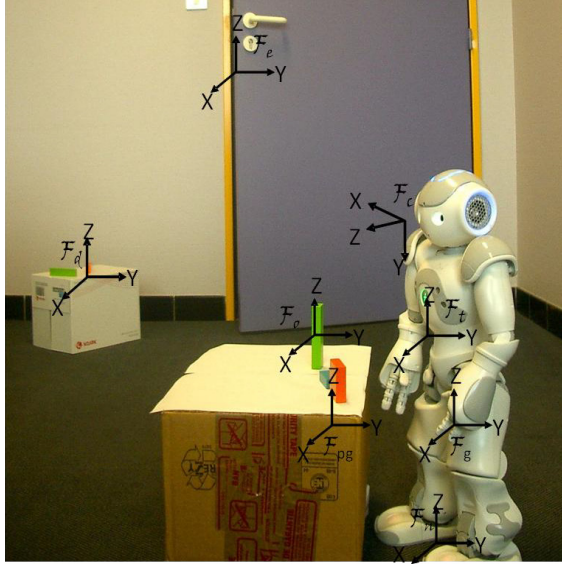
(b) **What must I do and How to do it ? → Task Scheduler**

The desired scenario is interpreted to the robot's language and decomposed into elementary tasks by the task scheduler [18]. In case of the presented scenario, the following tasks are considered:

- Localize the desired objects to manipulate
- Move in the appropriated direction
- Detect and track the desired objects
- Keep these pieces in the robot's field of view
- Move the robot's arms and grasp these pieces
- Go to the desired table and deposit objects

(c) **Which data will I use ? → Definition of used data**

To execute the defined elementary tasks, the scheduler chooses the appropriate models for each one: models of the objects to manipulate, model of the environment's items (for the self-localization task), data of the robot's camera, parameters of the robot's gripper and the used items...



**Fig. 3.** Useful frames in Nao's environment:

In Nao's body we consider the following frames:  $\mathcal{F}_n$  Nao's space frame (between robot's feet),  $\mathcal{F}_t$  on robot's torso,  $\mathcal{F}_c$  a camera attached frame,  $\mathcal{F}_h$  robot's hand frame,  $\mathcal{F}_g$  robot's gripper frame and  $\mathcal{F}_{pg}$  pre-grasping frame.

The following frames are defined in the robot's environment:

$\mathcal{F}_o$  object's frame,  
 $\mathcal{F}_d$  desired object's pose frame and  
 $\mathcal{F}_e$  environment's frame for the localization task.

**(d) Where to move ? → Robot Locomotion Task**

This task is used to move to the region where the task should be executed. Knowing the position of at least one item of the environment ( ${}^nM_e$ ), the robot walks in the appropriate direction of ( $\mathcal{F}_e$ ) until entering the range of a defined distance from the object.

During this task, the robot motion is controlled in real-time and closed loop to avoid some possible modifications or unexpected events. Thus an obstacle avoidance algorithm should be used to adapt the robot's trajectory [19].

**(e) How to perceive ? → Detection and Tracking Tasks**

Using the MBT technique of ViSP presented in [22] the tracker is manually initialized and the pose of the desired item is determined. Therefore it allows us to track in real-time the pose of the object to manipulate.



Furthermore, to increase its robustness, a module for automatic re-initialization of the tracker is implemented, it uses the last poses of the tracked object to estimate the actual pose. This module is used in case of a failure due to an occlusion or the fast motions of the robot's camera (especially when walking). Moreover, the automatic transition between the tracking of different objects is also implemented using the knowledge of the rough relative position between environment's objects.

(f) **How to keep the concerned points in the robot's field of view ? → Visibility Task**

This task consists of controlling the position/orientation of the robot's head to focus a (fixed/mobile) point of the environment (item's center, gripper, virtual point...) in the center of the camera's image.

This task can be used, for example, in hand-eye coordination for dynamic grasping of objects by focusing on the gripper-item midpoint, to keep the robot's hand and object in the robot's field of view. The used DOF in this task depends on the geometry of the robot's head and the desired complexity of the task. The head's Yaw/Pitch can be simply controlled to focus on the object's center, or a more complex task can also control the distance between the robot's head and the object.

(g) **How to perform the manipulation ? → Grasping Task**

This task uses the hand's control point and allows the robot to move it to a desired static/mobile pose. This task can be used to perform pre-grasping, grasping, and displacing objects tasks.

In case of pre-grasping task, the goal position ( ${}^s\mathbf{M}_{pg}$ ) is determined using one of the grasping strategies [20]. They depend usually on the geometry of the object to manipulate and on the shape of the robot's gripper. Thus, the grasping strategy controls the relative position and/or the angle between the gripper and the item to grasp.

Likewise, for a grasping task the same technique is considered: the robot's arm moves to a desired pose by minimizing the relative distance/angle between the object and the robot's hand ( ${}^s\mathbf{M}_o$ ) in the Gripper's Frame.

## 4 Tasks Definition

In this part, we present preliminary results of the general scenario detailed in the previous section applied on the humanoid robot Nao. The experimented tasks are: the self-localization of the robot when walking, and the simultaneous control of robot's head and arm for tracking and grasping an object.

During the execution of the first task, only a rough model of the door and a part of the room is used, and for the second one only the approximate model of the item to grasp is given. Note that no other exteroceptive data is given from the robot's environment.

### 4.1 Self-Localization Task during Locomotion

Throughout this task, the robot tracks the door and thus it is localized with respect to the environment while walking. We use the model of the door and the lines of the room

around it to initialize the MBT which gives subsequently the pose of the door in the camera's frame ( ${}^c\mathbf{M}_e$ ) in real-time. From the given pose value, the robot's pose can be calculated:  ${}^e\mathbf{M}_n = ({}^c\mathbf{M}_e)^{-1} ({}^n\mathbf{M}_c)^{-1}$ .

Note that the tracker is automatically reinitialized each time the tracking failed due to large camera displacements during walking; it uses the last found pose of the object to reinitialize the tracking. In our experiments, the robot walks in open loop in the direction of the door for a distance of 1 meter.

## 4.2 Detection and Tracking Task

The MBT is used to track simple item models; it's initialized manually at the beginning of the application. Afterwards, the model is automatically detected and tracked; this tool allows us to determine instantly the pose of the desired item frame in the robot camera's frame ( ${}^c\mathbf{M}_o$ ).

## 4.3 Visibility Task

Throughout this application, the visibility task is used for controlling the robot's head orientation to focus the item's center in the midpoint of the camera's image. Two DOF are used by this task to control the head's Yaw and Pitch. The task's goal is thus to regulate exponentially ( $\dot{\mathbf{e}} = -\lambda\mathbf{e}$ ) the horizontal and vertical position of the center of the object projection  $\mathbf{s}_{x,y} = {}^c\mathbf{T}_{o(x,y)}$  to zero ( $\mathbf{s}^* = (0, 0)$ ).

Using the object 3D pose  ${}^c\mathbf{T}_o = (X, Y, Z)^T$ , and the 2D pose  $(x, y)^T$  of the tracked point (projection of 3D point in the normal image plane), we apply the control law defined in (3) using the visual primitive  $\mathbf{s} = (x, y)$  and its corresponding interaction matrix  $\mathbf{L}_s$  given by:

$$\mathbf{L}_s = \begin{bmatrix} -\frac{1}{Z} & 0 & \frac{x}{Z} & xy & -(1+y^2) & y \\ 0 & -\frac{1}{Z} & \frac{y}{Z} & 1+x^2 & -xy & -x \end{bmatrix} \quad (4)$$

Note that for this task the  $\mathbf{J}$  matrix in (3) uses the Jacobian of the robot's head control point calculated from the robot's geometric model.

## 4.4 Pre-Grasping Task

During this task the robot's arm is supposed to move close to the item to grasp. According to Nao's gripper's geometry (of one DOF) and the item's shape (rectangular model), the number of constrained DOF and the pre-grasping position is predefined in the robot's manipulation parameters. This position is defined with respect to the object's position in Nao's frame  ${}^n({}^o\mathbf{M}_{pg})$ .

In our case, regarding the gripper's and item's shapes, 4 DOF are enough to execute this task: 3 DOF constraints the gripper's pose and 1 DOF (Yaw angle) for the gripper's orientation. Furthermore, the pre-grasping distance is fixed to 5 cm, but we should not forget that the tracker gives the item's pose in real-time, thus the desired pre-grasping pose is calculated in closed loop.

The task's target is then to move the robot's arm to the pre-grasping pose. The task's error is extracted from the relative pose between the gripper and pre-grasping point ( ${}^s\mathbf{M}_{pg}$ ) which is regulated to zero. Note that  ${}^s\mathbf{M}_{pg} = ({}^n\mathbf{M}_g)^{-1} {}^n\mathbf{M}_c {}^c\mathbf{M}_o {}^o\mathbf{M}_{pg}$  where  ${}^n\mathbf{M}_g$  and  ${}^n\mathbf{M}_c$  are given by the robot's proprioceptive sensors.

Considering that the visual primitive is parameterized by  $\mathbf{s} = (\mathbf{t}, \mathbf{u}\theta)$  where  $\mathbf{t}$  is the position error between the current and desired frame, while  $\mathbf{u}\theta$  is the orientation error, decomposed as the axis  $\mathbf{u}$  and angle  $\theta$  of the rotation between these two frames. The control law (3) is then applied using the Jacobian at the robot's gripper and the corresponding interaction matrix  $\mathbf{L}_s$  given by:

$$\mathbf{L}_s = \begin{bmatrix} -\mathbf{I}_3 & [\mathbf{t}]_{\times} \\ \mathbf{0}_3 & \mathbf{L}_\omega \end{bmatrix} \quad (5)$$

where  $\mathbf{I}_3$  and  $\mathbf{0}_3$  are the  $3 \times 3$  identity and zero matrices respectively, the  $\mathbf{L}_\omega$  matrix is given by  $\mathbf{L}_\omega = \mathbf{I}_3 - \frac{\theta}{2} [\mathbf{u}]_{\times} + \left(1 - \frac{\text{sinc}(\theta)}{\text{sinc}^2(\theta/2)}\right) [\mathbf{u}]_{\times}^2$ , and  $[\mathbf{t}]_{\times}$  is the skew symmetric matrix associated with vector  $\mathbf{t}$ .

#### 4.5 Grasping Task

Along this task the robot's arm moves to grasp the desired item. The same number of DOF is constrained and the same technique is used to define the grasping task as in the previous case of pre-grasping, but the desired gripper pose is changed to the object's pose. Thus the task's error will be extracted from the relative pose between the gripper and the item  ${}^s\mathbf{M}_o$  which is also regulated to zero (this pose is calculated using the relation:  ${}^s\mathbf{M}_o = ({}^n\mathbf{M}_g)^{-1} {}^n\mathbf{M}_c {}^c\mathbf{M}_o$ ).

After arriving to the desired pose, the robot's gripper closes to catch the item. We should note that the visibility and pre-grasping tasks are executed in parallel, and once the pre-grasping finished, the grasping task is executed automatically.

Furthermore, a task is completed when the error norm reaches a predefined threshold value. This threshold varies with respect to the executed task: during the grasping task of small objects, a high precision is necessary unlike in the case of pre-grasping task. In our experiments, the threshold is predefined to 5 mm in the camera's image for the visibility task. In case of pre-grasping and grasping tasks, the precision is predefined to 5 mm and 1 mm respectively and 3 degrees for the orientation.

## 5 Experimental Results

The presented tasks in the previous section have been implemented and tested several times on the Humanoid Nao robot to ensure the efficiency of this method [9]. During these experiments, we used the two cameras embedded on the robot's head: the top one for localization and the bottom one for manipulation tasks. The control rate of the robot is equal to that of the camera (20 Hz).

<sup>1</sup> See a video of the applied tasks on Nao robot on [www.youtube.be/Wf1A\\_jRBMkM](http://www.youtube.be/Wf1A_jRBMkM)

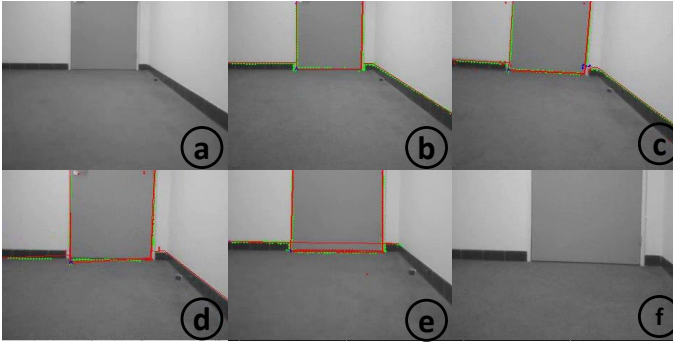


Fig. 4. Experiment photos of self-localization task during robot’s locomotion

### 5.1 Self-Localization Task Results

This part corresponds to the task defined in section 4.1 above. Fig. 4-a shows the robot’s environment before launching the tracking and locomotion tasks. Then Fig 4-b...e show the tracking of the door during the task’s execution, and when the task is completed (Fig. 4-f).

In Fig. 5, the distance between the robot’s frame  $\mathcal{F}_n$  and the origin of the door’s frame  $\mathcal{F}_e$  is plotted, in addition to the X, Y and Z components of this pose in  $\mathcal{F}_n$ . We can notice that the distance decreases from 3.94 m to 2.88 m. These results show then that the robot successfully tracks the door while walking the desired distance (1 m) with an final error of 6 cm, which allow us to localize the robot successfully.

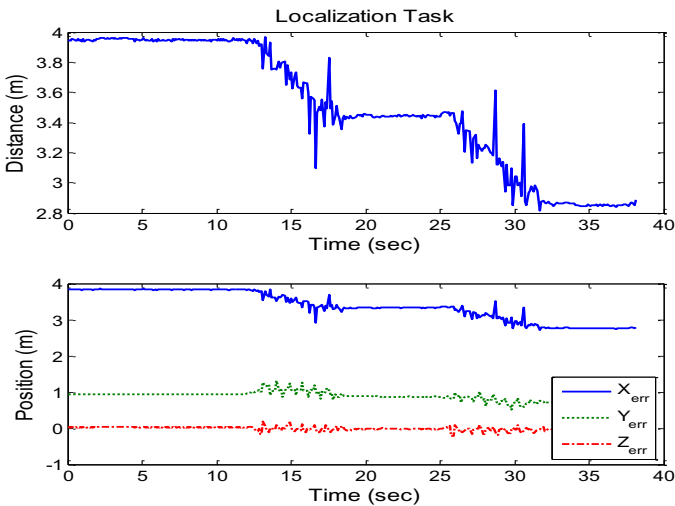


Fig. 5. Experimental results of the self-localization task in  $\mathcal{F}_n$  frame

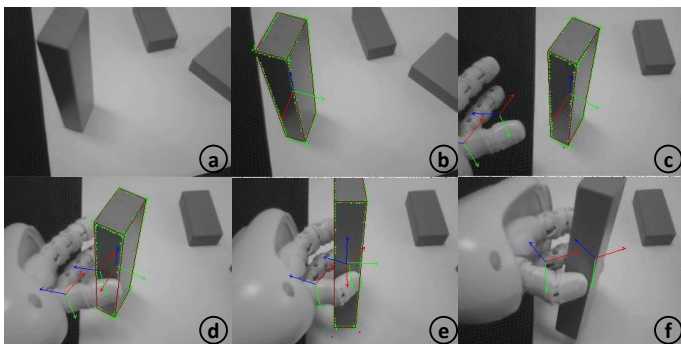


Fig. 6. Photos of tracking and grasping tasks showing the arm's, gripper's and object's frames

## 5.2 Visibility and Grasping Task Results

Experiment photos of the visibility and grasping tasks executed by Nao robot are presented in Fig. 6 and correspond to the tasks introduced in sections 4.2, 4.5 above. Fig. 6-a, shows the item to grasp before launching the MBT to detect and track it (Fig. 6-b). The visibility task is used to center the object on the camera's image, and the pre-grasping task is executed in Fig. 6-c, where we can identify the different frames on the robot's arm and gripper in addition to the object's frame. Afterwards, the gripper's frame approaches the object's when executing the grasping task (Fig. 6-d). Finally the gripper closes and the manipulation task is completed (Fig. 6-e-f).

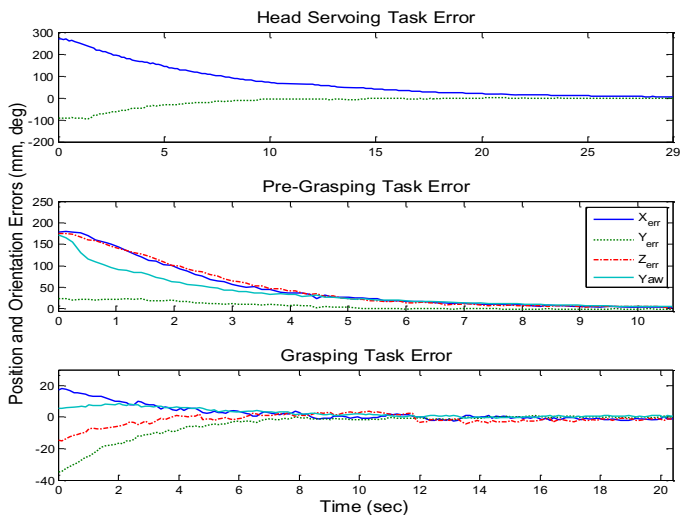


Fig. 7. Experimental results: Visibility task, Pre-Grasping task and Grasping task errors

The variation of the error in each task is presented in Fig. 7: the first graph represents the horizontal and vertical position error during the head servoing task (visibility task), initially the object is at a distance of approximately 30 cm from the center of the camera's image, we remark that this error is successfully regulated to zero during 29 sec with a precision of 5 mm.

For the 2<sup>nd</sup> and 3<sup>rd</sup> graphs, we represent the variation of pre-grasping and grasping tasks errors on X, Y and Z components (in Nao's frame), and the Yaw angle of the gripper orientation: the robot's hand is initially at an approximate distance of 25 cm from the predefined pre-grasping position, and the gripper is rotated of 180 deg with respect to the object. During this task, the position error and the Yaw angle are regulated exponentially to zero during 31 sec. Finally, we recall that these (pre) grasping tasks are successfully executed with a precision of 5 mm and 1 mm respectively and 3 deg for the orientation.

To ensure the robustness of the proposed visual servoing technique for manipulation tasks and the efficiency of the used control laws. The experiment on the previously presented tasks (tracking, head servoing and object grasping) have been successfully repeated 40 times with the same initial conditions. The average of the required convergence time for each task has been also calculated: 26.3 sec for visibility task and 28.4 sec for object manipulation task.

## 6 Discussions and Conclusions

In this study, we presented a concrete scenario of a humanoid mobile robot executing self-localization task while walking and manipulation tasks in an everyday life environment, and we detailed the elementary tasks used by the robot to perform this scenario. The experimental results point out the possibility and efficiency of using the MBT techniques to apply real-time 3D visual servoing on a simple humanoid robot (Nao) in case of localization and manipulation tasks.

For the self-localization task, the error may be relatively large with respect to other localization methods, but it is acceptable and sufficient when dealing with indoor locomotion for manipulation tasks, because of the high robustness of the MBT technique and the implemented automatic re-initialization of the tracking process. In Fig. 5 between the 10<sup>th</sup> and 32<sup>nd</sup> sec, we can identify some disturbances which are caused by the displacement of the camera during the robot's locomotion. During this period, the automatic re-initialization module of the tracking is used to prevent localization task from failure. To improve this method, an additional module could be used to dynamically compensate the camera's motions during robot's walking (using data from robot's accelerometer/gyrometer), or by fusion with other sensors data; for example using an external camera on the ceiling of the room to provide a wider view and more visual data, or the Nao's odometry module.

For the manipulation part, the results of the tracking and grasping tasks, which were tested several times, show that this method is robust to camera occlusion by the robot's hand, and robust to slight object movement due to hand-object collision. These tasks should be improved and tested on other robots. Indeed, the mechanic and software architecture of Nao is very constraining, especially for the small camera field of view, the

inability of simultaneous use of the two cameras and the constrained operational space of the hands. For this, future works will concentrate on the improvement and implementation of this method on other platforms with different objects of complex shapes with mobile or articulated elements. Other sensors' feedbacks can also be used to improve the manipulation robustness and reactivity against dynamic or unusual changes in the environment.

## References

1. Brock, O., Kuffner, J., Xiao, J.: Motion for manipulation tasks. In: Siciliano, B., Khatib, O. (eds.) *Springer Handbook of Robotics*, pp. 615–645. Springer, Heidelberg (2008)
2. Evrard, P., Mansard, N., Stasse, O., Kheddar, A., Schau, T., Weber, C., Peer, A., Buss, M.: Intercontinental, multimodal, wide-range telecooperation using a humanoid robot. In: *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems - IROS 2009*, pp. 5635–5640 (2009)
3. Nitzsche, N., Schmidt, G.: A mobile haptic interface mastering a mobile teleoperator. In: *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems - IROS 2004*, vol. 4, pp. 3912–3917 (September–October 2004)
4. Houston, K., Sieber, A., Eder, C., Vittorio, O., Menciassi, A., Dario, P.: A teleoperation system with novel haptic device for micromanipulation. *Int. Journal of Robotics and Automation* 26(3) (2011)
5. Miller, W.T.: Sensor-based control of robotic manipulators using a general learning algorithm. *IEEE Journal of Robotics and Automation* 3(2), 157–165 (1987)
6. Dang, H., Allen, P.: Robot learning of everyday object manipulations via human demonstration. In: *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems - IROS 2010*, pp. 1284–1289 (October 2010)
7. Jain, A., Kemp, C.: Pulling open novel doors and drawers with equilibrium point control. In: *IEEE-RAS Int. Conf. on Humanoid Robots*, pp. 498–505 (December 2009)
8. Mansard, N., Stasse, O., Chaumette, F., Yokoi, K.: Visually-guided grasping while walking on a humanoid robot. In: *IEEE Int. Conf. on Robotics and Automation*, pp. 3041–3047 (April 2007)
9. Quinlan, M.J., Middleton, R.H.: Multiple Model Kalman Filters: A Localization Technique for RoboCup Soccer. In: Baltes, J., Lagoudakis, M.G., Naruse, T., Ghidary, S.S. (eds.) *RoboCup 2009*. LNCS, vol. 5949, pp. 276–287. Springer, Heidelberg (2010)
10. Billington, D., Estivill-Castro, V., Hexel, R., Rock, A.: Using Temporal Consistency to Improve Robot Localisation. In: Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.) *RoboCup 2006*. LNCS (LNAI), vol. 4434, pp. 232–244. Springer, Heidelberg (2007)
11. Haverinen, J., Kempainen, A.: Global indoor self-localization based on the ambient magnetic field. *Int. Journal of Robotics and Autonomous Systems* 57(10), 1028–1035 (2009)
12. Thuilot, B., Martinet, P., Cordesses, L., Gallice, J.: Position based visual servoing: keeping the object in the field of vision. In: *IEEE Int. Conf. on Robotics and Automation*, vol. 2 (2002)
13. Gouaillier, D., Hugel, V., Blazevic, P., Kilner, C., Monceaux, J., Lafourcade, P., Marnier, B., et al.: Mechatronic design of Nao humanoid. In: *IEEE Int. Conf. on Robotics and Automation*, pp. 769–774 (May 2009)
14. Chaumette, F., Hutchinson, S.: Visual servo control. i. basic approaches. *IEEE Robotics Automation Magazine* 13(4), 82–90 (2006)
15. Cervera, E.: A Cross-Platform Network-Ready Visual Servo Simulator. In: *IEEE Int. Conf. on Intelligent Robots and Systems (IROS 2006)*, Beijing (China), pp. 2314–2319 (2006)

16. Marchand, E., Spindler, F., Chaumette, F.: Visp for visual servoing: a generic software platform with a wide class of robot control skills. *IEEE Robotics Automation Magazine* 12, 40–52 (2005)
17. Comport, A., Marchand, E., Pressigout, M., Chaumette, F.: Real-time markerless tracking for augmented reality: the virtual visual servoing framework. *IEEE Transactions on Visualization and Computer Graphics* 12(4), 615–628 (2006)
18. Galindo, C., Fernandez-Madrigal, J.-A., Gonzalez, J., Saffiotti, A.: Robot task planning using semantic maps. *Robotics and Autonomous Systems* 56(11), 955–966 (2008)
19. Stasse, O., Verrelst, B., Vanderborght, B., Yokoi, K.: Strategies for humanoid robots to dynamically walk over large obstacles. *IEEE Transactions on Robotics* 25(4), 960–967 (2009)
20. Sorribes, J., Prats, M., Morales, A.: Visual tracking of a jaw gripper based on articulated 3d models for grasping. In: *IEEE Int. Conf. on Robotics and Automation*, pp. 2302–2307 (May 2007)



## **Chapter II**

# **Unmanned Aerial and Underwater Vehicles**

Introduced by Kwang-Joon Yoon

A lot of research activities in aerial and underwater vehicles have been done not only for developing efficient platforms, but also for making them intelligent. The intelligent autonomous system in vehicles especially permits the vehicles to perform surveillance, reconnaissance missions and environmental monitoring without the human supervision. This technology generates various applications of the platforms in research and entertainment fields.

The development of unmanned autonomous vehicles has been of great interest, in which different kinds of autonomous vehicles have been studied and developed all over the world in recent decades. As a development of an intelligent system, UAVs (unmanned aerial vehicles) were born to support the military applications at the early stage of development. Recently, however, the demand has arisen for UAVs to be used in emergency situations and in industrial applications. UAVs are expected to support rescue for the situation unreachable by human in a dangerous disaster, such as earthquake, flood, volcano or nuclear disaster.

In addition to UAVs, biologically-inspired robotics has also received much attention recently. As an interdisciplinary field between biology and robotics, it is often referred as biomimetic robots or bio-robotics. The approach of biologically-inspired robotics development is often done by applying biological motivations or ideas to engineering problems or by using robots as physical models to answer biological questions.

In order to understand the achievements in the development of the unmanned aerial and underwater vehicles, this chapter presents the contents of the 12(13) papers presented in IAS-12, which are introduced in the category of autonomous aerial/underwater vehicles, bio-inspired flight, and special environment localization/navigation.

The papers related to bio-inspired flight cover the basic principles of specific natural species, strategy to mimic the features, fabrication and experimentation of flight systems, and their findings. Potential applications of flapping-wing systems are also discussed.

In the unstructured environment, such as under-water, air and disastrous area, the localization/navigation technologies are essential for rescue and construction. The measuring devices and algorithms for the localization/navigation are discussed in the papers related to the development of underwater robot. These papers introduce the

recent progress on underwater vehicle robots including its control method, simulation and flow analysis. These papers are also aimed at sharing the latest accomplishments and innovations in autonomous technologies of underwater vehicles and providing their potential future directions.

In this chapter, the contents of 12 papers presented in IAS-12 are introduced in the category of autonomous aerial/underwater vehicles, bio-inspired flight, and special environment localization/navigation.

# MAVwork: A Framework for Unified Interfacing between Micro Aerial Vehicles and Visual Controllers

Ignacio Mellado-Bataller<sup>1</sup>, Jesús Pestana<sup>1</sup>, Miguel A. Olivares-Mendez<sup>1</sup>, Pascual Campoy<sup>1</sup>, and Luis Mejias<sup>2</sup>

<sup>1</sup> Centre for Automation and Robotics (CAR), Universidad Politécnica de Madrid  
C/ Jose Gutierrez Abascal, 2. 28006 Madrid, Spain  
[www.vision4uav.com](http://www.vision4uav.com)

<sup>2</sup> Australian Research Centre for Aerospace Automation (ARCAA),  
Queensland University of Technology  
GPO Box 2434, Brisbane Queensland 4001  
[luis.mejias@qut.edu.au](mailto:luis.mejias@qut.edu.au)

**Abstract.** Debugging control software for Micro Aerial Vehicles (MAV) can be risky out of the simulator, especially with professional drones that might harm people around or result in a high bill after a crash. We have designed a framework that enables a software application to communicate with multiple MAVs from a single unified interface. In this way, visual controllers can be first tested on a low-cost harmless MAV and, after safety is guaranteed, they can be moved to the production MAV at no additional cost. The framework is based on a distributed architecture over a network. This allows multiple configurations, like drone swarms or parallel processing of drones' video streams. Live tests have been performed and the results show comparatively low additional communication delays, while adding new functionalities and flexibility. This implementation is open-source and can be downloaded from [github.com/uavster/mavwork](https://github.com/uavster/mavwork)

**Keywords:** MAV, UAV, communications, framework, software architecture.

## 1 Introduction

"Fail early, fail often" is a wise mantra. The earlier you find the mistakes in your new idea, concept or system design, the sooner you can fix them and get your path to success. This is especially applicable in the field of visual control, where dynamic systems are controlled using images from one or more cameras as feedback. Visual control algorithms that work fine on the simulator may fail catastrophically in the real world. In this paper, we propose a flexible unified software framework for visual control of Micro Aerial Vehicles (MAV).

In the last few years, personal MAVs have been hitting the consumer market. Currently, our framework supports the Parrot AR.Drone [10] and the AscTec Pelican [11], while support for other vehicles like [12], [13] and [14] is still in progress. The Parrot AR.Drone is sold as a toy at amateur-affordable prices. It has out-of-the-box

onboard cameras and Inertial Measurement Units (IMU). No user software can be run on board; any control algorithm lies off board, on a wirelessly linked computer. Whereas the overall quality is low compared to a professional MAV, it can be 20–40 times cheaper. In addition, it can be bought at several toy stores and taken straight to the lab without worrying about delivery delays. Furthermore, because of its low cost, taking risks is acceptable: if you crash and break one, you can just buy a new unit. For these reasons, it is worth to be taken into account as prototyping platform, especially when developing algorithms for MAV swarms with many units, where the total cost might be prohibitive with more professional MAVs.

AscTec Pelican is a professional solution and, consequently, much more expensive. It can carry additional payload and its frame is modular, so extra hardware may be mounted, like laser range finders or processing boards. It comes with an Atom board powered by a 1.6 GHz processor with 1 Gbyte RAM. Thus, the user is able to load and run programs on board. Unlike the AR.Drone, it has a GPS receiver, a barometric altimeter and a magnetometer (AR.Drone 2.0 also has the latter), but it does not have any cameras by default or a sonar altimeter, like AR.Drone does.

Both MAVs have Software Development Kits (SDK) that enable applications from third-party developers to communicate with the drones. In the case of the AR.Drone, the application is run on an external workstation and it sends commands and receives information from the sensors, the camera and the IMU through a WiFi link. For the Pelican, there is an onboard server that communicates with the Atom board through a serial link. However, both SDKs are too limited for our research requirements, as they only supports a single point-to-point link between a program and the drone, thus, a program can only communicate with a single drone. Besides that, we would like to work with networked communication schemes like those shown in Fig. 1.

The required new functionalities are provided by the proposed software framework, while increasing the isolation between the application and the hardware platform, and opening the possibility to easily port applications among MAVs from different manufacturers with either on-board or off-board computing.

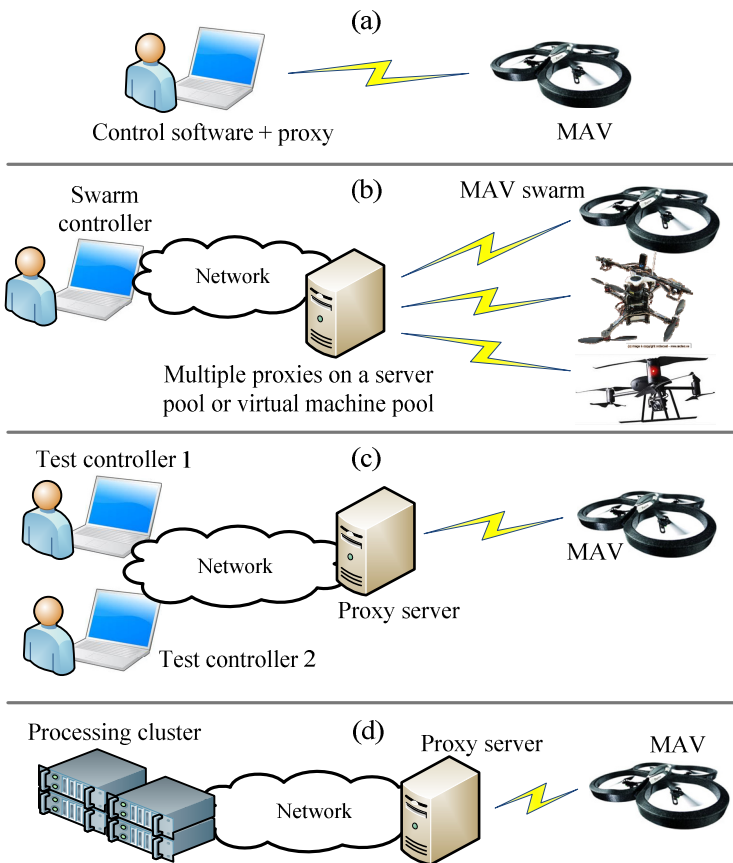
In section 2, other related works are explored. In section 3, we introduce some general guidelines of the framework architecture, while a specific implementation is discussed in section 4. In section 5, some test results of this implementation are presented and, in section 6, they are discussed. Section 7 concludes the paper.

## 2 Related Work

The AR.Drone SDK already offers an API for developing third-party applications [18]. Examples of research works with the AR.Drone are [7], [8] and [9]. However, by the time this paper is written, it does not support communications with multiple drones across a network. In the Pelican case, the autopilot board comes with a server that can send information and accept commands through a serial port, but it does not offer any networking either.

With regards to the communications between the application and the drone, reference [15] points to an existing open project by ETH Pixhawk. It offers a communication architecture for MAVs that is based on a library for message

transmission over a network [16], but it does not have native support for the Parrot AR.Drone or any other low-cost MAV. On the other hand, there is a driver for AR.Drone by Brown University [17] for the Robot Operating System (ROS) that was used in [9]. Nevertheless, it does not implement either access control to the drone or parameter configuration. Moreover, we would like our framework to remain lightweight, without burdening the new developer with the installation of heavy and complex packages like ROS. The framework implementation presented in this paper tries to fill the gap left by the other alternatives. So far, it has already enabled some research works like [1], [2], [3], [4] and [5].



**Fig. 1.** New communication schemes provided by our framework. In (a), a point-to-point scheme, also allowed by the AR.Drone SDK; in (b), an application controls an MAV swarm; in (c), multiple researchers may share the same MAV resources -one at a time-; in (d), video is broadcasted over a cluster for parallel processing.

### 3 Framework Architecture

In this section, we define a general model for the implementation of our framework architecture. These are guidelines and requirements that are extensible to any MAV, any packet network and any application programming language. In the next section, the model will be applied to the supported MAVs, to specific network technologies and a C++ API.

To give network capabilities to the drone, a proxy-based architecture has been defined. The architecture is depicted in Fig. 2. The proxy is responsible for connecting a single drone with the network. With one proxy per drone, all drones can share the network as communication mean. At the application side, we define an Application Programming Interface (API). Thanks to this API, the application is able to communicate with the proxies of the different drones that it aims to control.

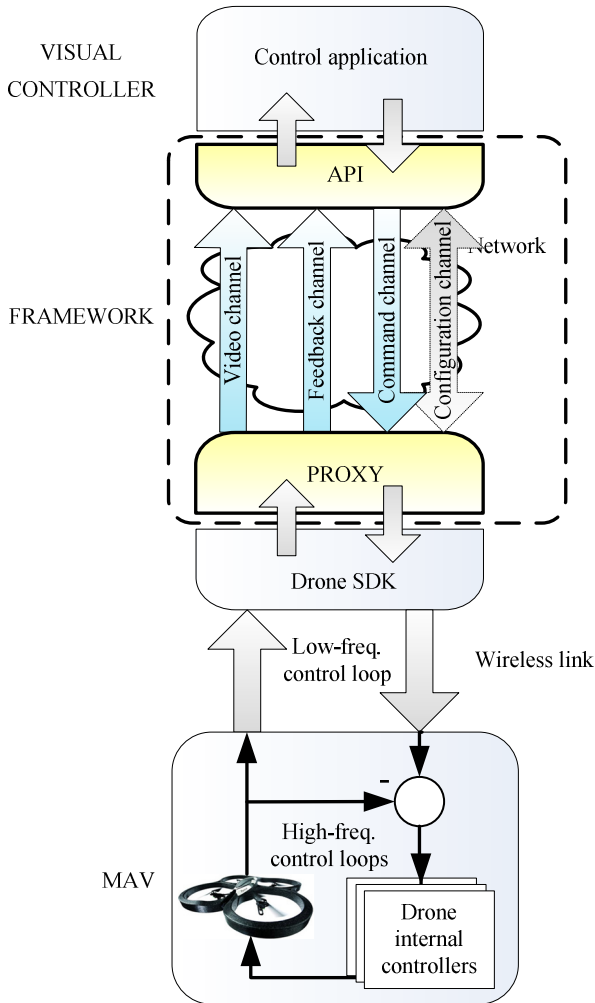
It is worth to notice that the framework components do not have fixed running locations. If the MAV lets the user run software on board, the proxy can stay there. Then, the control application can reside either onboard, communicating locally, or off-board, through a wireless link.

Otherwise, if the MAV does not let the user run software applications on board, which is the case for the AR.Drone, the proxy is run off board, and the control application can be executed either on the same platform or on any other that is connected through a network, as seen in Fig. 1.

Regarding portability, while the proxy depends on the drone manufacturer, the Application Programming Interface (API) library is platform-independent. In other words, the proxy isolates the application from the drone specifics. In this way, there is no need to update all control applications every time the manufacturer releases a new SDK version. Most times, updating the proxy will be enough. Another advantage of this isolation, is the possibility of porting the API to programming environments or languages not supported yet by the manufacturer's SDK. For instance, a Matlab API could be programmed, despite not existing any specific software by the manufacturer.

#### 3.1 Communications

The communication link between the proxy and the MAV depends on the manufacturer specification and it may vary between different models. It is the manufacturer who defines the communication protocol of the drone and it will not be discussed in this paper. Our framework is responsible of the link between the proxy and the application. This link is formed by four independent communication channels, named: command, feedback, video and configuration. These channels are logical, not necessarily physical, as they are established over the network. They just represent an information flow between both network nodes. To implement the channels, no specific communication protocols are defined as mandatory; there are only recommendations.



**Fig. 2.** System architecture. The framework interconnects the visual controller with each MAV through a network. The total system follows a cascade control structure. Usually, there are high-frequency controllers on board that maintain the MAV's attitude and altitude as desired, while the visual controller closes an outer loop with lower frequency. The network link for the control loop is formed by three low-delay independent channels. Configuration channel is not intended to close a control loop, but to read and change configuration parameters occasionally.

The network between the proxy and the application may fail. A cable might break, a router might stop or a WiFi link might lose the signal. Both ends of the link must be robust to these situations and implement self-recovery mechanisms, which must be transparent to the application. The application will be notified of a failure situation but will not have to perform any actions to fix it. In case of stateless protocols

—those not requiring to establish a connection— there is no extra effort to be done, as packets will continue to be transmitted after the network is recovered. Nevertheless, the application must be notified if packets do not arrive at expected times. Oppositely, connected protocols must automatically try to reconnect until the network is restored, besides informing the application of the link state.

### 3.1.1 Command Channel

The command channel transports all the control actions from the application to the proxy: a signature, a sequence number, the desired access level and drone-specific commands defined by the implementation (desired attitude, etc.). The signature identifies the packet as a command channel packet and may be used as a start token.

Through this channel, data packets are transmitted periodically. Low delays are valuable in this channel because control loops depend on it and delays generally harm loop stability [6]. So, low delays are favored by dropping in the receiver any malformed packets or assuming as lost packets those that did not arrive on time. As a rule of thumb, it is better to lose a packet and receive as soon as possible the next one with the most up-to-date information than to retry the transmission of a lost packet by delaying future ones with more current data. Because of that, datagram protocols, like UDP over an IP network, are suitable for implementing this mechanism because they do not have automatic retransmission of faulty packets. At the proxy side, if a packet is lost or it arrives after a previously sent one, it is discarded. Instead of asking for a retransmission or reordering packets after a sequence error, the proxy expects that a new packet with up-to-date command information will eventually arrive. The purpose of the sequence number is to determine if a packet has arrived out of sequence.

An MAV can only be commanded by one control application at a time. Therefore, no concurrent access is allowed on this channel. When the channel is in a free state, any application willing to control the MAV can lock it by sending an initialization packet for write access. After that, no other control packets from other applications are processed until the original application unlocks the channel or stays inactive for a time longer than a pre-configured threshold. It is also possible for the applications to define their role as "only listening", so the framework never gives them control. This is achieved with the access level field.

### 3.1.2 Feedback Channel

In the feedback channel, navigation information flows from the proxy to the application. The content of each feedback packet is: signature, sequence number, granted access level and drone-specific information defined by the implementation (proxy-to-drone link health, battery level, measured attitude, etc.). The signature identifies the packet as pertaining to this channel and may be used as start token. Like in the command channel, packet dropping at the receiver —based on the sequence number— is encouraged in order to minimize delays in control loops.

Through the feedback channel, the proxy can feed data to multiple applications simultaneously. It will do it with those that are only listening as well as with those that are willing to take control of the MAV. With the granted access level field, every application knows if it is allowed to control the MAV.



### 3.1.3 Video Channel

In a video channel, video from a drone camera is transmitted to the control application. Like the feedback channel, multiple applications can request video channels from a proxy. While a feedback channel sample will usually fit in a network packet, a video channel sample, i.e. a video frame, will need to be encoded, packetized and transmitted with some transport protocol. Like for the other channel types, the lower the transmission delay is, the higher the stability margin of a visual control loop will be. Hence, implementations with compression-ready encodings, low-delay protocols and frame-dropping mechanisms would be preferred.

The video channel transports periodic fragments with frame data that include a header with a signature (it may be used as a fragment start token), information about the video encoding and a timestamp, so the application knows how to decode the video stream and when each frame was captured. The timestamp must be as close as possible to the real capture time of a frame. If neither the camera nor the MAV provide this information, the proxy will give an estimation. When possible, timestamps of different channels must use the same clock reference. Although this reference is unknown by the application, sample times of different channels can be compared and ordered if needed.

### 3.1.4 Configuration Channel

The configuration channel is used to read and write configuration parameters of the MAV from the application. It is intended for parameters that are not time-critical, such as allowed attitude ranges or video capture features, which are mainly changed at startup. In order not to disturb any other channels requiring a higher bandwidth and a lower delay, any fast changing parameters must be transferred through the command and feedback channels.

When the application writes a parameter through the configuration channel, it must have a confirmation that it has actually been changed in the MAV, as it might be safety-critical. Likewise, when reading a parameter, the application must know that it was actually read. Therefore, a connection-oriented transport protocol is required for this channel. For example, TCP on an IP network would suit these requirements.

## 3.2 Application Programming Interface

The API library enables the application to access the communication architecture programmatically. The control application processes the feedback information from the MAV and generates the commands to be sent in response, closing the loop. The API defines methods that are directly called to change these commands.

The application can gather the feedback information –video and navigation– in two ways. The first one consists in explicitly polling the data when needed. However, because of the asynchronous nature of the feedback channel, the data is not requested on demand to the drone, but periodically received. And, consequently, the request method returns the last sample that was received from the proxy. The second method for feedback retrieval is event-driven. The application registers a listener through the API and the listener gets a notification whenever the data is received from the proxy, so it can be processed immediately. Navigation data and video frame notifications are received independently, as they are transmitted through unrelated channels, due to their different bandwidth requirements.

## 4 Framework Implementation

The framework model has been implemented for IP networks. So far, the implementation supports the AR.Drone and the Pelican, focusing on indoor environments.

For the AR.Drone, a specific proxy was built over the manufacturer's SDK examples [18]. The proxy is a separate executable that runs off-board the MAV because the onboard computer is closed to third-party code. The manufacturer's point-to-point communication with the drone is established via WiFi.

For the Pelican, a generic proxy has been implemented in a modular fashion for Linux systems, using the C++ language. It runs on the Pelican's Atom board, which has a WiFi card. In fact, this generic proxy can be customized to any drone running Linux, by changing the modules that communicate with the autopilot and cameras.

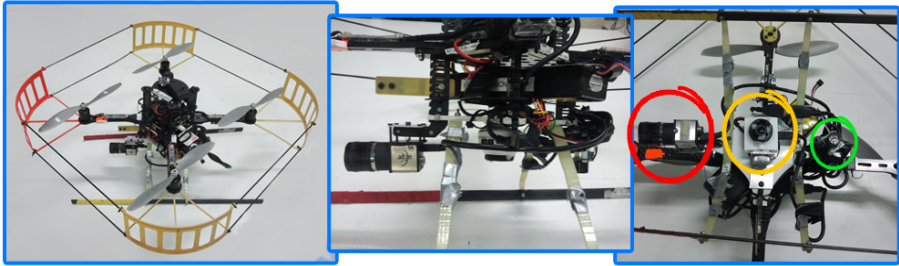
### 4.1 Channels

Besides the mandatory fields defined by the framework model for communications and networking, drone-specific information is transmitted through the command and video channels. These fields conform a protocol in the application layer according to the OSI model. We call this protocol MAV1 and it also defines units and reference frames. The AR.Drone supports MAV1 natively, but the Pelican needs some additional hardware (Fig. 3) and software to behave equivalently. Specifically, we had to add a sonar for indoor altitude measuring and a down-facing camera for velocity estimation. Assuming the floor is flat, an algorithm was designed to estimate the ground speed from the optical flow in the image and the sensed attitude and altitude. Moreover, automatic take-off, hovering and landing modes were implemented in the Pelican proxy to comply with MAV1.

The command channel is implemented using a UDP socket. Besides the mandatory information defined by the framework model, it carries the following payload data for the MAV: timestamp, required flying mode, attitude desired values and desired altitude rate.

The feedback channel uses a UDP socket, too. In addition to the fields required by the framework, it transports the following information: timestamp, proxy-to-drone link health, current flying mode, battery level, measured attitude, measured altitude and measured velocity.

As UDP is not a reliable protocol, command and feedback channels are provided with a periodic update mechanism. It means that, at the application side, as soon as commands are changed by the application, the API library transmits them to the drone through the proxy. When the application is not generating new commands, the API library keeps transmitting the last commands periodically to ensure that they eventually arrive to the other end. The proxy has the same mechanism: new sensor readings are sent immediately, but if they are not available at a predefined minimum frequency, the last readings are periodically sent through the feedback channel to ensure that they arrive to the other end. In this way, there is constant activity in the channels and both ends know that they are linked.



**Fig. 3.** Additional hardware setup for the AscTec Pelican. Thanks to this hardware and the onboard proxy algorithms, the Pelican can comply with the MAV1 protocol and may be controlled as an AR.Drone. The circled elements are: front camera (red), down-facing camera for velocity estimation (yellow), altitude sonar (green).

The video channel is implemented with a TCP socket. According to the model definition, this is not the most adequate protocol because it is not designed for real time, but for reliability. Nonetheless, the transparent streaming capabilities of the protocol make the video channel implementation straight-forward and delays may be diminished with some tuning. Anyway, we expect to use a more adequate protocol in future releases of the implementation.

Each frame is transmitted over the video channel with its own encoding. Currently, the supported encodings are JPEG, raw 3-channel images with 8 bits per plane (for RGB cameras) and single-channel images with 16 bits per plane (for rangefinders). To pass the received video frame to the application, the API library has a triple buffering mechanism: the reception buffer, the frame-ready buffer and the processing buffer. The first one is continuously retrieving the frames from the network, preventing the TCP buffers from overflowing, which would time out the transmission at the proxy side and would be interpreted as a connection failure. Right after a frame is received, it is copied to the frame-ready buffer, to keep it accessible by other program modules, while the reception buffer is free to receive the next frame from the network. However, the frame-ready buffer is overridden as soon as a new frame is received, therefore any operation on this buffer should last less than a frame period. For longer processing times, the processing buffer is provided. When the frame-ready buffer gets new contents, all the video channel listeners are notified. One of them is a video processor module that copies the frame-ready buffer contents to its own processing buffer only after the last processing operation has finished. Meanwhile, the frames are dropped for that video processor. Multiple video processors can be freely initiated by the application, allowing concurrent frame processing with independent frame dropping for each processor.

The configuration channel is implemented with a TCP socket, as low delays are not mandatory but reliability is. Each parameter operation is performed in a transaction consisting in a request and a response. Each request contains a signature, the request type, the parameter identifier and the parameter desired value. The desired value will only be interpreted by the proxy if it is a write request. The response is formed by a

signature, a value indicating whether the last request was successful and the parameter value. The returned parameter value is only meaningful if the last request was for reading.

## 4.2 Robustness

At both communication ends, there is code responsible for keeping communication channels synchronized. If faulty behavior occurs, the corresponding channel is restarted, so both ends are automatically synchronized back. The channel behavior can be understood as faulty when a malformed packet is received or when packets are not received as frequently as expected. Every time this happens, the application is notified so it can react accordingly. For example, it could display an alarm on a user interface. However, the channel recovery mechanism is completely transparent and all efforts for the channel restoration are performed by the framework.

At the application side, all the API errors are handled with C++ exceptions. This mechanism favors that errors show up during the development phase so they can be fixed early. In this API implementation, every thread has a last line of defense that catches all non-caught exceptions, writes the exception in a log file for debugging and prevents the thread from being terminated, so it can try to recover the normal state.

## 4.3 Extra Features

The API library is able to interface with a Vicon positioning system. With this system, position and attitude information of MAVs can be gathered inside a delimited space. This information can be very useful, for instance, to close control loops or as ground truth for visual pose estimation algorithms.

On the other hand, the API library provides data logging functionalities. The data logger can gather events generated by all the channels, the Vicon interface or other objects defined by the developer in the application. Hence, commands, navigation feedback, video feedback, Vicon data and developer-defined information can be stored in a disk for later analysis. The data logger runs asynchronously, so the delays of the disk write operations do not bother other ongoing threads.

Finally, the API defines classes that help developing a controller by only overriding three methods. Two of them are automatically called whenever navigation or visual feedback is received from the MAV. The third method is called any time the framework requires the application to reset the configuration parameters; for instance, after the MAV is rebooted. A controller may be implemented inside the first two methods. The received information is used as input to the controller and the controller's output is sent to the MAV directly calling the appropriate API methods.

To help writing the controller code, the API also exposes matrix data types that perform common algebraic operations. In addition, the images from the cameras are passed back with the encoding used by OpenCV, for easy integration with that library.

## 5 Experimental Results

The total communication delay between the application and the drone will be the sum of the delays introduced by the API library, the network, the proxy and the proxy-to-drone link. The drone manufacturer is accountable for the latter. The second one is given mainly by the physical network infrastructure. The API and proxy delays are responsibility of the framework implementation and must be measured.

In order to measure the framework contribution to the delay, the proxy is run in the same host where the application resides, so the API-proxy link is established through local sockets. Timestamps are added to channel packets at the sender and the time lapse is calculated at the receiver. As both processes run on the same computer, they share the same clock reference and time calculations can be performed without additional synchronization.

Regarding the proxy-to-application delay, the timestamps are obtained right after receiving the data from the drone, so all proxy processing time is also taken into account. The arrival time is acquired right after releasing the data to the application. For the application-to-proxy delay, the timestamps are taken right after issuing the commands to the API and the arrival time is calculated at the proxy, before sending the commands to the drone through the point-to-point link. The test was run with the AR.Drone proxy, on an Acer Aspire 5750G with a Intel Core i7-2630QM 2GHz processor and 8 Gbytes of DDR3 RAM. The Operating System was Linux Ubuntu 11.04. During the test, the data logging was disabled. The packet frequency for the command and feedback channels was set to 32 Hz. The video frame rate was 15 frames per second in average (this is determined by the AR.Drone) and the video channel frames were encoded as raw RGB with eight bits per plane. The test application consists on a simple visual teleoperation interface with a waypoint-based path controller. The test duration is 5 minutes.

Figs. 4 and 5 show the distribution of delays introduced by the framework in the command and video channels (the feedback channel distribution is similar to the command channel one). Table 1 gives some numerical details about the delay distributions. Fig. 6 shows the time evolution of the delays. In all figures, delays are expressed as percentages of the channel period. The channel periods are 31.25 ms for command and feedback, and 66.7 ms for video.

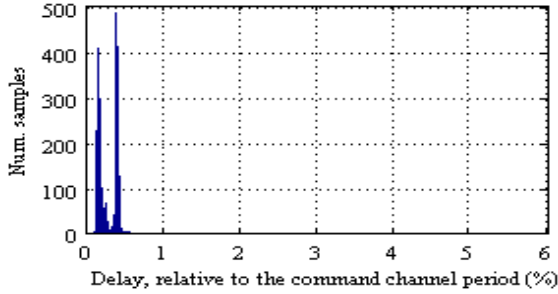
**Table 1.** Characterization of channel delays

Channel	Delays (ms)			Num. samples	
	Mean <sup>a</sup>	Min.	Max.	Total <sup>b</sup>	Delay < 1% of channel period <sup>c</sup>
Command	0.091 (0.29%)	0.013	1.739	9,771	99.93%
Feedback	0.109 (0.35%)	0.025	1.670	9,828	99.73%
Video	1.038 (1.56%)	0.310	1.936	5,011	0.22%

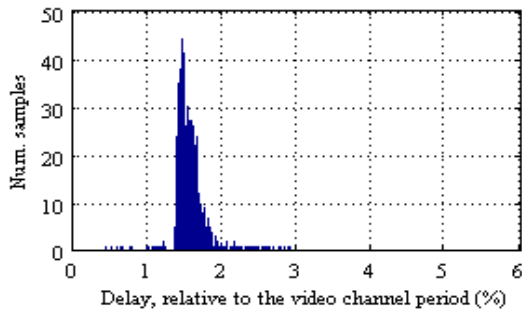
a. Absolute delay in milliseconds and delay relative to channel period.

b. Total number of delay samples.

c. Samples lower than 1% of channel period (31.2 ms command and feedback; 66.7 ms video).



**Fig. 4.** Distribution of the delays introduced in the command channel by the framework. The relative delays are percentages of the command channel period, i.e. 31.25 ms. The highest sample is 5.56%.

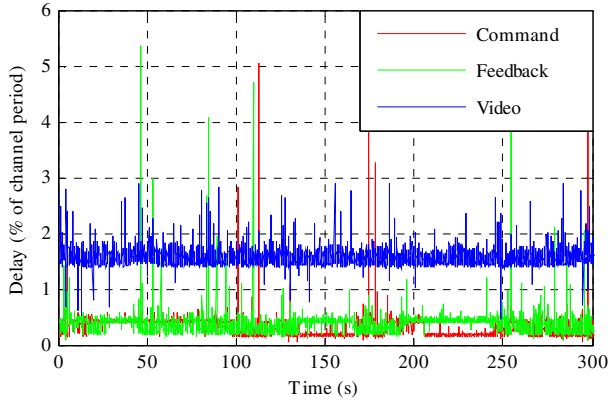


**Fig. 5.** Distribution of the delays introduced in the video channel by the framework. The relative delays are percentages of the average video channel period, i.e. 66.67 ms. The highest sample is 2.9%, but the horizontal scale is set as in Fig. 4 for easy comparison.

## 6 Discussion

As seen in table 1, the average delays introduced by the framework are considerably low, compared to the frequencies of the channels. For a visual controller, the impact of the framework in the reaction time would be the result of adding the visual and command channel delays, i.e. the time it takes to see an event plus the time to react accordingly. In average, it is a contribution of 1.129 ms to the total loop delay. Assuming a visual control loop at 15 frames per seconds, this represents a 1.7% of the loop period.

In Fig. 6, there are spurious samples that might be caused by the fact that the implementation is not running on a real-time Operating System (OS). Instead, this OS has a preemptive scheduler that can interrupt a task anytime to yield some time for other tasks. Despite it might not cause problems during usual prototyping, it must be taken into account for high-frequency delay-sensitive applications.



**Fig. 6.** Evolution of relative channel delays introduced by the framework. Most of the time, command and feedback delays are below 1% of the channel period. Highest peaks reach 5%, but are very unusual. The video channel delay is always under 3%.

## 7 Conclusion

We have introduced a framework to interface visual controllers with Micro Aerial Vehicles (MAV) in a unified way. Thanks to the framework, control applications can work with MAVs from multiple manufacturers without changing the code. This allows faster and safer prototyping through pre-testing with low-cost drones before moving to the professional MAVs. In addition, the framework transforms any MAV into a network node, opening the door to new prototyping configurations, like drone swarms, distributed vision processing or MAV sharing by multiple researchers.

First, a framework model with general guidelines has been presented, without regarding specific technology details, in order to leave it open to other implementations. The framework defines a distributed architecture that allows for multiple experimental setups like drone swarms, MAV sharing or distributed processing. Moreover, applications can communicate with the unified API in easy and efficient ways. Currently, there is a framework implementation based on the proposed model. It supports the AscTec Pelican, as professional drone, and the Parrot AR.Drone, as prototyping platform. More MAVs will be supported in the near future.

In order to show the framework applicability to visual control loops, the introduced additional delays have been analyzed. In the experimental results, these delays are significantly low, compared to the loop periods. However, the timings are not deterministic because the implementation is not running on a real-time Operating System. Thus, its applicability is disregarded to controllers where safety is extremely critical. Anyhow, it does not affect the majority of applications. On the contrary, the framework has proven to be a useful tool for rapid testing and experimentation. This implementation is an open-source project available at [github.com/uavster/mavwork](https://github.com/uavster/mavwork).

**Acknowledgements.** The authors thank the Australian Research Centre for Aerospace Automation - Queensland University of Technology (ARCAA-QUT), where a major part of the development and testing for this work was carried out. The authors would also like to thank Caja Madrid for the mobility research grant of one of the authors. This work has been sponsored by the Spanish Science and Technology Ministry under the grant CICYT DPI2010-20751-C02-01 and by the IRSES Program Marie Curie FP7-PIRSES-GA-2009-230797 - ICPUAS (International Cooperation Program for Unmanned Aerial Systems Research and Development).

## References

1. Pestana, J., Mellado-Bataller, I., Fu, C., Sanchez-Lopez, J.L., Mondragon, I.F., Campoy, P.: A visual guided quadrotor for IMAV 2012 indoor autonomy competition and visual control of a quadrotor for the IMAV 2012 indoor dynamics competition. In: Int. Micro Air Vehicle Conference and Flight Competition (IMAV), Braunschweig, Germany (July 2012)
2. Olivares-Mendez, M.A., Mejias, L., Campoy, P., Mellado-Bataller, I.: See-and-avoid quadcopter using fuzzy control optimized by cross-entropy. In: IEEE World Congress on Computational Intelligence, Brisbane, Australia (June 2012)
3. Pestana, J., Sanchez-Lopez, J.L., Mellado-Bataller, I., Fu, C., Campoy, P.: AR Drone identification and navigation control at CVG-UPM. In: XXXIII Jornadas de Automática (September 2012)
4. Olivares-Mendez, M.A., Mejias, L., Campoy, P., Mellado-Bataller, I.: Quadcopter see and avoid using a fuzzy controller. In: Proc. 10th Int. FLINS Conf., Istanbul, Turkey (in press, August 2012)
5. Mellado-Bataller, I.: Vision-based pose estimation using 3D markers (March 2012), <http://www.vision4uav.com/?q=node/274>
6. Branicky, M.S., Phillips, S.M., Zhang, W.: Stability of networked control systems: explicit analysis of delay. In: Proc. American Control Conf, pp. 2352–2357 (June 2000)
7. Visser, A., Dijkshoorn, N., van der Veen, M., Jurriaans, R.: Closing the gap between simulation and reality in the sensor and motion models of an autonomous AR.Drone. In: Proc. International Micro Air Vehicle Conference and Flight Competition, IMAV 2011, pp. 40–47 (September 2011)
8. Bills, C., Chen, J., Saxena, A.: Autonomous MAV flight in indoor environments using single image perspective cues. In: Int. Conf. Robotics and Automation (ICRA), Shanghai, China, pp. 5776–5783 ( May 2011)
9. Koval, M.C., Mansley, C.R., Littman, M.L.: Autonomous quadrotor control with reinforcement learning, <http://mkoval.org/projects/quadrotor/files/quadrotor-rl.pdf>
10. Parrot. AR.Drone, <http://ardrone.parrot.com>
11. AscTec. Pelican, <http://www.asctec.de/asctec-pelican-3>
12. UAS Technologies LinkQuad, [http://www.uastech.com/LinkQuad\\_system\\_uastechnologies.pdf](http://www.uastech.com/LinkQuad_system_uastechnologies.pdf)
13. MikroKopter. Products, <http://www.mikrokoetter.de>
14. Diy, D.: ArduCopter, <http://code.google.com/p/arducopter/>



15. Pixhawk. MAVCONN Aerial Middleware,  
<https://pixhawk.ethz.ch/software/start>
16. QGroundControl. MAVLink protocol,  
<http://www.qgroundcontrol.org/mavlink/start>
17. Brown University. ROS driver for the Parrot AR.Drone,  
[http://code.google.com/p/brown-ros-pkg/wiki/ardrone\\_brown](http://code.google.com/p/brown-ros-pkg/wiki/ardrone_brown)
18. Parrot, ARDrone API (December 14, 2009), <https://projects.ardrone.org>

# Smart Filter Design for the Localization of Robotic Fish Using MEMS Accelerometer

Tae Suk Yoo<sup>1</sup>, Sang Cheol Lee<sup>2</sup>, Sung Kyung Hong<sup>2,\*</sup>, and Young Sun Ryuh<sup>3</sup>

<sup>1</sup> LIG Nex1 Co, Korea

<sup>2</sup> Department of Aerospace Engineering,  
Sejong University, Korea

<sup>3</sup> Korea Institute of Industrial Technology, Korea  
taesuk.yoo@lignex1.com, tend@sju.ac.kr,  
skhong@sejong.ac.kr, ysryuh@kitech.re.kr

**Abstract.** This paper presents the design of smart filter for the 2D localization of robotic fish using low-cost MEMS (Micro-Electro Mechanical System) accelerometer. The main purpose of the paper is to minimize the drift error that is inevitable in the double integration process in accelerometer-only navigation system. The proposed approach relies on two parts: 1) an effective calibration method to remove the major part of the deterministic sensor errors and, 2) a novel smart filtering scheme based on fuzzy-logic in order to accurately estimate a 2D position with an accelerometer triad. In addition, we compare the results of the fuzzy logic based on 2D position estimation system with simulation result from a conventional Kalman Filter.

**Keywords:** robotic fish, 2D position estimation, MEMS inertial sensor, calibration, fuzzy filter.

## 1 Introduction

Performing reliable localization and navigation within highly unstructured underwater environments is a difficult task. A typical navigation sensor outfit for an underwater vehicle may consist of standard components such as compass, pressure sensor, and some class of INS (Inertial Navigation System) [1]. High precision (inertial grade) IMUs (Inertial Measurement Units) offer excellent strapdown navigation capabilities, but their size, power consumption, and cost have precluded their widespread use in small robotic fish. One possibility is to use MEMS based inertial sensors. In recent years, MEMSs have capability to fabricate cheap, lightweight, and low-power chip-based inertial sensors which have been adopted into various applications such as automobiles, consumer electronics, and medical electronics [2-3]. Although the MEMS inertial sensors have such advantages, they are still suffering from large errors including bias, scale factor error, non-orthogonality, random noise (drift), temperature effect, and so on. It is customary to distinguish between deterministic errors and

---

\* Corresponding author.

stochastic errors of MEMS inertial sensors. Deterministic errors include bias, scale factor error and non-orthogonality. These error sources can be removed by calibration procedure. On the other hand, stochastic error sources are not easy to handle due to its random nature. For the navigation system employing MEMS inertial sensors, several estimation techniques are chosen currently to eliminate noise problems. For example, the Kalman Filters use statistical error models to predict the behavior of sensor components. This approach, however, is not ideal for MEMS accelerometer based robotic fish system because the statistical models cannot represent well single “catastrophic” events [4-5]. Fuzzy systems have been rapidly emerging as a powerful resource in the field of instrumentation and measurement. Indeed, a variety of applications have been recently proposed encompassing control systems, multi-sensor data fusion, data analysis and digital signal processing. The main purpose of the paper is to minimize the drift error that is inevitable in the double integration process in accelerometer-only navigation system. We compare the results of our fuzzy logic system with simulation result obtained from a conventional Kalman Filter.

## 2 Experimental Setup for the 2D Position Estimate of the Robotic Fish

During the entire process of the paper, a set of data obtained in a water tank. As shown in Fig. 1, the test runs are performed in the water tank (700cm×400cm×70cm) with a small robotic fish prototype by moving along a rectangular trajectory (200cm×200cm). A MEMS triad IMU module, 3DM-GX3-35 of Microstrain Inc., was selected for this research [6]. The positioning estimation filters are implemented in a real-time DSP (Digital Signal Processor) (TMS320F28335) within the size of 6cm×4cm×3cm.



**Fig. 1.** Water tank and robotic fish and DSP module with MEMS IMU (3DM-GX3-35)

With acceleration measurement attached to the body of the robotic fish, the 2D position estimate process consists of three parts as shown in Fig. 2: 1) coordinate transformation of the acceleration in the vehicle’s body frame to the navigation frame 2) butterworth bandpass filtering for the alleviation of both drift and noise, and 3) double integration process for the final position estimates.

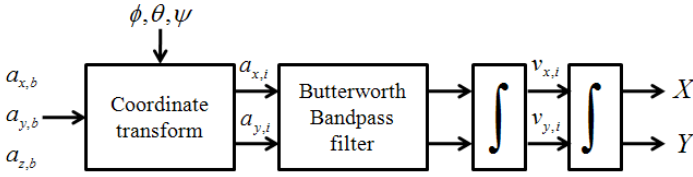


Fig. 2. 2D position estimation process

Clearly, an accelerometer will introduce an error proportional to  $t^2$  in the position due to the drift by double integration. To alleviate drift in position estimates without employing other aided external sensors, one possible solution is to employ a linear bandpass filtering by choosing a cutoff frequency somewhere between the frequencies of low-frequency drift signal and that of the periodic motion which has relatively high frequency [7]. The butterworth filter is a type of signal processing filter designed to have as flat (no ripples) a frequency response as possible in the passband, so it is termed a maximally flat magnitude filter. In order to observe the nominal maneuvering frequency of robotic fish, the discrete Fourier transforms of the signal is found by taking the FFT (Fast Fourier Transform) algorithm, as shown in Fig. 3. Considering the nominal maneuvering frequency of robotic fish ( $\approx 2\text{Hz}$ ), the 0.1 and 3 Hz cutoff frequencies are chosen for the butterworth filter. Bode plot is also shown Fig. 3.

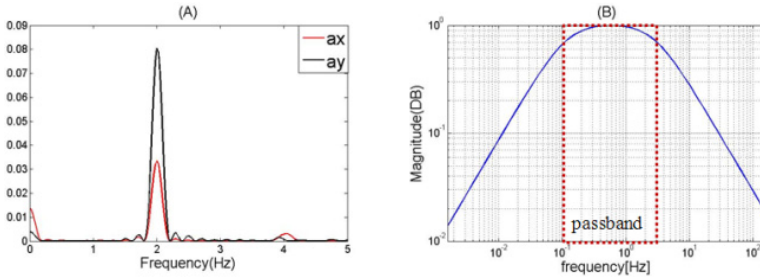


Fig. 3. Analyze (A) and bode plot of butterworth bandpass filter (B)

### 3 Calibration Method

#### 3.1 Sensor Error Model

A measurement provided by an accelerometer may be expressed in terms of an applied acceleration and the sensor error coefficients. The following equation is typically used to describe the measured acceleration:

$$a_{m,i} = a_i + b_i + S_i a_i + m_{i,j} a_j + \epsilon_i \tag{1}$$

Where  $a_{m,i}$  represents the measured acceleration for  $i$  sensitivity axis,  $a_i$  is the true acceleration about the sensitivity axis,  $b_i$  is the accelerometer instrument bias,  $S_i$  is the scale factor error matrix,  $m_{i,j}$  is the non-orthogonality matrix, and  $\varepsilon_i$  is the random noises. From Equation (1), it is evident that bias, scale factor, and non-orthogonality errors are the dominant deterministic elements. If these errors are minimized, their residual effects during navigation will minimize the position drift of the navigation system.

### 3.2 Least Squares Algorithm

Calibration is the procedure for comparing instrument output with known reference (true) information about the quantity to measure. Standard calibration method can be used to determine the bias and scale factors of the sensors, however, cannot estimate the non-orthogonality. An improved six-position test method which takes into account all three types of errors (bias, scale factor, and non-orthogonality) at the same time is provided.

$$\begin{aligned}
 & \mathbf{G} = \mathbf{MA} + \mathbf{B} \\
 & \underbrace{\begin{bmatrix} a_{m,x} \\ a_{m,y} \\ a_{m,z} \end{bmatrix}}_{\mathbf{G}} = \underbrace{\begin{bmatrix} (1+S_x) & m_{xy} & m_{xz} \\ m_{yx} & (1+S_y) & m_{yz} \\ m_{zx} & m_{zy} & (1+S_z) \end{bmatrix}}_{\mathbf{M}} \underbrace{\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}}_{\mathbf{A}} + \underbrace{\begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}}_{\mathbf{B}} \tag{2}
 \end{aligned}$$

Equation (2) can be rearranged as:

$$\begin{aligned}
 & \mathbf{G} = \tilde{\mathbf{M}}\tilde{\mathbf{A}} \\
 & \underbrace{\begin{bmatrix} a_{m,x} \\ a_{m,y} \\ a_{m,z} \end{bmatrix}}_{\mathbf{G}} = \underbrace{\begin{bmatrix} (1+S_x) & m_{xy} & m_{xz} \\ m_{yx} & (1+S_y) & m_{yz} \\ m_{zx} & m_{zy} & (1+S_z) \end{bmatrix}}_{\tilde{\mathbf{M}}} \underbrace{\begin{bmatrix} b_x & a_x \\ b_y & a_y \\ b_z & a_z \\ & 1 \end{bmatrix}}_{\tilde{\mathbf{A}}} \tag{3}
 \end{aligned}$$

By aligning the accelerometer using the six-position method, the ideal acceleration would be measured as:

**Table 1.** Ideal acceleration for the six-position method

	case1	case2	case3	case4	case5	case6
$a_x[g]$	1	-1	0	0	0	0
$a_y[g]$	0	0	1	-1	0	0
$a_z[g]$	0	0	0	0	1	-1

$\tilde{A}$  is the matrix for the least squares adjustment.  $G$  is measured accelerometer matrix. The desire is to extract the components of the matrix  $\tilde{M}$  in Equation (3). This can be done using the well-known solution of least squares:

$$\tilde{M} = G \cdot \tilde{A}^T \cdot (\tilde{A}\tilde{A}^T)^{-1} \tag{4}$$

### 4 Fuzzy Logic Signal-Processing Algorithm

Stochastic error source is not easy to handle due to its random nature. In addition, due to the high impact of perturbations within highly unstructured underwater environments, a simple linear filter is not sufficient for the elimination of noises. The random noise provides a large error which increases with time due to double integration process. Therefore, a smart signal-processing algorithm that is capable of the detection of corrupted data is required.

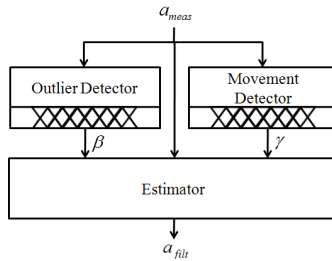


Fig. 4. Fuzzy filtering strategy

A fuzzy logic based on nonlinear filter is used to approximate the navigation trajectory as precisely as possible. The method to design fuzzy filtering for inertial sensors is explained in [9-9]. In the paper, the proposed fuzzy filtering strategy is composed of an outlier detector, a movement detector, and estimator processing. The structure of integrated fuzzy filtering is shown in Fig. 4.

#### 4.1 Outlier Detector

The first step towards obtaining a valid data is the detection of outlying observations. Detected outliers are candidates for aberrant data that may otherwise adversely lead to biased parameter estimation and incorrect results. In the paper, the outlier detector computes the last three data and the structure of outlier detector is shown in Fig. 5.

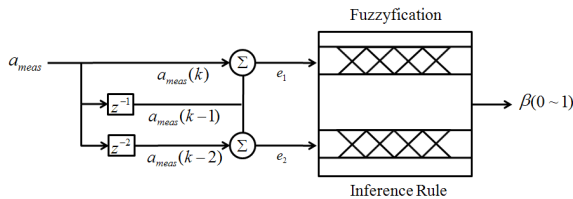


Fig. 5. Structure of outlier detector

In the proposed scheme, the outlier is identified based on the magnitudes  $e_1$  and  $e_2$ , which are defined as:

$$e_1 = a_{meas}(k) - a_{meas}(k-1) \tag{5}$$

$$e_2 = a_{meas}(k-1) - a_{meas}(k-2) \tag{6}$$

Fuzzy rules have the following form:

$$\text{If } e_1 \text{ is } A_{1,i} \text{ and } e_2 \text{ is } A_{2,i}, \text{ then } \beta = \beta_i \tag{7}$$

Where  $A$ 's is MFs (Membership Functions),  $\beta$  (the possibility of the outlier) is the output ( $\beta=0$  means valid signal and  $\beta=1$  means outlier), and the subscript  $i$  denotes the rule number. The MFs of the fuzzy sets for  $e_1$  and  $e_2$  are shown in Fig. 6. Note that  $\beta_i$  may be considered as a singleton MF as shown in Fig. 6. The truth value of the rule ( $\mu_i$ ) in Equation (7) is obtained by the product of the MF values

$$\mu_i = \mu_{A_{1,i}}[e_1] \cdot \mu_{A_{2,i}}[e_2] \tag{8}$$

Where  $\mu_{A_{1,i}}$  and  $\mu_{A_{2,i}}$  are the MF values of the fuzzy set given a value of  $e_1$  and  $e_2$ , respectively. Based on  $\mu_i$ , the defuzzification value of  $\beta$  yields the following:

$$\beta = \sum \mu_i \beta_i \tag{9}$$

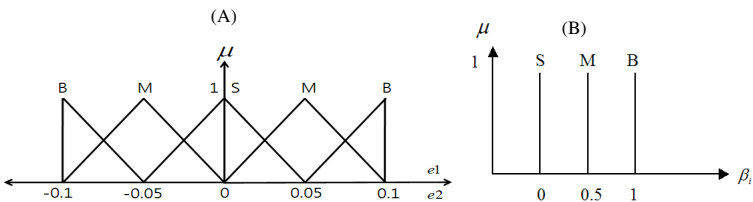


Fig. 6. Membership functions (A) and Singleton membership function (B)

### 4.2 Movement Detector

An easy way to avoid a position drift is to detect when the system is immobile. The concept of movement detector is shown in Fig 7.

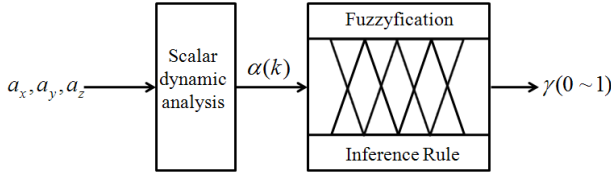


Fig. 7. Structure of movement detector

The scalar dynamic acceleration ( $\alpha(k)$ ) is defined as:

$$\alpha(k) = \left| \sqrt{a_x(k)^2 + a_y(k)^2 + a_z(k)^2} - 1 \right| \tag{10}$$

Fuzzy rules have the following form:

$$\text{If } \alpha(k) \text{ is } A_i, \text{ then } \gamma = \gamma_i \tag{11}$$

Where  $A$ 's is MF,  $\gamma$  (the possibility of the movement) is the output ( $\gamma=0$  means immobile and  $\gamma=1$  means mobile), and the subscript  $i$  denotes the rule number. The MF's of the fuzzy sets for  $\alpha(k)$  is shown in Fig. 8. Note that  $\gamma_i$  may be considered as a singleton MF as shown in Fig. 8.

$$\gamma = \sum \mu_i(\alpha) \cdot \gamma_i \tag{12}$$

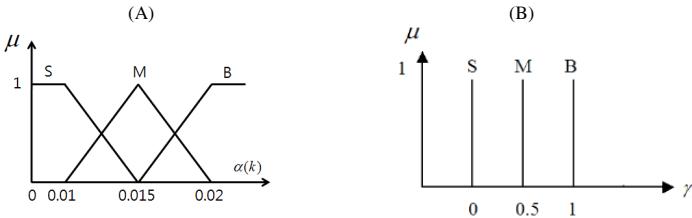


Fig. 8. Membership functions (A) and Singleton membership function (B)

### 4.3 Estimator

After these two parallel steps, a final decision has to be taken. At this point, all the needed values are known (outlier probability ( $\beta$ ) and probability of movement ( $\gamma$ )). The resulting filtered acceleration ( $a_{filt}(k)$ ) is determined by (13) with defuzzified values,  $\beta$  and  $\gamma$

$$a_{filt}(k) = \gamma \cdot ((1 - \beta) \cdot a_{meas}(k - 1) + \beta \cdot a_{meas}(k - 2)) \tag{13}$$



This outcome indicates that there is an improvement in the error reduction (as shown Table 2) with the aid of the proposed method. Fig. 9 is a comparison between the original and the total practical trajectory result.

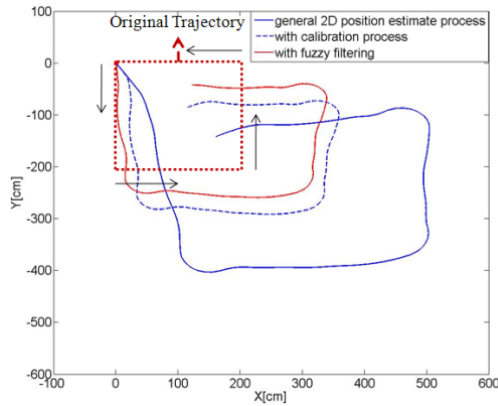


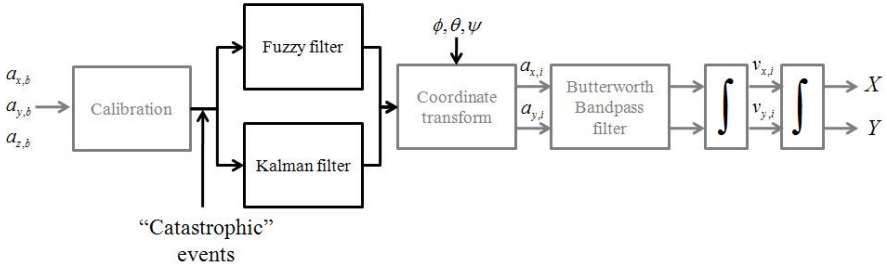
Fig. 9. Comparison of total trajectory test results

Table 2. Error reduction at the end point for proposed method

	General method	Calibration process	Fuzzy filtering
<b>Error</b>	≈210cm	≈150cm	≈110cm
<b>Reduction</b>	-	28%	48%

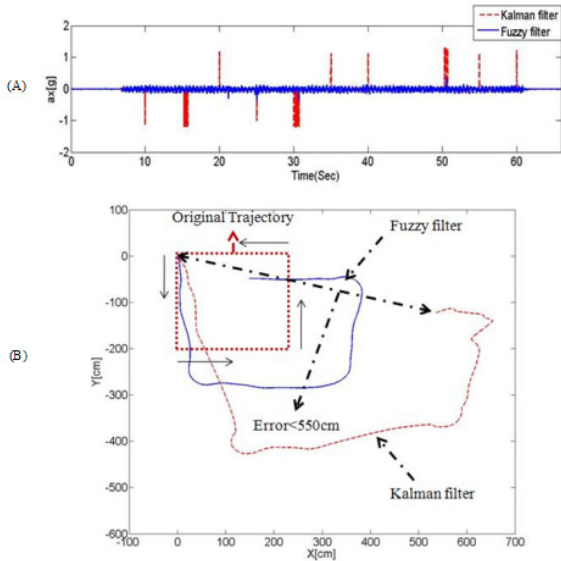
## 5 Comparison Simulation Result of Fuzzy Logic Based System with Kalman Filter

Fuzzy logic based position estimation system is fundamentally different from the well-known Kalman filter method. Kalman filters generally use statistical error models to predict the behavior of sensor components. A successful application of the Kalman filter is strongly dependent on whether the system dynamics is perfectly modeled in the mean sense. In practice, disturbances such as wind and current may have random variations (catastrophic events) superimposed on changing mean values (trend). This is very difficult task, if not impossible to model the system dynamics of a kinematics positioning system perfectly because the environment always changes. As far as measurement noise is concerned the system will require knowledge of the random errors, which will be assumed Gaussian. As a result, Kalman filter is not ideal for robotic fish system based on MEMS accelerometer. Fig. 10 shows a comparison simulation between setup Fuzzy logic based system and Kalman filter.



**Fig. 10.** Comparison simulation setup fuzzy logic based system with Kalman filters

Some random values into the data represent catastrophic events, such as collision or impact that can occur in natural underwater environment. Fig. 11(A) shows such data applying Kalman filter and Fuzzy filter. In case of Fuzzy filter, it removes the effect of the random values that describe “catastrophic” event. In case of Kalman filter, however, could not remove these effects. Fig. 11(B) is the simulation of 2D position estimation result by using the data from Fig. 11(A). This outcome indicates that Fuzzy filter is much robust in the “catastrophic” events compared with the famous Kalman filter method



**Fig. 11.** Outlier detector effect(A) and Comparison of trajectory test-with “catastrophic”event(B)

**Table 3.** Position error at the end point for between Fuzzy filter and Kalman filter

	Without catastrophic events	With catastrophic events
Fuzzy filter	≈110cm	≈110cm
Kalman filter	≈150cm	≈550cm

## 6 Conclusion

The paper presents an effective calibration method and a fuzzy logic based smart signal-processing algorithm in order to accurately estimate a 2D position for small robotic fish with a low-cost MEMS accelerometer. The main purpose of the paper is to minimize the drift error that is inevitable in the double integration process in accelerometer-only navigation system. The performance of proposed method is evaluated by the trajectory test. The proposed scheme has shown good characteristics for the drift errors from the MEMS IMU measurement as well as the noise corruption from the fuzzy filter in all dynamic conditions. The proposed scheme gave an accuracy of better than 110cm compared to the end point of original trajectory (navigation for 70 seconds). The test results are very encouraging since the estimation process was performed with only MEMS accelerometer.

The prime purpose of the paper is to make low-cost MEMS accelerometer sensor as accurate as possible via proposed calibration method and fuzzy-based filtering scheme. As a further study, the accelerometer will be fused with multisensory data such as USBL (Ultra Short Baseline) for our miniaturized robotic fish. To this end, extend Kalman filter and their variants will be used.

**Acknowledgements.** This research was supported from Development of Biomimetic Robot System Research Project funded by Korea Institute of Industrial Technology of the Korean Government.

## References

1. Hegrehaes, O., Berglund, E., Hsllingstad, O.: Model-aided Inertial navigation for underwater vehicles. In: Proceedings of the IEEE international conference on robotic and automation, PASADENA, pp. 1069–1076 (2008)
2. Yoo, T.S., Hong, S.K., Yoon, H.M., Park, S.S.: Gain scheduled complementary filter design for a MEMS based attitude and heading reference system. *Sensors* 11, 3816–3830 (2011)
3. Hong, S.K.: Fuzzy logic based closed-loop strapdown attitude system for unmanned aerial vehicle (UAV). *Sens. Actuat. A-Phys.* 107, 109–118 (2003)
4. Grewel, M.S., Henderson, V.D., Miysako, R.S.: Application of Kalman Filtering to the Calibration and Alignment of Inertial Navigation Systems. *IEEE Trans. Automatic Control* 36, 3–13 (1991)
5. Lauro, O., Giuloi, R., Daniel, C., Johann, B.: The FLEXnav Precision Dead-reckoning System. *International Journal of Vehicle Autonomous System* 4, 173–195 (2006)

6. Miniature Attitude Heading Reference System (AHRS) with GPS, 3DM-GX3-35; Micro strain: 459 Hurricane Lane, Suite 102, Williston, VT 05495 USA (2011)
7. Win, T.L., Kalyana, C.V., Wei, T.A.: Drift-free position estimation of periodic or quasi-periodic motion using inertial sensors. *Sensors* 11, 5931–5951 (2011)
8. Park, M.H., Yang, G.: Error analysis and stochastic modeling of low-cost MEMS accelerometer. *The Journal of Intelligent and Robotic Systems* 46, 27–41 (2006)
9. Hong, S.K.: A Fuzzy Logic based Performance Augmentation of MEMS Gyroscope. *Journal of Intelligent & Fuzzy Systems* 19(6), 393–398 (2008)

# View Planning of a Multi-rotor Unmanned Air Vehicle for Tree Modeling Using Silhouette-Based Shape Estimation

Dae-Yeon Won<sup>1</sup>, Ali Haydar Göktoğan<sup>2</sup>, Salah Sukkarieh<sup>2</sup>, and Min-Jea Tahk<sup>1</sup>

<sup>1</sup> Division of Aerospace Engineering, KAIST,  
Daejeon, Republic of Korea

{dywon,mjтахk}@fdcl.kaist.ac.kr

<sup>2</sup> Australian Centre for Field Robotics, University of Sydney,  
Sydney, Australia

{a.goktogan,s.sukkarieh}@acfr.usyd.edu.au

**Abstract.** The use of a multi-rotor unmanned air vehicle (UAV) in image acquisition tasks is promising for three-dimensional (3D) object modeling. Such an autonomous data acquisition system can be useful to handle the geometric complexity of objects such as trees and the inherent difficulties of image capture. In this paper, we address the problem of view planning for a camera-equipped multi-rotor UAV to acquire an adequate set of images that leads to more detailed and complete knowledge of the 3D tree model. The proposed algorithm based on shape-from-silhouette methods incorporates both expected new visual information and vehicle movement. Occupancy estimation for volumetric object model serves as a baseline measure of new information. The outlined approach determines next best views across the viewpoint space bounded by the sensor coverage and the capability of the UAV with minimal a priori knowledge of the object. Simulation studies conducted with virtual reality environments show the effectiveness of the algorithm.

## 1 Introduction

Three-dimensional (3D) geometry of trees including outlines, shape, and height is widely required for scientific and commercial applications. In computer graphics, the use of realistic tree modeling provides a significant level of virtual scenes. In weed biological control, on the other hand, the geometric information of trees is important to improve detection, classification, and quantification between several species [1].

In this context, image-based tree modeling, the process of reconstructing 3D tree models from photographs, has been studied in computer vision area [2]–[6]. Image-based 3D modeling process typically involves taking multiple view images, reconstructing camera pose with sparse geometry, and registering geometric data in a reference frame [7,8]. These image-based modeling methods are based on structure from motion techniques that use correspondences between 2D image measurements to recover 3D information. Recently, digital image-based modeling software such as Autodesk's Photofly [9] or Microsoft's Photosynth [10] is getting popular and readily available online for the purpose of object modeling and exploring unstructured environments.

Although automated reconstruction process reduced the amount of human work, multiple image acquisition from views is still done manually via trial-and-error approach to achieve acceptable level of modeling results. Also, if the geometric complexity of trees increase, it becomes more difficult to get a proper set of images by hands. Concerning the data acquisition performance, employing a multi-rotor unmanned air vehicle (UAV) suits well for this purpose in terms of mobility and repeatability. An intelligent view planning mechanism designed to equip the UAV for the automated data acquisition is required to guarantee the fidelity of 3D tree reconstruction.

In this paper, we address the problem of view planning for 3D tree reconstruction using a camera-equipped multi-rotor UAV. To locate next viewpoints for improving reconstruction quality, shape-from-silhouette based approach [11] is employed. Since the silhouettes of a tree depend on both the tree itself and the region allowed to the viewpoints, backprojecting of these 2D information can be sources of volumetric information. As the UAV captures a number of views in different viewpoints, the reconstruction quality can be measured by shape estimation from multiple silhouettes. On the other hand, structure from motion approaches, which generates 3D point cloud, is challenging due to trees' immensely complex structure. The presented approach considers utilisation of a multi-rotor UAV platform capable of flying to a given way point and hovering there while holding the commanded heading. The platform will be carrying a light weight, low-cost, passive camera instead of expensive and relatively heavier active sensors such as laser scanners [12]. The performance of this approach is characterized by applying the algorithms to a simulated tree model in a virtual environment. Paper is organized as follows; Section 2 describes the problem statements, Section 3 discusses some related work. The proposed view planning algorithm for silhouette-based tree model reconstruction is addressed in Section 4. Section 5 provides the simulation model and the numerical results. The final section summarizes the overall discussion.

## 2 Problem Statement

The main aim of this paper is to develop a solution for the automated image acquisition of a single tree by using a small multi-rotor UAV equipped with a low-cost, camera. Fig. 1 illustrates this image acquisition task in outdoor field. Fig. 2 shows a multi-rotor UAV platform equipped with a single camera and one of captured images from the camera during manual flight. It is assumed that the geolocation of the tree is known a priori and the desired distance between the centre of the tree and the UAV is given. Because of on-board computational capabilities of conventional small-size UAVs, the live video stream from the on-board camera is transmitted to a ground control station for image processing. When the computation of next best viewpoints is completed, the desired viewpoint is uploaded to the flight control system. Also, we assume that the UAV always finds collision-free paths over desired viewpoints.



**Fig. 1.** Image acquisition task for 3D tree model reconstruction using a multi-rotor UAV



**Fig. 2.** A multi-rotor platform equipped with a single camera (left) and an image taken by the platform during manual flight (right)

### 3 Related Work

Finding an optimal viewpoint plan necessary for the best possible reconstruction is an intractable problem and addressed by researchers as next-best-view (NBV) problem [13]. Following is a brief summary of view planning schemes that have been proposed for robotic applications.

Sujan et al. [14] conduct a study on view planning for unknown, unstructured environment mapping by cooperative sensing. To determine the agent's next best view, the authors considered optimizing a rating function over new information about the environment based on Shannon's information theory. Paul et al. [15] also propose a view planning algorithm for an industrial robot manipulator based on information theory. Their exploration approach incorporates multiple objectives to minimize manipulator movement and maximize its safety. Schmid et al. [16] address the problem of model-based view planning for 3D reconstruction of outdoor sites based on a 2.5D digital surface model. Applicable viewpoints from the list of possible viewpoints are chosen by heuristic constraints imposed by the multi-view stereo algorithm. This approach gives manual viewpoint planning guidelines resulted from reconstruction experiments.

Although the above mentioned algorithms claim good performance, most of them concerned with the use of range measurements. As opposed to those works, view

planning algorithms using a number of images have been proposed with silhouette-based approaches [17][18][19]. While the resulting methods have been well established theoretically and applied to various class of objects, these work largely deal with the interactive object-specific reconstruction problems for convex polyhedra.

## 4 View Planning Algorithm

### 4.1 Overview

As described in Fig. 3, our automated data acquisition process consists of four stages. The first stage is to construct a volumetric hull and define boundaries of viewpoint space based on a priori knowledge of the tree of interest. The 3D tree model is considered as a probabilistic discretized occupancy grid. In the second stage, the silhouette of the tree is extracted by partitioning an input image into the tree and background. In the third stage, occupancy values of the volumetric model is updated up to the current viewpoint using silhouette-based shape estimation. If the end criteria of view planning is not satisfied, the final stage is to find the next best view by defining and optimizing a rating function over the possible viewpoint space and move the UAV to the desired pose.

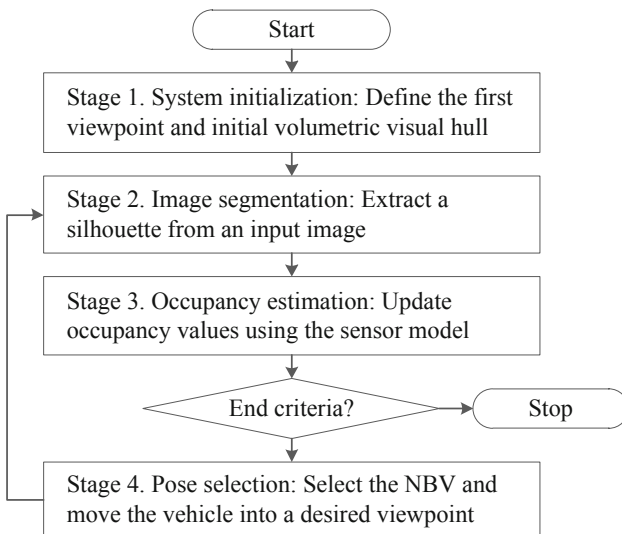


Fig. 3. Outline of the view planning algorithm

### 4.2 System Initialization

As a first stage, a circular path of the UAV is designed as a baseline for view planning. The path parameters including radius and height are bounded by the sensor's field of view to cover the volumetric object model entirely. These parameters are also limited



by the UAV's operating range. The closest point on the circular path is taken as the first viewpoint. The dimension of the voxel model depends on both computational loads and the size of the object given as a priori knowledge. The volume of the object is stored in an octree data structure, which is a common method for 3D object modeling from a set of images [18,20,21]. In contrast to other modeling approaches in 3D such as a point cloud, elevation map or multi-level surface map, an octree-based approach is suitable for a multi-resolution representation in probabilistic occupancy modeling whereas it has the problem of memory consumption and over-confidence in the model [22]. During an image acquisition task, the octree is initialized as a cube bounding the object, and recursively subdivided all occupied voxels down to the maximum subdivision level as a number of views are added.

### 4.3 Image Segmentation

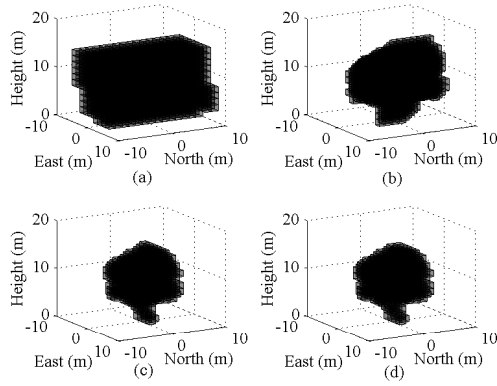
In the second stage, the input image from the on-board camera is segmented into the tree of interest and the background. The silhouette of the object, obtained as a perspective projection onto the image plane, provides 3D geometric shape understanding as the number of silhouettes increase [11]. To extract the object's silhouette, classifier based algorithms such as graph cuts or support vector machines have been applied to a wide range of problems in object segmentation [23,?,?]. Typically these algorithms can be carried out in polynomial time with minimal user interaction. Hence, these approaches are not considered in this paper. Fig. 4 shows an example of image segmentation for a tree object.



**Fig. 4.** Image segmentation. An input image (left) is segmented into the tree and the background (right) by using an image segmentation algorithm. The extracted silhouette is used to construct the volumetric model of the object.

### 4.4 Occupancy Estimation

In the third stage, the 3D object model is computed by occupancy estimation up to the current viewpoint. If object's silhouettes and corresponding pose measurements are obtained with absolute confidence, this stage can be considered as shape modeling by volume intersection. In this case, intersection of the outline cones backprojected from all silhouettes carves free space lied outside of silhouettes and reveals the closest approximation of the tree model referred to as the visual hull [21]. The intersection process



**Fig. 5.** Progression of the visual hull from (a) to (d) constructed by the intersections of volumes of four different silhouette cones

identifies whether each voxel lies entirely inside, entirely outside, or partially intersects the tree. Although the resulting visual hull cannot capture some of the details, it captures most of the shape appearance-defining elements [2]. Fig. 5 depicts an example of developing the visual hull for a simulated tree model using octree representation.

As the observations from the UAV system are not necessarily error-free, a statistical occupancy grid model is employed to carve the volumetric model by fusing multiple silhouette measurements. We calculate the posterior probability for each voxel of a discrete grid instead of intersecting volumetric cones directly. Each cube in octree representation has its occupancy probability based on whether its projection onto the image plane lies inside or outside of silhouettes from a view plan. This approach gives a constant-time access to grid cells and provide a proper way to represent uncertainties [26]. We consider that the model  $m$  consist of a discrete, three-dimensional grid of  $L$  voxels denoted as  $m_1, \dots, m_L$ . The measurement  $z_k$  is backprojected space from the 2D silhouette of the  $k$ th image at the corresponding viewpoint  $x_k$ . Given independence assumptions for static environments with a non-changing object model, a posterior probability  $p(m_l|x_{1:k}, z_{1:k})$  is estimated recursively as follows.

$$p(m_l|x_{1:k}, z_{1:k}) = \left( 1 + \frac{1 - p(m_l|x_k, z_k)}{p(m_l|x_k, z_k)} \cdot \frac{p(m_l)}{1 - p(m_l)} \cdot \frac{1 - p(m_l|x_{1:k-1}, z_{1:k-1})}{p(m_l|x_{1:k-1}, z_{1:k-1})} \right)^{-1} \quad (1)$$

This updates voxels that fall into the sensor's field of view. To prevent overconfident state of the model, a clamping update policy is applied to the occupancy estimation [27]. The clamping update policy bounds the maximum and minimum probability so that it makes voxels remain updatable by adding new observations until the end of the task.

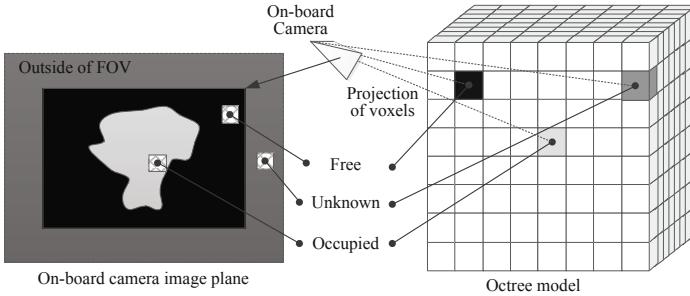
The update equation of the clamping update policy is following:

$$p(m_l|x_k, z_k) = \max(\min(p(m_l|x_k, z_k), p_{\max}), p_{\min}). \quad (2)$$

The inverse sensor model  $p(m_l|x_k, z_k)$  strongly depends on the type of sensor and its application environment. For 3D reconstruction from 2D image sensors, occupied voxels are classified by backprojecting a silhouette from the sensor, unlike ray-casting operations which use range sensors. If a voxel falls inside the silhouette's backprojection cone, the voxel is classified as occupied space in the octree representation. This inverse sensor model can be formulated as

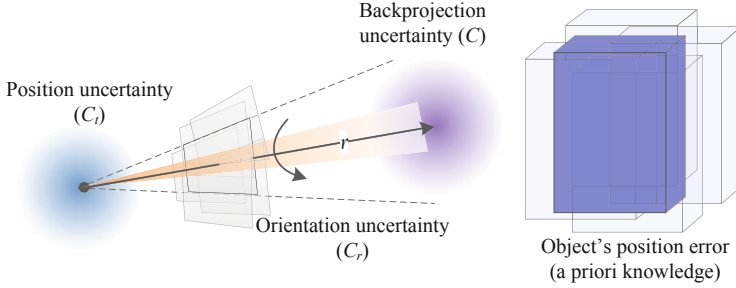
$$p(m_l|x_k, z_k) = \begin{cases} p_{\text{prior}} & , \text{ outside of the field of view} \\ p_{\text{occ}} & , \text{ inside of backprojection} \\ p_{\text{free}} & , \text{ outside of backprojection} \end{cases} \quad (3)$$

It holds  $0 \leq p_{\text{free}} \leq p_{\text{prior}} \leq p_{\text{occ}} \leq 1$ . Fig. 6 illustrates an example of voxel classification for 2D image sensors. This method assigns an occupancy value of  $p_{\text{occ}}$  to all voxels whose projection onto the viewpoint lies within the silhouette.



**Fig. 6.** Inverse sensor model for updating occupancy values of the octree model. Voxel-based shape estimation is updated by an occupancy classification of each volume element into one of the discrete states: free, unknown, and occupied.

To take the uncertainty in a real environment into account, our sensor model incorporates three types of noise and uncertainty sources: pose measurement noise, silhouette extraction errors, and unexpected tree location. As depicted in Fig. 7 those uncertainty sources are essential to making the sensor model with silhouette-based shape estimation. Pose uncertainty makes a difference between a computed view plan and what will actually be executed by the UAV's flight control system. If the effects of pose uncertainty are not handled properly, data acquisition by the view plan can be compromised. Image segmentation, whether manual or automatic, can be achieved only to a certain finite accuracy level. Any silhouettes extracted from images, therefore, are subject to measurement errors. Uncertainty in tree location is due to the GPS errors in the system initialization stage. This locational error may be critical when a planning process has a pre-programmed view plan without an online correction manner.

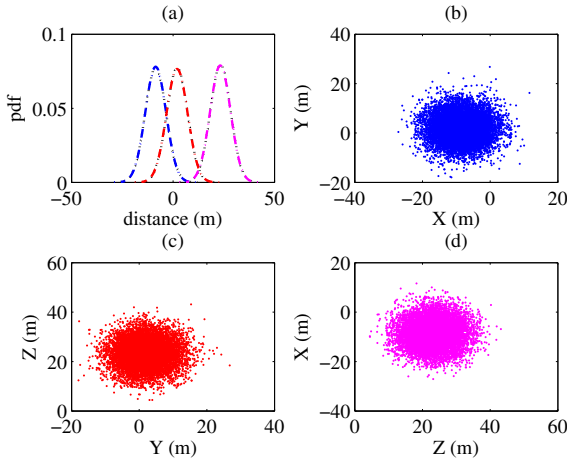


**Fig. 7.** Sources of uncertainty and its propagation to 3D points: pose measurement noise and unexpected position of an object.

Analytical expressions for the sensor uncertainty model should incorporate noise characteristics with the shape-from-silhouette technique. We consider how those errors in measurement systems propagate through the geometry in order to quantify the uncertainty on the final backprojected 3D point. The model can be simplified as

$$C = C_t + r^2 R(\theta) C_r R(\theta)^T \quad (4)$$

where  $C$ ,  $C_t$ , and  $C_r$  are covariance matrices of the sensor model, position measurement, and orientation measurement noises, respectively.  $r$  is the distance to the environment point from the body frame, and  $R(\theta)$  is the rotation matrix from the world frame to the body frame by the Euler angle  $\theta$ . Errors due to image segmentations, which has erosion



**Fig. 8.** Monte Carlo simulation for the uncertainty analysis: (a) the analytical (dashed) and simulated (dotted) distribution of the computed distance for each axis of the world coordinate. The two curves corresponding to each axis are almost perfectly overlapping. (b), (c) and (d) are distributions of the simulated data points backprojected from the viewpoint to the center of image plane.

or dilation effects on segmented silhouettes, are modeled as a part of orientation errors. Since the orientation measurements are made in the camera coordinate frame, a transformation of covariance parameters from the observation frame to the world frame is required. From this probabilistic backprojection of silhouettes into the volumetric tree model, voxels inside of 1-standard deviation from the backprojected point are considered as occupied space. Fig. 8 shows a Monte Carlo simulation of this model for the backprojected 3D point placed at 25m distance from its corresponding viewpoint. The analytical and simulated distributions are almost perfectly overlapping.

#### 4.5 Pose Selection

The final stage finds the next best view for the image-based 3D reconstruction until the end criteria are satisfied. View planning process can be terminated on two conditions: travelling around its circular path once or the low rate of carved volume by an added view. A quality viewpoint is selected from viewpoint candidates in a manner of optimization that incorporates both expected new visual information and distance travelled by the UAV. For the current occupancy probability model, the projected area  $S$  of occupied voxels onto the image plane of a possible viewpoint and the corresponding distance  $d$  to be moved by the UAV are combined into a single objective function formulated as

$$f = w_S \cdot S + w_d \cdot d^{-1} \quad (5)$$

where  $w_S$  and  $w_d$  are weighting coefficients. Maximizing the first term over a set of viewpoints promotes the pursuit of new geometric information to reveal remaining free space inside the current visual hull. The evaluation of  $S$  projects voxels which has the occupancy probability higher than  $p_{occ}$ . The second term of the objective function aims to minimize the movement of the UAV from the current then to the next pose. Fig. 9 depicts this evaluation of expected information from a viewpoint candidate.

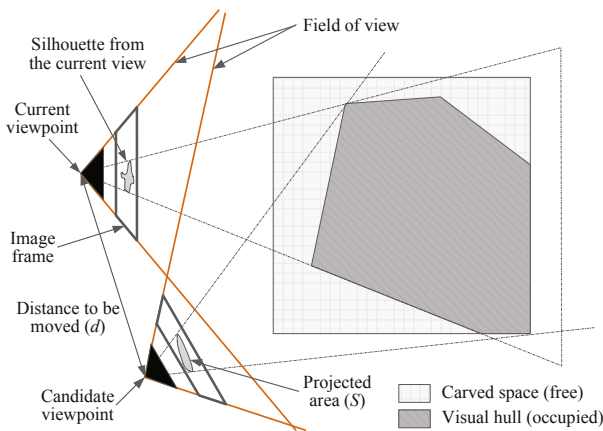
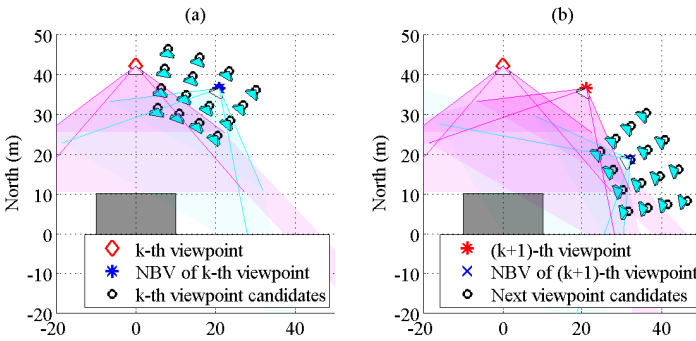


Fig. 9. Evaluation of the objective function for a viewpoint candidate

Since the objective function cannot be optimized analytically, finding an optimum next viewpoint requires searching algorithm over a set of uniformly distributed candidate viewpoints in the environment. While determining an appropriate viewpoint, the viewpoint candidates for the next pose are restricted by two constraints: a coverage constraint and a maximum angle constraint. The coverage constraint indicate that certain part of an object has to be seen in the field of view. The maximum angle constraint means that the angle between two consecutive views to the center of the volumetric model should not exceed the maximum angle. Fig 10 describes the geometry of the pose selection scheme. A number of geometric entities such the viewpoint region constraints and the maximum angle constraint are fixed by considering capabilities of both the UAV and the image sensor with a priori knowledge of the tree.



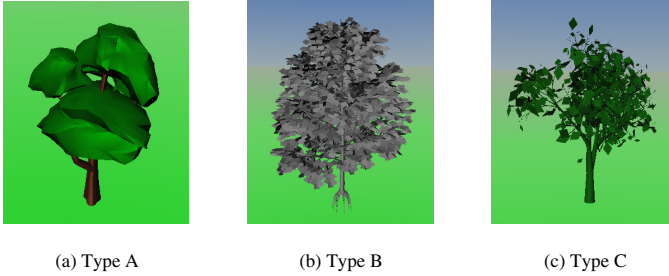
**Fig. 10.** Viewpoint selection scheme: (a) and (b) show top-down view with viewpoints picked out for next-best-view at each step. Viewpoint candidates are indicated by their centers (circle) and pyramids.

## 5 Numerical Results

Here we give the numerical results of the proposed view planning algorithm. Three viewpoint selection methods are compared. The view plan of the first method is selected randomly within viewpoint candidates for each step. The second method selects the next camera viewpoint sequentially from a set of uniformly gridding viewpoints on the circular path. In the third method, the next camera viewpoint is selected with maximum expected information by optimization.

A 3D graphics scene including a tree is generated by using VRML model that is directly incorporated in MATLAB. Because trees come in many different shapes and sizes, we use three types of simulated trees in the virtual environment. Those tree models have distinctive visual parts of trunk, branches, leaves, and twigs as shown in Fig. 11. The tree of interest, which is around 15 m tall and a 10 m in diameter, is located at (7m, 5m, 0m) in a local East-North-Up (ENU) coordinate system whereas the expected tree location is the origin. The initial octree cube from a priori knowledge is set to  $30\text{ m} \times 30\text{ m} \times 30\text{ m}$  with appropriate margins for unexpected tree location. Viewpoint candidates across the free space bounded by the size of the octree cube and design parameters.

Since we employ the circular path as a baseline, the viewpoint space has a doughnut shape. The heading angle of the UAV and the tilting angle of the vision system frame are determined to point the center of the volumetric tree model. The occupancy probability of all voxels is initialized to the uniform prior of  $p_{prior} = 0.5$  and we used values of  $p_{occ} = 0.8$  and  $p_{free} = 0.2$  for occupied and free voxels, respectively. The clamping thresholds are set to  $p_{max} = 0.9$  and  $p_{min} = 0.1$ . The uncertainties of the position and orientation are set to standard deviations of 5 m and 3 degrees, respectively.

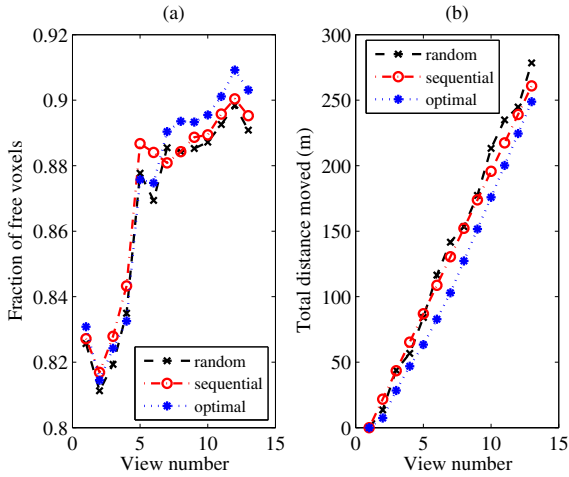


**Fig. 11.** Three different types of tree in virtual environment

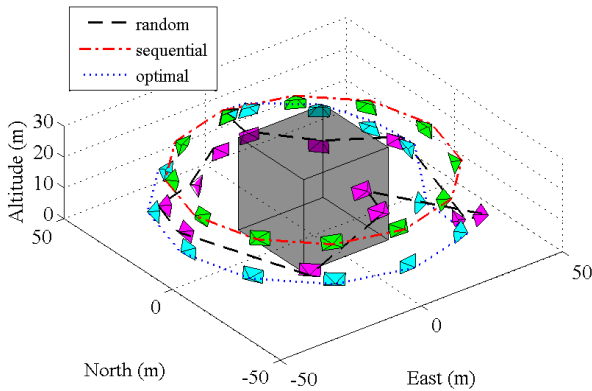
As a numerical example, Fig. 12 shows the fraction of free voxels and the distance travelled by the UAV for the three planning schemes in the case of Type A. Fig. 13 describes corresponding paths of viewpoints planned by the three selection methods and Fig. 14 depicts the 3D occupancy grid models for the visual hull made by all viewpoint candidates and the proposed view planning method. Under the 30 degrees maximum angle constraint, the total of 12 viewpoints is planned from the first viewpoint located at (0 m, 42m, 21m). As the number of views increases, the visual hull becomes the closest approximation of the tree whereas the rate of carved volume decreases. It is desirable to have a large fraction of the visual hull modeled with small displacement.

In order to compare the proposed view planning method against other two conventional methods, relative modeling error and total distance moved are estimated over 100 Monte Carlo runs using the same parameter values in a previous example. The relative modeling error is measured with respect to the visual hull estimated by all possible viewpoints. The resulting relative errors and total distance are presented in Table 1. From the results, it can be seen that our method provides slightly better final reconstruction accuracy in terms of total distance travelled by the UAV. On the other hand, one can observe that the overall performance of the proposed method is degraded as the tree of interest has less leaves and thinner branches. Since this limitation is, fundamentally, a consequence of using octree representation, the trade-off between computation efficiency and the level of octree data structure is required to manage an image acquisition task.

While the proposed method allows to capture a proper set of images via probabilistic occupancy estimation, this approach remains the problem of handling uncertainty in a



**Fig. 12.** Comparison of the three viewpoint selection methods for modeling an unknown tree: (a) fraction of free voxels as a function of view number. (b) total distance travelled by the multi-rotor UAV as a function of view number.

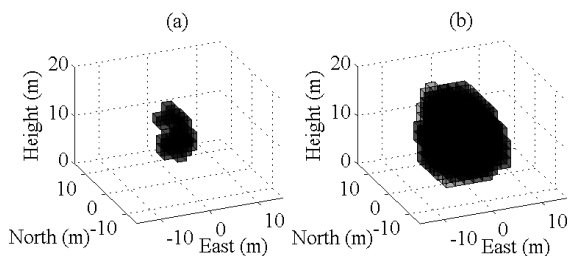


**Fig. 13.** Path of viewpoints planned by the three selection methods of view planning for modeling an unknown tree.

**Table 1.** Performance of the three view planning methods

View planning method	Relative modeling error (%)			Total distance moved (m)		
	Type A	Type B	Type C	Type A	Type B	Type C
Random	1.15	1.79	0.391	276.8	275.2	263.4
Sequential	1.37	1.92	0.360	260.9	260.9	260.9
Optimal	1.23	1.79	0.385	269.7	245.9	269.7





**Fig. 14.** The probability occupancy model for Type A from the overall viewpoints in the error-free situation (left) and the optimal selection method under measurement uncertainty (right). The probabilistic model of (b) is enlarged by uncertainty during occupancy estimation.

view plan. Reducing pose uncertainty, however, can be achieved by a localization and mapping approach with prior knowledge of known landmarks. Interactive or learning techniques for the image segmentation stage also can be a practical alternative method of dealing with uncertainty in silhouette segmentation.

## 6 Conclusions

There have been research efforts to select the next best viewpoint of 2D image sensors in unknown objects via automated process. In this paper, we proposed a view planning algorithm for image-based tree modeling using a camera-equipped multi-rotor UAV. The focus is on view planning of the passive sensor for improving reconstruction quality because most of the previous works have dealt with range images from active sensors. The possible next viewpoint is guided by the geometric concept of visual hull obtained by backprojecting the tree's silhouettes at the corresponding viewpoints. The outlined approach provides a useful framework for both modeling of the tree and planning of viewpoints by constructing of the space occupancy grid from multiple different views. The numerical results in virtual environments confirm that the proposed approach has capability to locate efficient next viewpoint in terms of expected new information and total distance travelled by the multi-rotor UAV.

**Acknowledgement.** This work is supported by the ARC Centre of Excellence programme, funded by the Australian Research Council (ARC) and the New South Wales State Government. Also, part of this work is sponsored by the Korea Aerospace Research Institute (KARI) under the Vision-Based Collision Avoidance Techniques for UAVs Project.

## References

1. Göktoğan, A.H., Sukkarieh, S., Bryson, M., Randle, J., Lupton, T., Hung, C.: A Rotary-wing Unmanned Air Vehicle for Aquatic Weed Surveillance and Management. *J. Intell. Robot Syst. 57*, 467–484 (2012)

2. Shlyakhter, I., Rozenoer, M., Dorsey, J., Teller, S.: Reconstructing 3D Tree Models from Instrumented Photographs. *IEEE Computer Graphics and Applications* 21, 53–61 (2001)
3. Reche-Martinez, A., Martin, I., Drettakis, G.: Volumetric Reconstruction and Interactive Rendering of Trees from Photographs. *ACM Trans. Graph.* 12, 720–727 (2004)
4. Tan, P., Zeng, G., Wang, J., Kang, S.B., Quan, L.: Image-based tree modeling. *ACM Trans. Graph.* 26 (2007), doi:10.1145/1276377.1276486
5. Neubert, B., Franken, T., Deussen, O.: Approximate Image-based Tree-modeling Using Particle Flows. *ACM Trans. Graph.* 26 (2007), doi:10.1145/1275808.1276487
6. Binney, J., Sukhatme, G.S.: 3D Tree Reconstruction from Laser Range Data. In: *Proceedings of the 2009 IEEE International Conference on Robotics and Automation*, Kobe, Japan, pp. 3183–3188 (2009)
7. Remondino, F., El-Hakim, S.: Image-based 3D Modelling: A Review. *The Photogrammetric Record* 21, 269–291 (2006)
8. Snavely, N., Seitz, S.M., Szeliski, R.: Photo tourism: exploring photo collections in 3D. *ACM Trans. Graph.* 25, 835–846 (2006)
9. Photofly, Autodesk,  
<http://labs.autodesk.com/technologies/photofly/>
10. Photosynth, Microsoft, <http://photosynth.net/>
11. Laurentini, A.: The Visual Hull Concept for Silhouette-Based Image Understanding. *IEEE Trans. Patt. Anal. Machine Intell.* 16, 150–162 (1994)
12. Lin, Y., Hyypä, J., Jaakkola, A.: Mini-UAV-Borne LIDAR for Fine-Scale Mapping. *IEEE Geoscience and Remote Sensing Letters* 8, 426–430 (2011)
13. Scott, W.R., Roth, G., Rivest, J.F.: View Planning for Automated Three-dimensional Object Reconstruction and Inspection. *ACM Comput. Surv.* 36, 64–96 (2003)
14. Sujan, V.A., Dubowsky, S.: Efficient Information-based Visual Robotic Mapping in Unstructured Environments. *International Journal of Robotics Research* 24, 275–293 (2005)
15. Paul, G., Liu, D., Kirchner, N., Dissanayake, G.: An Effective Exploration Approach to Simultaneous Mapping and Surface Material-type Identification of Complex Three-dimensional Environments. *J. Field Robotics* 26, 915–933 (2009)
16. Schmid, K., Hirschmüller, H., Domel, A., Grix, I., Suppa, M., Hirschinger, G.: View Planning for Multi-View Stereo 3D Reconstruction Using an Autonomous Multicopter. *J. Intell. Robot Syst.* 65, 309–323 (2011)
17. Shanmukh, K., Pujari, A.K.: Volume intersection with optimal set of directions. *Pattern Recognition Letters* 12, 165–170 (1991)
18. Nitya, V.B., Sridevi, N., Pujari, A.K.: Linear octree by volume intersection using perspective silhouettes. *Pattern Recognition Letters* 13, 781–788 (1992)
19. Bottino, A., Laurentini, A.: What's NEXT? An interactive next best view approach. *Pattern Recognition* 39, 126–132 (2006)
20. Szeliski, R.: Rapid Octree Construction from Image Sequences. *CVGIP: Image Understanding* 58, 23–32 (1993)
21. Fitzgibbon, A.W., Cross, G., Zisserman, A.: 3D Structure from Multiple Images of Large-Scale Environments, pp. 23–32. Springer, Berlin (1998)
22. Wurm, K.M., Hornung, A., Bennewitz, M., Stachniss, C., Burgard, W.: OctoMap: A probabilistic, flexible, and compact 3D map representation for robotic systems. In: *Proc. of the ICRA 2010 Workshop on Best Practice in 3D Perception and Modeling for Mobile Manipulation* (2010)
23. Rother, C., Kolmogorov, V., Blake, A.: GrabCut: interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.* 23, 309–314 (2004)

24. Yuri, B., Gareth, F.: Graph Cuts and Efficient N-D Image Segmentation. *International Journal of Computer Vision* 70, 109–131 (2006)
25. Bishop, C.M.: *Pattern Recognition and Machine Learning*. Springer (2009)
26. Thrun, S., Burgard, W., Fox, D.: *Probabilistic Robotics*, pp. 279–398. MIT Press (2005)
27. Yguel, M., Aycard, O., Laugier, C.: Update policy of dense maps: Efficient algorithms and sparse representation. *Field and Service Robotics* 42, 23–33 (2008)

# Design and Field Testing of Water Quality Sensor Modules Designed for Round-the-Clock Operations from Buoys and Biomimetic Underwater Robots

Joongki Park, Juchan Sohn, Sunghoon Kim, and Jonghyun Park

Intelligent Robot Control Research Team,  
Department of Robot/Cognitive Convergence System, ETRI  
138 Gajeongno, Yuseong-gu, Daejeon, 305-700, Korea  
{jkip, jcsohn, saint, jhp}@etri.re.kr

**Abstract.** At the point of monitoring coverage, the legacy technologies of water quality measurement show the weakness. It manually or semi-automatically measures various water quality information at just only small parts of a large area. In this paper, we propose new water quality sensor apparatus which can consistently provide high resolution water quality data as well as attachable and PnP type deployable to underwater robots and buoys. The water quality sensor module, as a result of going through three water system tests in Han River, Gold River and Gapchun respectively and benchmarking with the commercial water quality meter, shows not only outstanding performance but also receives an official experimental certification by the supervisor of the Korea Testing Laboratory.

**Keywords:** water quality sensor, robotic fish, biomimetics.

## 1 Introduction

The legacy water quality monitoring systems just measures manually in extremely small parts of a large location and measures through a semi-automated method [1][2][6]. For example, currently in 52 measurement stations in the 4 major rivers, only part items are measured amongst 5 common items (temperature, electric conductivity, pH, dissolved oxygen, total organic carbons) and 11 selective items (biology surveillance, flammable organic compounds, total nitrogen levels, total phosphorus levels, degree of turbidity etc.) and once a month water samples are manually measured through 1500 water quality measurement stations in national river/lakes/city/industrial areas. There are difficulties in maintenance and management as the sensors used for the measurement are high priced foreign-made [9].

Of foreign cases, in the case of New York's Hudson river in the US, a REON ((River & Estuary Observatory Network) project was jointly conducted from 2008 to implement a USN water pollution monitoring demonstration service by the Beacon laboratory and IBM, and apart from the wireless communication system, solar battery based self-charging mobile sensor nodes and the embedded compound sensors ((water temperature, EC, pH, DO, TOC)) are currently being developed [3][4][7].



Fig. 1. Domestic examples of water quality monitoring systems

In this paper, we recognize the problems of legacy technologies as mentioned above, use a biomimetic robot which can freely navigate underwater on not just parts of the river but the entire river area, and design a small size and easy to use water quality sensor module which is requested for continuous and high resolution water quality surveillance.

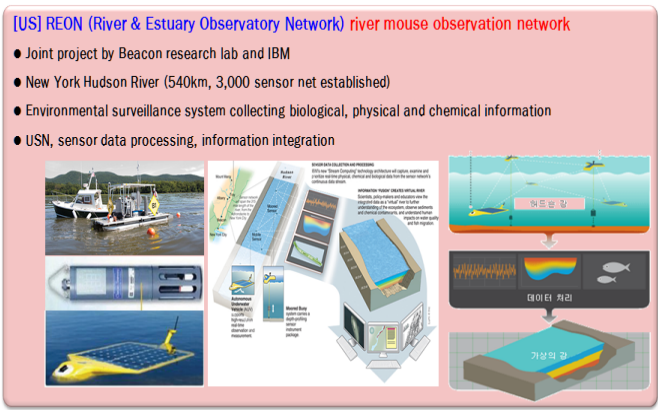


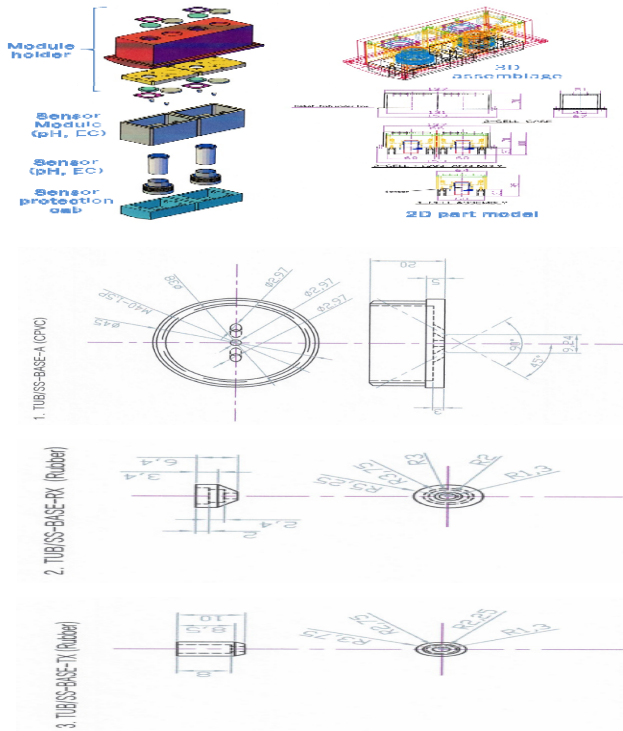
Fig. 2. New York’s Hudson River’s monitoring system

For example, our sensor modules are installed into a robotic fish as underwater vehicles to monitor water quality and pollution surveillance [5][8][10].

## 2 Design of Sensor Modules

Our apparatus of water quality measurement is a water quality sensor module in which continuous operation underwater is possible and is composed of the sensor module’s fixed shape, infrared light communication module, magnet, sensor module, sensor and sensor protection case.

The infrared light communication module is equipment which communicates sensing information and water quality sensor module and is designed with a wireless infrared communication module, not through a communication cable method connected with a cable. This is done to strengthen the operation ability under water and especially overcome invasions such as electric static of cables, thereby preventing distortion of electric signal treatment of water quality sensors.



**Fig. 3.** Assemblage and 2D/3D CAD Model

The sensor outline is a fixed type outline in order to place sensors, sensor protection case, sensor modules and is installed in the underwater moving object which is loaded with the water quality sensor module equipment. The underwater moving object refers to all moving objects that can independently swim underwater like biomimetic robot fish.

The magnet makes adhesion between the sensor module and module holder possible. A necessary and important requirement when constantly operating underwater is the simple maintenance capability. Taking these requirements into consideration, the magnet is used to organically connect the sensor module to the module holder. Screws instead of magnet can be installed because of prevention of interference to position sensor like gyro sensor or digital magnet compass.

The sensor module includes the sensor's control board which observes the water quality sensor types rooted in the sensor module and performs the program and circuit drive which fits in each sensor. The sensor's analogue signal is extracted and the

operation and operated data is transmitted through the sensor fixed outline and infrared light method. Furthermore, it forms the sensor's overall information status (temperature, remaining battery life, and sensor is workable or not).

The sensor protection case is to protect the sensor and is composed of a cover form in order to continuously operate underwater. For example in the case of pH sensor, when operating by holding the standard solution in a glass tube, the glass tube can be damaged due to the obstacles underwater or water pressure. In this regards, the sensor protection case mounts by taking these points into consideration and is designed to eliminate if necessary.

### 3 Implementation of Sensor Modules

At the issue of implementation point of view, we have to set up principles in regarding with module size and power source. The board size is minimized by package on package and system in package strategies respectively. The most important consideration in design of power source is to improve the easy to use in real environment. Execution model of our sensor module is designed by PnP style. As the sensor module should be attached and run by oneself to any kind of unmanned underwater vehicles, battery source is embedded into the sensor module. Currently, Li-Ion battery with 3.7v/970mAh is embedded.

Sensor modules are consisted with 3 on-board modules, which are signal detection and amplification board, signal processing board and communication relay board. The role of SDAB is to detect and amplify the analog signal come from sensor such as pH and EC. SPB as a main sensor board processes signals from SDAB and make the measured value of each sensor. SPB supports dual-mode signal processing, that is, if pH or EC sensors connect to the SPB, then SPB run as pH or EC modes. The measured values are transferred to CRB via IR communication protocol. CRB is basically installed robotic fish and fulfills role of communication repeater between sensor modules and main controller of robotic fish. Key features and specifications are as follows.

	Range	Resolution	Accuracy
Turbidity Sensor Module	0.0 to 1,000 NTU	0.1 NTU	± 2.0% of Full Scale
DO Sensor Module	0.00 to 20.00 mg/L	0.01 mg/L	± 3.0% of Full Scale
pH Sensor Module	0.0 to 14 pH	0.1 pH	± 1.0% of Full Scale
EC Sensor Module	0.0 to 3,000 $\mu$ S/cm	0.5 $\mu$ S/cm	± 2.0% of Full Scale

Fig. 4. Major specification and features of sensor module

### 4 System Integration

As a moving object which can make the water quality sensor module equipment to freely move underwater, the robotic fish is shown as a real life example. The sensor module equipment is installed in the abdomen of the robotic fish's body as the following.



Fig. 5. Conceptual view of installing sensor modules

The reason of installing in the abdomen is because the pH sensor and the dissolved oxygen sensor have transition standard solution, and in order to measure correctly, the standard solution should always be installed below the sensor [11][12]. Furthermore, if the light sensor such as the turbidity sensor or suspended solid sensor observes the water surface upside down or through its side, it is affected by the natural light (sunlight) and causes a defect in the measurement.

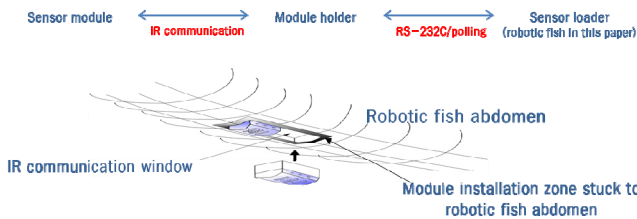


Fig. 6. Integration view of sensor modules

Module installation zone contains at most two different sensor modules respectively among pH, EC, DO, turbidity with marine thermometer. The module installment zone is fixed, and the fixed magnet is adhered to the sensor module and module installation zone. Communication between robotic fish and water quality sensor module is carried out by RS-232 Polling method. The communication protocol is designed as following.

#### 4.1 Communication Protocol

Classification	Specifications
Communication method	RS-232C asynchronous method
Transmission specifications	9600 bps, 8bit, Parity None, Stop bit 1
movement method	Polling method (robot fish-host, measurement tool-slave)
Data transmission method	Half Duplex, block transmission
Mark code	ASCII



## 4.2 Command Code

Command code (host to sensor module)	Specifications	Sensor module response (sensor module to host)
0x02DATA0x03XX	request for real time measurement data transmission	data transmission (FORMAT-1)
0x02CCHK0x03XX	search Correction value (Request for transmission of corrected data)	data transmission (FORMAT-2)

## 5 Real World Experiments and Results

In order to analyze the performance and on-site operational capability between our sensor module and the commercial multiple water quality meters, benchmark tests are conducted in real environment such as Daejeon' s Gapchun and Seoul' s Han River. Table 1 shows the results as follows.

**Table 1.** Results of benchmark test in real field

Daejeon Gold River	DO (mg/L)	Turbidity (NTU)	
Development Module	7.73 ~ 7.83	3.73 ~ 4.08	
Commercial Meter	7.53 ~ 7.95	3.35 ~ 3.69	
Error Rate	1.50 % ~ 2.65 %	9.56% ~ 10.19%	

Daejeon Gapchun	pH	Water Temp.	EC
Development Module	8.6 pH	11.7	320.40 $\mu$ S/cm
Commercial Meter	8.85pH	11.7	347.40 $\mu$ S/cm
Error Rate	2.8%	0%	7.7%

Seoul Han River	pH	Water Temp.	EC
Development Module	7.1 ~ 7.3 pH	16.4	189.0 ~ 190.5 $\mu$ S/cm
Commercial Meter	7.4 ~ 7.6 pH	17.3	193.2 ~ 195.4 $\mu$ S/cm
Error Rate	4.05% ~ 3.95%	5.20%	2.17% ~ 2.51%

According to the result of benchmark test in real field, it is convinced that both of pH and electric conductivity sensor module satisfies the targeted performance. Especially in the case of Han-river' s test site, the measurement value which was tough on the pH sensor' s water pressure and water depth changes in velocity 1.5m/s,

water depth 1m – 6m was shown to be within 20% of the relative accuracy requested for the formal approval of the environmental measurement equipment by the Agency for Technology and Standards regarding hydrogen ion concentration which uses electrodes.

In the case of Gapchun' s test site, the difference in the measurement value of the development module and the commercial meter lies in the differences in measured water depth. The development module is measured in water depth of at least 1m and the commercial meter is measured in water depth around 70cm due to limited cable length.

In the case of the Han River test, the development module is measured by changing the water depth from 0.3m to 6m and the commercial product is measured in water depth of around 70cm.

In order to simulate the robotic fish, the Gapchun test preceded the experiment by loading the sensor modules in the RF based remote control ship and the Han River test loaded the sensor module in a motor-embedded duck boat.

Our water quality sensor module received an authorized test report from the Korea Testing Laboratory and was recognized for the credibility on the measurement result value. The official test certification numbers are 11-1596-114-1 and 2 respectively.

## 6 Conclusion

Based on this research, compared to the legacy water pollution monitoring system which measured manual work of various river information in extremely small parts of a large location or measures through a semi-automated method, this system can be used as the main sensor equipment which can conduct water quality surveillance by attaching the equipment on moving objects which can freely move continuously underwater to conduct water quality surveillance such as the robotic fish. Accordingly, by persistently managing the river's water quality information as well as getting high resolution data of water quality, we can look forward to making prior actions before the river's environmental changes spread widely.

Future works are to system integration test based on biomimetic robots such as robotic fish incessantly in real fields.

**Acknowledgements.** This research has been supported by the Korea Research Council for Industrial Science and Technology (Project No. B551179-10-02-00). We thank the Daeyoon Scale Cooperation for support, especially real world experiment in Han River in Seoul and Gapchun and Gold River in Daejeon respectively.

## References

1. Sub, A.G.: Revised and resolved problems with th pH roof. *Understanding Sensors and Main Technology* 22(3), 281–290 (2010)
2. Shin, D., Na, S.Y., Kim, J.Y., Baek, S.-J.: Fish robots for water pollution monitoring using ubiquitous sensor networks with sonar localization. In: *ICCIT*, pp. 1298–1303 (2007)

3. Yoon, H.S.: Measurement of newly advancing water quality multichannels. In: C&I, pp. 15–19 (2008)
4. Kolar, H.R., Cronin, J., Hartswick, P., Sanderson, A.C., Bonner, J.S., Hotaling, L., Ambrosio, R.F., Liu, Z., Pawwow, M.L., Reath, M.L.: Complex real-time environmental monitoring of the Hudson River and estuary system. *IBM Journal of Research and Development* 53(3), 4:1-4:10 (2009)
5. Wen, L., Wang, T., Wu, G., Li, J.: A novel method based on a force-feedback technique for the hydrodynamic investigation of kinematic effects on robotic fish. In: ICRA, pp. 203–208 (2011)
6. Rickerby, D.G., Skouloudis, A.N.: Biosensor networks for monitoring water pollution. In: GHTC, pp. 276–282 (2011)
7. Zhang, S., Zhang, L.: Water pollution monitoring system based on Zigbee wireless sensor network. In: ICECC, pp. 1775–1779 (2011)
8. Waff, W.B., Wolfgram, P.A.: Field testing of a water quality sensor package designed for long-term operation from buoys and other unattended marine platforms. In: OCEAN, pp. 380–384 (1975)
9. Water Quality Measurement Management Team, National Water Quality Automatic Measurement Operation Status, Environment Management Corporation, pp.1-3 (2008)
10. Ye, X., Su, Y., Guo, S.: A Centimeter-scale autonomous robotic fish actuated by IPMC actuator. *Robotics and Biomimetics*, 262–267 (2007)
11. Zaborowski, M., Jaroszewicz, B., Tomaszewski, D., Prokaryn, P., Malinowska, E., Grygolowicz-Pawlak, E., Grabied, P.: Fabrication of MOS-Compatible Ion-sensitive devices for water pollution monitoring (warmer. In: MIXDES, pp. 477–481 (2007)
12. Sun, Z., Zhao, Y., Li, S.: Research on polarized remote sensing of monitoring of water pollution. In: ICBBE, pp. 1–5 (2010)

# Analysis on the Robotic Fish Propulsion for Various Caudal Fin Shapes

Dongwon Yun, Jinho Kyung, and Chanhum Park

Dept. of Robotics and Mechatronics, Korea Institute of Machinery & Materials  
156 Gajeongbuk-Ro, Yuseong-Gu, Daejeon, 305-343 Korea  
dwyun@kimm.re.kr

**Abstract.** In this paper, propulsion characteristic of various shape caudal fin is analyzed. Using added mass theory, some caudal fins for robotic fish propulsion are analyzed. From this analysis, torque and propulsion characteristic for caudal fins with various shapes can be investigated. This method can be adopted to identify the caudal fin performance more simply and faster than computational method, although with less accuracy.

**Keywords:** caudal fin, added mass theory, robotic fish.

## 1 Introduction

Recently, studies on the biomimetic robots have been increased. Biomimetics set its sights on developing better man-made systems by copying the shape and motion of animals and plants for more improved performance, because it is believed that all creatures have evolved for a long time to survive in wild life. For such a reason, studies on mimicking fish have also been increased and conducted widely in the field of biology, robotics, marine engineering and oceanography for underwater mission [1], [2], [3].

In nature, each fish has its own shape and motion underwater. The shape of a caudal fin and swimming mode varies for each fish [4]. This uniqueness is a product of long-time revolution according to its living habits and environments. Propulsion efficiency of about 91% of fish was reported [5]. Accordingly, various researches about robotic fish have been conducted with various shape, motion and functions. Previous studies on robotic fish mainly contain the mechanism, analysis, control, localization and sensor fusion for robotic fish [6], [7]. Each fish can propel itself in different ways. Some use a caudal fin and other use a pectoral fin and so on. Fish motion can be classified according to the body part used for propulsion.

In this paper, a caudal fin of fish was studied. Especially, the propulsion characteristics of various shape caudal fins are analyzed. Actually, there are too many kinds of fins fish uses for propulsion to consider all cases of fins. So, in this paper, three shapes of a caudal fin brought from caudal fins of a carp and a shark are investigated. To do this, added mass theory was used and torque and propulsion characteristic for some caudal fins with various shapes were investigated. This method can be adopted to identify the caudal fin performance more simply and faster than computational method, although it shows less accuracy surely.

## 2 Theory for Fish Propulsion

To analyze the actuation of caudal fin of fish, several methods can be used. Among those, CFD shows the most correct results, however it is not easy to model and simulate the caudal fin by using CFD also it spends much time to calculate the performance. So, lumped approach for analysis of fish swimming has been also studied widely. In this approach, water around the fins is modeled as mass or volume of water, and dynamics of water movement is investigated. In these researches, power and torque was calculated using simplified model [8], [9]. According to this theory, oscillating angle of fin can be expressed as follows

$$\theta = A \sin 2\pi ft \quad (1)$$

where  $A, f$  are oscillatory amplitude and frequency respectively. The time differential of formula (1) becomes

$$\dot{\theta} = 2\pi fA \cos 2\pi ft. \quad (2)$$

Then the velocity at the mass center of caudal fin can be

$$v_G = r \times \omega = \dot{\theta}r = 2\pi rf \cos 2\pi ft. \quad (3)$$

From above equations, force  $F$  which acts on the fin in the opposite direction of  $v_G$  and torque  $T$  at the fin hinge can be evaluated from equations,

$$F = C_D \rho V_G^2 S_A / 2 = C_D \rho (2\pi rf \cos 2\pi ft)^2 S_A / 2, \quad (4)$$

$$T = r \times F = 2C_D \rho S_A (\pi rfA \cos 2\pi ft)^2 r \quad (5)$$

where  $C_D$  represents the drag coefficient;  $\rho$  is water density;  $S_A$  is the largest cross section area of fin.

The mechanical power  $P$  for a fish can be calculated as [9]

$$P = mWwU - \frac{mw^2U}{2 \cos \theta} \quad (6)$$

where  $m$  represents added mass;  $W$  is rms of the lateral speed of the trailing edge;  $w$  is water velocity at the trailing edge;  $U$  is average speed of fish.

Thrust force  $T_h$  of a fish is evaluated from formula 6.

$$T_h = \frac{P_t}{U}. \quad (7)$$

In this paper, this added mass approach is used to calculate for the performance of the caudal fin with various shapes.

### 3 Analysis for Caudal Fin with Various Shapes

Figure 1 shows the view of a caudal fin of a fish. In this figure, y axis lies in the direction of head and x axis lies in the direction of the span.

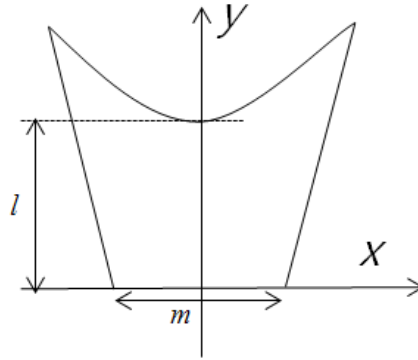


Fig. 1. Caudal fin

In this study, three kind shape of caudal fin are considered. One is symmetric and the others are asymmetric in the shape. To consider this shape differences, the model of figure 1 is assumed as assembly of lines and curves as figure 2.  $f_1(x)$  and  $f_2(x)$  are straight line and  $f_3(x)$  is a quadratic curve. In this figure, straight lines determine the boundary shape of a caudal fin and quadratic curve changes the shape of trailing edge. If the curve  $f_2$  moves in the x-direction, which means the center of curve moves, the leading edge shape can be asymmetric shape like dotted line in figure 2. So, by adjusting the center position of curve, various shapes of caudal fins can be obtained.

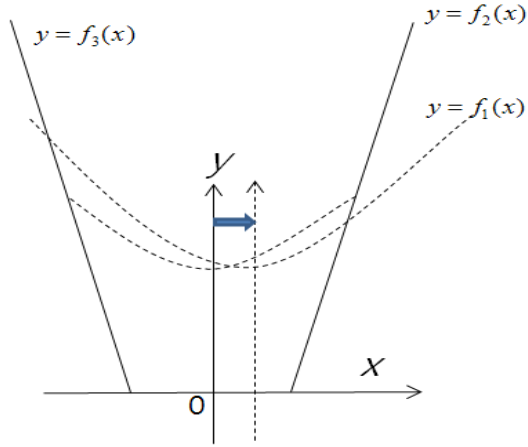
Let  $c$  as a x-coordinate of the center of the curve, the quadratic curve can be expressed as

$$y = a(x - c)^2 + b \tag{8}$$

where  $a, b$  is constants. And straight lines are also expressed

$$y = \pm dx - e . \tag{9}$$

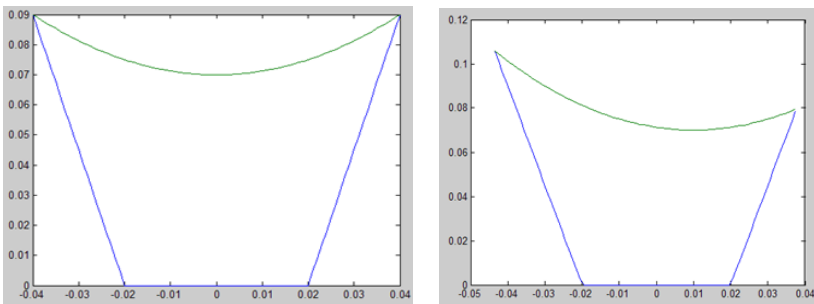
In this paper, the length of  $m$  is assumed as 40mm and  $l$ , 70mm. And the other parameters are assumed as Table 1. According to  $c$ , the shape of a caudal fin changes and the shape variations are shown in figure 3. As shown in these figures, we can get the different shapes of fins with different values of  $c$ , where  $c$  is chosen to mimic the shape of caudal fins of a carp and a shark.



**Fig. 2.** Modeling of caudal fin

**Table 1.** Parameters of equations

Parameters	Value
a	12.5
b	0.07
d	4.5
e	0.09



**Fig. 3.** Cases of various fin shapes, case A:c=0mm(top left), case B:c=10mm(top right), case C:c=25mm(lower center)

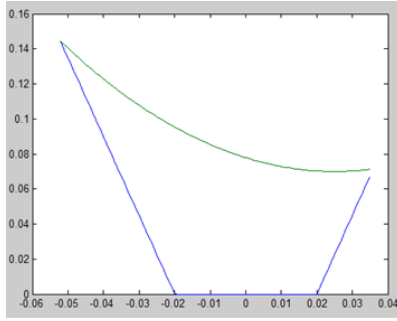


Fig. 3. (continued)

To evaluate the torque and thrust using formula (5)~(7), the area of the fin should be calculated. For this, it is assumed that the whole area of the fin is composed of a small area,  $a_i$  like figure 4.

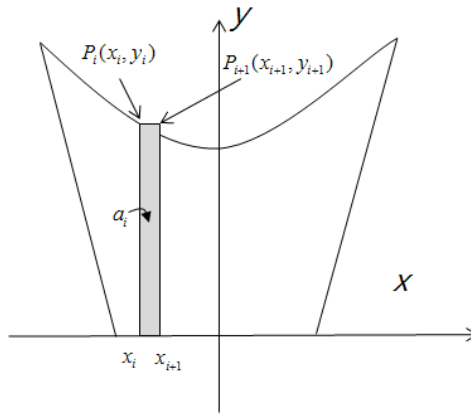


Fig. 4. Calculation for area

From Fig. 4, the area of a caudal fin,  $A$  can be

$$\begin{aligned}
 A &= \sum_{i=1}^N a_i = \sum_{i=1}^N (x_{i+1} - x_i) f_2(x_i) \quad \left(-\frac{m}{2} < x_i < \frac{m}{2}\right), \\
 &\sum_{i=1}^N (x_{i+1} - x_i) (f_2(x_i) - f_{1,3}(x_i)) \quad \left(-\frac{m}{2} > x_i \text{ or } x_i > \frac{m}{2}\right).
 \end{aligned}
 \tag{10}$$

In this paper, the area of the caudal fin is divided into 88 sub areas. Figure 5 shows the divided area for calculation.

Now, we can get the torque and thrust through formula (1) ~ formula (10) for the fins of figure 3. Calculated area of each fin is shown in Table 2.



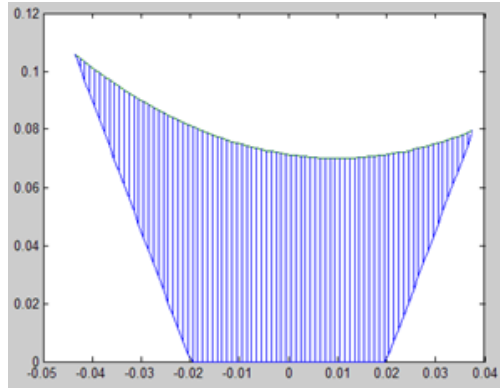


Fig. 5. Example of divided area of case B

Table 2. Calculated area of each fin

Case	Area [m <sup>2</sup> ]
A	0.0044
B	0.0048
C	0.0044

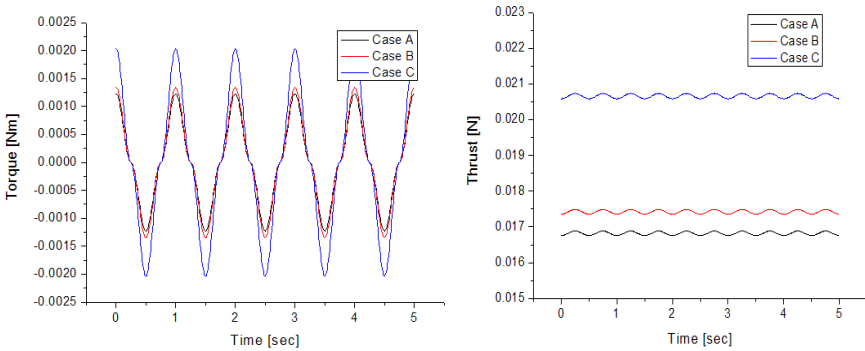


Fig. 6. Analysis results for torque(left) and thrust(right)

Torque and thrust calculated from formula (1) ~ formula (10) are shown in figure 6, and the averaged thrust and maximum of torque are also summarized in Table 3 and 4. From these results, it can be known that Case C shows the biggest torque and thrust among those. But, Table 2 shows that the area of each fin is not same. So, to compare those results more clearly, normalized torque and thrust are calculated by dividing each value by each area. These normalized values are also shown in formula 7 and the values are summarized in Table 5 and 6.

**Table 3.** Average of thrust

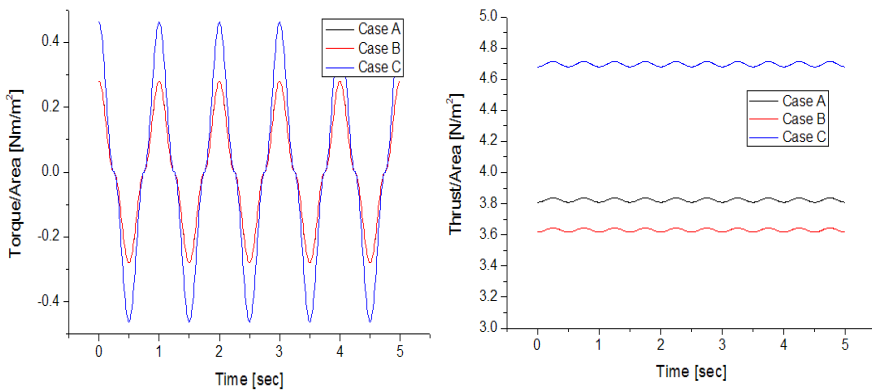
Case	Thrust [N]	Ratio
A	0.0168	1
B	0.0174	1.036
C	0.0207	1.232

**Table 4.** Peak of torque

Case	Torque [Nm]	Ratio
A	0.00123	1
B	0.00134	1.089
C	0.00203	1.650

Figure 7, Table 5 and 6 show that normalized torque and thrust of case C are the biggest among those and case A shows the smallest torque and thrust. One of the most interesting points from the results is that the normalized torque of case A and B is same although the values of normalized thrust peaks are different.

From above analysis, we can calculate for thrust and torque of various shapes by using added mass theory.



**Fig. 7.** Normalized analysis results of torque(left) and thrust(right)

**Table 5.** Normalized average of thrust

Case	Thrust [N/m <sup>2</sup> ]	Ratio
A	3.882	1
B	3.954	1.018
C	4.33	1.115

**Table 6.** Normalized peak of torque

Case	Torque [Nm/m <sup>2</sup> ]	Ratio
A	0.28	1
B	0.28	1
C	0.46	1.64

## 4 Conclusion

In this paper, some analyses for a caudal fin of fish with various shapes have been performed. Using added mass theory, torque and thrust of fish propulsion were calculated and compared. From this analysis, we can conclude with certainty that the shape of a caudal fin influences on the fish propulsion performance. Future study includes that the CFD analysis or real water tank experiments for three kind of caudal fin shown in this paper will be performed and compared.

## References

1. Lauder, G.V., Madden, P.G.A.: Learning from fish: kinematics and experimental hydrodynamics for roboticists. *International Journal of Automation and Computing* 3, 325–335 (2006)
2. Lauder, G.V., Madden, P.G.A.: Fish locomotion: kinematics and hydrodynamics of flexible foillike fins. *Experiments in Fluids* 43, 641–643 (2007)
3. Low, K.: Modelling and parametric study of modular undulating fin rays for fish robots. *Mechanism and Machine Theory* 44, 615–632 (2009)
4. Lindsey, C.: Form, function, and locomotory habits in fish. *Fish Physiology* 7, 1–100 (1978)
5. Sfakiotakis, M., Lane, D.M., Davies, J.B.C.: Review of Fish Swimming Modes for Aquatic Locomotion. *IEEE Journal of Oceanic Engineering* 24(2) (April 1999)
6. Yu, J., et al.: Development of a biomimetic robotic fish and its control algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 34, 1798–1810 (2004)
7. Morgansen, K.A., et al.: Geometric methods for modeling and control of free-swimming finactuated underwater vehicles. *IEEE Transactions on Robotics* 23, 1184–1199 (2007)
8. Chen, W., Xia, D., Liu, J.: Modular Design and Realization of a Torpedo-Shape Robot Fish. In: *Proceedings of 2008 IEEE International Conference on Mechatronics and Automation*, pp. 125–130 (2008)
9. Hiara, K., Kawai, S.: November 26 (2001), <http://www.nmri.go.jp/eng/khirata/fish/experiment/upf>
10. Webb, P.: Is the high cost of body caudal fin undulatory swimming due to increased friction drag or inertial recoil. *Journal of Experimental Biology* 162, 157–166 (1992)

# Development of a 3-DOF Fish Robot 'ICHTHUS V5'

Gi-Hun Yang, Hyunjin Lee, and Youngsun Ryuh

Biomimetic Robot Research Group,  
Korea Institute of Industrial Technology  
1271-18, Sa-3-dong, Sangrok-gu, Ansan,  
426-910, Korea  
{yanggh, hjlee22, ysryuh}@kitech.re.kr

**Abstract.** In recent, there is a rising interest on studying fish-like underwater robots because of real fish's great maneuverability and high energy efficiency. However, researches about the fish-like underwater robots have not been investigated so much and there are still diverse problems in respect of using of the fish robot in the real environment such as in the river for detecting water pollution. For example, the fish robot has a short operating time and cannot move narrow passage such as swimming between aquatic plants. Therefore, this paper mainly describes a development of robotic fish which can be used for water quality monitoring system. The fish robot 'Ichthus V5' has a 3-DOF serial link-mechanism and is developed in KITECH. Furthermore, we propose a dynamic equation of the fish robot to use the underwater environment. We added several sensors to navigate autonomously in the real environment like river. Also, we added two kinds of sensor to detect temperature, electric conductivity, pH (hydrogen ion concentration) of water. Therefore, the developed system can be applied to environmental monitoring system for detect pollution or quality of river.

**Keywords:** robotic fish, dynamic analysis, propulsion, propulsive.

## 1 Introduction

In General, underwater mobile robots have been widely used for the purpose of the marine exploration and resource extraction. In order to perform these goals, the propulsion of the high efficiency and the excellent dynamic property of the robots are needed [1]. However, when it comes to the efficiency and a moving ability, there are no positive advantages for the underwater vehicle that has pre-existing types of propeller to move forward. The driving efficiency of the vehicle does not exceed 70 % and this could lead to a short searching distance of the system. Also, the increasing capacity of the battery could cause its ability to carry the apparatuses decreasing. Instead of conventional rotary propellers used in ships or other underwater vehicles, undulatory movement provides the main energy source for robotic fish [14,15]. An observation of real fish shows that this kind of propulsion is less noisy, more effective, and more maneuverable than propeller-based propulsion [13]. Moreover, the radius of the rotational motion is too big and evading speed is too

slow. For this reasons, it is not appropriate in moving or exploring in the packed area [2]. As a solution for this, there is a study underway and the study is related with the underwater mobile robots which mimic a fish's swimming movement.

First of all, the swimming mechanism of the fish is well known as more than a 20% efficient way rather than the way of propeller proceeding method which is more general one. For instance, MIT research team conducted an experiment about the comparing a performance between the fin which looks like a tuna's tail fin and a propeller in 1994 [9]. And they found that the fins are more efficient method to proceed in the water. The fins of fish could repulse the water more than the square of propeller's blade. And this makes more power to proceed forward. As a result the energy efficiency of the propeller became 70% but the robot tuna's fin recorded 87 % [3]. Several researchers developed various kinds of robot fish. In Japan, Nagoya University developed a micro robotic fish using ICPF actuators [11], while Tokai University produced a robot Blackbass [12] in order to research the propulsion characteristics of pectoral fins. North-western University applied shape memory alloy (SMA) to produce a robot lamprey [10] which aimed to provide mine countermeasures.

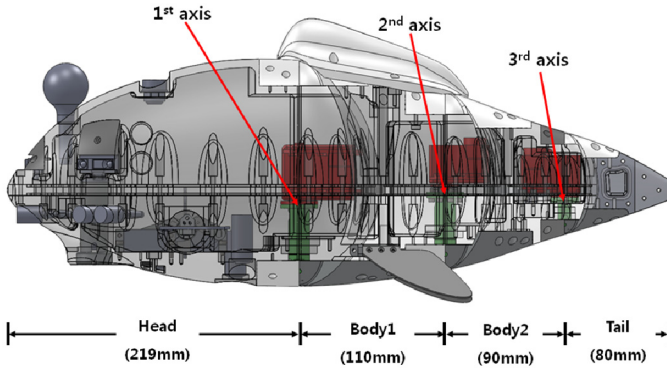
Second, fishes usually have more excellent rapid movement rather than other underwater vehicle. So they can turn their way even though there is only 1/10 length of their body. However, submarines which developed by human need almost 10 times of the returning distance, because it needs a time to slow down and turning their body [4].

In our laboratory, we already suggested and experimented about the new mechanism of a side-fin which is designed to enhance the movement of the underwater fish-like robots[14]. In this paper, the development of robotic fish will be introduced from its design stage to the calculation of the Ichthus's dynamic equation which was executed to make more efficient swimming mechanism of the robot 'Ichthus V5'. The developed system has autonomous navigation ability and the water quality monitoring capability. Sensors and related signal processing systems are integrated to the developed system for achieving its abilities. The simulation of the torque calculation and the swimming velocity has been completed. To make those calculations done, the model for the fluid force was derived.

## **2 Dynamic Equation of Robotic Fish**

### **2.1 Mechanical Link Model**

The target model as shown figure 1 of this paper is 'Ichthus V5' which has 3 Joints in the body and the tail. Also, it swims and moves its tail to the left and right side as like a real fish. On each joint, the servo motors are connected to the body frame of the robot. Therefore, by using the servo motors, we are able to control frequency and amplitude of the robot. Table 1 and Figure 2 show specifications of "Ichthus Ver. 5" and actual implementation of the robotic fish.



**Fig. 1.** Mechanism model of 'Ichthus'

**Table 1.** Specifications of 'Ichthus Ver. 5'

Body size	500mm(L) x 130mm(W) x 170mm(H)
Weight	4.3 kg
DOF	6 (Tail 3, Side Fin 2, mouth and gill 1)
Battery	14.8V Li-Io Battery



**Fig. 2.** Actual Implementation of Robotic Fish 'Ichthus V5'

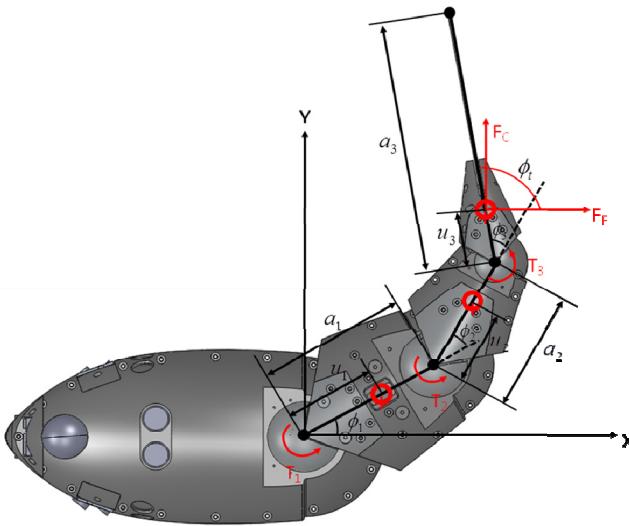
## 2.2 Sensors

The developed system embedded several kinds of sensors for autonomous navigation and quality detection of water. Sensors to be used for autonomous navigation are IR sensors, ultrasound range sensors, GPS sensor, water pressure sensor, INS(Inertial Navigation System), USBL. Basically the developed robotic fish uses GPS sensor for

localization when the robot is on the surface of water. In case that the robotic fish is located in the underwater environment, GPS data is not accessible. Therefore the robot needs to use water pressure sensor for depth detection and INS for localization. To compensate the error of INS, USBL location data is transmitted through an ultrasound modem. With these features, the robotic fish can move autonomously in the underwater environment as well as the surface of water. Since one of main purpose of the robotic fish is to prevent water pollution, the developed robotic fish has three kinds of sensors measuring temperature, EC(electric conductivity), pH(hydrogen ion concentration) of water. These features can be monitored anytime the operator wants to.

### 2.3 'Ichthus' Dynamic Model

To make the simulation simple, we take several reasonable assumptions. The Thrust Force  $F_F$  and Lateral Force  $F_C$  are only applied to the Link 3 that is a tail fin and a yawing motion and a motion along the y-axis direction are fixed as shown in Figure 3 [6].



**Fig. 3.** Analytical Model of 'Ichthus'

With these assumptions, the dynamic equation can be applied to our developed robotic fish, and we can constitute the dynamic model as same as a method of the Serial Link robot which contains a 3-DOF(degrees of freedom) links with applied external forces[14].

Therefore, the torque value which exists on each joint could be obtained by the following equation.

$$T_{\Phi} = [I_{\Phi\Phi}^*] \ddot{\Phi} + \dot{\Phi}^T [P_{\Phi\Phi}^*] \dot{\Phi} - [G_{\Phi}^U]^T T_U. \quad (1)$$

In these equation, the  $I_{\Phi\Phi}^*$  means that the effective inertia, and the  $P_{\Phi\Phi}^*$  represents a changing system inertia.

$$[I_{\Phi\Phi}^*] = \begin{bmatrix} I_{11}^* & I_{12}^* & I_{13}^* \\ I_{21}^* & I_{22}^* & I_{23}^* \\ I_{31}^* & I_{32}^* & I_{33}^* \end{bmatrix}$$

$$\begin{aligned} I_{11}^* &= ([\bar{I}_1 + \bar{M}_1 u_1^2] + [\bar{I}_2 + \bar{M}_2 u_2^2] + [\bar{I}_3 + \bar{M}_3 u_3^2] + \bar{M}_2 l_1^2 + \bar{M}_3 [l_1^2 + l_2^2]) \\ &\quad + (2\bar{M}_2 l_1 u_2 \cos \Phi_2 + 2\bar{M}_3 [l_1 l_2 \cos \Phi_2 + l_2 u_3 \cos \Phi_3 + l_1 u_3 \cos(\Phi_2 + \Phi_3)]) \\ I_{21}^* = I_{12}^* &= ([\bar{I}_2 + \bar{M}_2 u_2^2] + [\bar{I}_3 + \bar{M}_3 u_3^2] + \bar{M}_3 l_2^2) \\ &\quad + (2\bar{M}_2 l_1 u_2 \cos \Phi_2 + \bar{M}_3 [l_1 l_2 \cos \Phi_2 + 2l_2 u_3 \cos \Phi_3 + l_1 u_3 \cos(\Phi_2 + \Phi_3)]) \\ I_{31}^* = I_{13}^* &= ([\bar{I}_3 + \bar{M}_3 u_3^2] + (\bar{M}_3 [l_2 l_3 \cos \Phi_3 + l_1 u_3 \cos(\Phi_2 + \Phi_3)])) \\ I_{22}^* &= ([\bar{I}_2 + \bar{M}_2 u_2^2] + [\bar{I}_3 + \bar{M}_3 u_3^2] + \bar{M}_3 u_2^2) + (2\bar{M}_3 l_2 u_3 \cos \Phi_3) \\ I_{32}^* = I_{23}^* &= ([\bar{I}_3 + \bar{M}_3 u_3^2] + (\bar{M}_3 l_2 u_3 \cos \Phi_3)) \\ I_{33}^* &= ([\bar{I}_3 + \bar{M}_3 u_3^2]) \end{aligned}$$

$[P_{\Phi\Phi}^*]$  is a matrix of three-dimensional array and represents as follows.

$$[P_{n\Phi\Phi}^*] = \begin{bmatrix} P_{n11}^* & P_{n12}^* & P_{n13}^* \\ P_{n21}^* & P_{n22}^* & P_{n23}^* \\ P_{n31}^* & P_{n32}^* & P_{n33}^* \end{bmatrix} = n^{th} \text{ plane of } [P_{\Phi\Phi}^*]$$

$$[P_{1\Phi\Phi}^*] = \begin{bmatrix} P_{111}^* & P_{112}^* & P_{113}^* \\ P_{121}^* & P_{122}^* & P_{123}^* \\ P_{131}^* & P_{132}^* & P_{133}^* \end{bmatrix} = \begin{bmatrix} 0 & A & B \\ A & A & B \\ B & B & B \end{bmatrix}$$

$$[P_{2\Phi\Phi}^*] = \begin{bmatrix} P_{211}^* & P_{212}^* & P_{213}^* \\ P_{221}^* & P_{222}^* & P_{223}^* \\ P_{231}^* & P_{232}^* & P_{233}^* \end{bmatrix} = \begin{bmatrix} -A & 0 & C \\ 0 & 0 & C \\ 0 & C & C \end{bmatrix}$$

$$[P_{3\Phi\Phi}^*] = \begin{bmatrix} P_{311}^* & P_{312}^* & P_{313}^* \\ P_{321}^* & P_{322}^* & P_{323}^* \\ P_{331}^* & P_{332}^* & P_{333}^* \end{bmatrix} = \begin{bmatrix} -B & -C & 0 \\ -C & -C & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$A = -\bar{M}_2 l_1 u_2 \sin \Phi_2 - \bar{M}_3 (l_1 l_2 \sin \Phi_2 + l_1 u_3 \sin(\Phi_2 + \Phi_3))$$

$$B = -\bar{M}_3 (l_2 u_3 \sin \Phi_3 + l_1 l_3 \sin(\Phi_2 + \Phi_3))$$

$$C = -\bar{M}_3 l_2 u_3 \sin \Phi_3$$



The distance between the center of mass, the inertial moment of each link, and a mass were calculated from the 3D CAD (SolidWorks<sup>TM</sup>) model.

In the equation (1),  $[G_{\Phi}^U]$  means the Jacobian matrix which represents the correlation between the angular velocity and the velocity of the tail fin. So we can transfer the Thrust Force  $F_F$  and Lateral Force  $F_C$  which are applied to the fluid into the torque of each joint.

### 2.4 Model of Fluid Force

In the modeling of fluid force, a thrust force is mainly generated from the tail fin. Therefore we could suppose that the drag force by the fluid could be applied to the body of the robotic fish.

In figure 4, the  $F_V$  means an inertia which generated by the movement of a tail fin. The magnitude is proportioned with the angular velocity's value and has opposite direction.

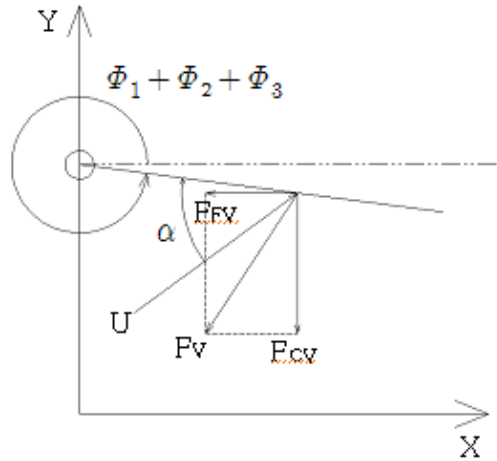


Fig. 4. Model of inertial fluid force

$$F_V = -\pi\rho LC^2 (U \sin \alpha + \alpha U \cos \alpha). \tag{2}$$

Where,  $U$  means the relative velocity generated from the tail fin,  $C$  represents the chord length of tail fin, and the  $L$  means span. The density of water ( $\rho$ ) is  $998 \text{ kg/m}^3$ . The floating force generated by the  $U$  on the occasion of rotation in the tail fin is only applied as the way of vertical direction. Thus, the real force,  $F_J$  could be expressed as the equation (3).

$$F_J = 2\pi\rho LCU^2 \sin \alpha \cos \alpha. \tag{3}$$

Therefore, the thrust force element  $F_F$  and the lateral force element  $F_C$  could be expressed as (4) and (5).

$$F_F = F_{FV} + F_{FJ} \tag{4}$$

$$F_C = F_{CV} + F_{CJ} \tag{5}$$

In the mean time, we assumed that the drag force  $F_D$  was only applied to the main body.

$$F_D = \frac{1}{2} C_D \rho V^2 S \tag{6}$$

In the above equation, the  $C_D$  represents drag coefficient, the  $V$  means comparative trust velocity and the  $S$  means an area which are directly projected along with the flow of fluid, the  $C_D$  was assumed as 0.5.

Also if we confine the proceeding direction of the robot as the x-axis direction, the relative velocity along the y-axis direction in the middle of tail fin could be derived as follows.

$$u = l_1 \cos \Phi_1 \dot{\Phi}_1 + l_2 \cos(\Phi_1 + \Phi_2)(\dot{\Phi}_1 + \dot{\Phi}_2) + a_3 \cos(\Phi_1 + \Phi_2 + \Phi_3)(\dot{\Phi}_1 + \dot{\Phi}_2 + \dot{\Phi}_3) \tag{7}$$

Because the  $U_m$  which represents the velocity of a moving fluid along x-axis direction and the  $u$  are directly crossing, So  $U$  could be represented as follows.

$$U^2 = U_m^2 + u^2 \tag{8}$$

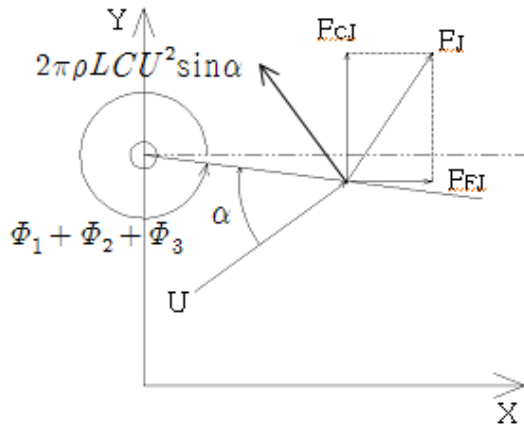


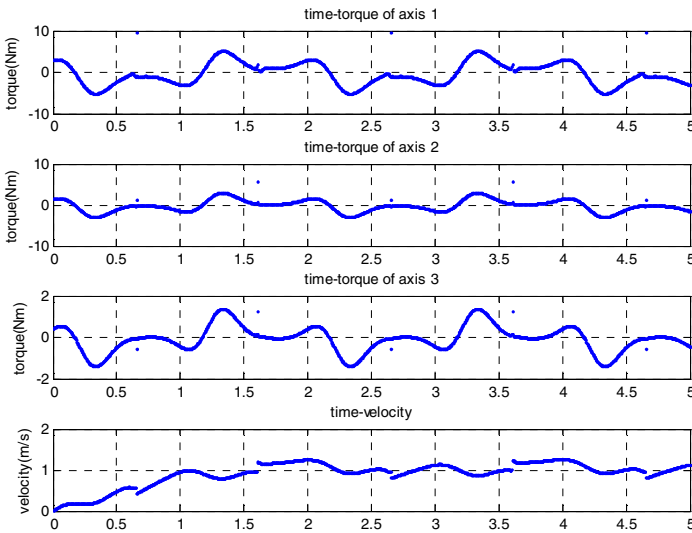
Fig. 5. Model of lift force

## 2.5 Simulation Design

The simulation was conducted through the Matlab<sup>TM</sup> and the acceleration of the robot was calculated from the equation (9), and the velocity of robot was derived by the equation of Runge-Kutta.

$$M\ddot{x}_G = F_F - F_D. \quad (9)$$

At this calculation,  $M$  means the total mass of the robotic fish,  $x_G$  means position of the mass center,  $F_F$  means the total thrust force, and  $F_D$  means the total drag force. In the simulation, the amplitude and the period are the input value, therefore, the swimming velocity along x-axis direction could be calculated. The figure 6 shows the simulation results.



**Fig. 6.** Results of the simulation. In order joint 1 torque, joint 2 torque, joint 3 torque, velocity(Input : 0.8 Hz, Amplitude 20 deg Um 0.03m/s)

## 3 Conclusion

In this paper, we introduced our research work on the design and construction of robotic fish 'ICHTHUS V5'. The developed robotic fish has embedded many kinds of sensors to navigate autonomously in the real environment like river. Also, three kinds of sensors are integrated to measure temperature, electric conductivity and pH of water for monitoring the quality of water. We were derived the mathematical model of the fish robot which swims using tail fin of 3-DOF link mechanism. Since we conducted parameter optimization with our previous research and confirmed that the optimized input value compared to the existing value that can produce faster velocity

by using smaller torque value, we can apply the input parameters which include the minimum RMS Torque value. These parameters were found to increase energy efficiency. We can improve the energy efficiency and operating time of the robotic fish by using this research result. Also, the developed robotic fish is able to avoid unknown obstacles in the real environment through those experimental results. In the future, the fish robot should be improved its performance. For example, it will be achieved high speed swimming like human. We validate the simulation and will complement by constructing the dynamical experimental environment. And, in many case of occurring in real-world environment, we would study how to optimize energy and prove through an experiment of field test of the robot.

**Acknowledgements.** This research was supported by Korea Research Council for Industrial Science and Technology, research grant B551179-10-02-00.

## References

1. Hirata, K., Takimoto, T., Tamura, K.: Study on turning performance of a fish robot. In: Proc. 1st Int. Symp. Aqua Bio-Mech., pp. 287–287 (2000)
2. Cho, K., Park, H., Kim, S.-W., Yang, H., Park, Y.-P.: Development of robot mimicking propulsion of a fish. In: Korean KSME, pp. 40–45 (2007)
3. Kim, Y.-H.: Robotic fish, the prince of ocean. Dong-A Science(Magazine in Korean) 8, 54–59 (2005)
4. Ryuh, Y.-S.: Development of swimming mechanism and algorithm of fish-like underwater robot. In: KROS, pp. 43–48 (2009)
5. Chung, C., Lee, S.-H., Kim, K.-S., Cha, Y.-S., Ryuh, Y.-S.: Design of side fin mechanism of fish-like underwater robot. In: ICROS, pp. 582–585 (2009)
6. Nakashima, M., Ohgishi, N., Ono, K.: A study on the propulsive mechanism of a double jointed fish robot utilizing self-excitation control. JSME International Journal, Series C 46(3), 982–990 (2003)
7. Chung, G.B., Yi, B.-J., Lim, D.J.: An efficient dynamic modeling methodology for general type of hybrid robotic systems. In: Proc. of the 2004 IEEE International Conference on Robotics & Automation (April 2004)
8. Lee, S.-H.: Data analysis of engineering statistics using minitab. EreTech (2008)
9. Streitlien, K., Triantafyllou, G.S., Triantafyllou, M.S.: Efficient foil propulsion through vortex control. AIAA Journal 34, 2315–2319 (1996)
10. (2004), <http://www.dac.neu.edu/msc/burp.html>
11. Guo, S., Fukuda, T., Kato, N., Oguro, K.: Development of underwater microrobot using ICPF actuator. In: Proceedings of the 1998 IEEE International Conference on Robotics & Automation, pp. 1829–1834 (1998)
12. Kato, N.: Control performance in the horizontal plane of a fish robot with mechanical pectoral fins. IEEE Journal of Oceanic Engineering 25(1), 121–129 (2000)
13. Hu, H.: Biologically Inspired Design of Autonomous Robotic Fish at Essex. In: Proceedings of the IEEE SMC UK-RI Chapter Conference 2006 on Advanced Cybernetics Systems, pp. 1–8 (2006)
14. Yang, G., Choi, W., Lee, S., Kim, K., Lee, H., Choi, H., Ryuh, Y.: Control and Design of a 3 DOF Fish Robot ICHTHUS. In: Proceedings of 2011 IEEE International Conference on Robotics and Biomimetics, pp. 2108–2113 (2011)

15. Yamamoto, I., Terada, Y.: Robotic fish and its technology. In: Proceedings of SICE Annual Conference in Fukui, Fukui University, Japan, August 4-6, pp. 73-76 (2003)
16. Tapia, E.M., Intille, S.S., Larson, K.: Activity Recognition in the Home Using Simple and Ubiquitous Sensors. In: Ferscha, A., Mattern, F. (eds.) PERVASIVE 2004. LNCS, vol. 3001, pp. 158-175. Springer, Heidelberg (2004)
17. Yamamoto, I.: Research on an oscillating fin propulsion control system. In: Proc.IEEE OCEANS 1993, vol. 3, pp. 259-263 (1993)

# Polar Histogram Based Sampling Method for Autonomous Vehicle Motion Planning

Dmitriy Ogay, Jee-Hwan Ryu, and Eun-Gyung Kim

Korea University of Technology and Education, Cheonan, South Korea  
{cactus, jhryu, egkim}@koreatech.ac.kr

**Abstract.** In this paper we present a sampling based motion planning algorithm for an autonomous vehicle, which allowed our vehicle to navigate smoothly at high speed with limited computation resources. A new sampling method, limiting candidate states, is introduced to reduce computation burden, associated with sampling based motion planning algorithms. The proposed method is experimentally evaluated driving an autonomous vehicle at speeds up to 60 km/h. It showed how advantages of both sampling based and space discretization based planning algorithms can be combined in one method, providing short planning time in higher dimensional configuration space and good performance in narrow and cluttered environment.

## 1 Introduction

Recently there have been many research activities in the area of autonomous vehicles. Being integrated complex systems, autonomous vehicles consist of many functional subsystems, and their development brings challenges in different fields of research. One of the problems is safe and reliable planning of vehicles motion. Development of our planning algorithm was motivated by Hyundai-Kia Autonomous Vehicle Competition.

Even though much work has been done in the field of obstacle avoidance and path planning, simple obstacle avoidance algorithms such as potential fields [2] and vector field histogram (VFH) could not be applied for navigation of autonomous vehicles because of their local nature, local minima problem, and inability to take dynamic constraints of autonomous vehicle into account. There were also many path-planning algorithms that work in a discrete configuration space [7], but, because of continuous nature of car motion, and non-holonomic constraints, generated paths are not satisfactory for use [1].

There were some hybrid (discrete-continuous) approaches, some of which were successfully implemented for autonomous vehicles [1]. But this method needs considerable high resolution for certain environments such as narrow passage. The specificity of the competition mission was passing in narrow passages. Obstacles were not complicated enough to require backward motion, and the car speed in narrow lanes was important. If we have used discrete approaches, then we would have to have considerably high resolution of space discretization, which in turn would have led to high computation burden and high latencies in the planner. Another problem with such algorithms is that in higher-dimensional hyperspaces their complexity grows exponentially with added dimensions.

The other type of path-planning algorithms is continuous space planning algorithms, among which Rapidly Exploring Random Trees (RRT)[\[6\]](#) based ones are well known, and studied a lot. One of the weaknesses of RRT algorithm is its performance in a cluttered environment and narrow passages.

We have selected a sampling based approach, and have developed a method to limit sampling area for candidate states, to achieve a reasonable computation time. This method is based on our a priori knowledge, that in most cases a vehicle would have to move in the direction of a given reference path. Our algorithm is a search tree, gradually growing in the direction of a reference path. Sampling is made in such a way that states belonging to the propagating front can be clustered in every step. We use the centers of these clusters to represent the higher probability state, where vehicle would have a better visibility of further environment. And then use a polar histogram to configure a probability distribution function for the sampling at the next step.

Sampling biasing for RRT like algorithms is not a new approach and the use of visibility information is neither. However, to the best of our knowledge, there is no such approach that combines sampling and space discretization approaches using clustering of states, belonging to the propagating front. The main advantage is that we can maintain good performance of sampling based methods in higher dimensional hyperspaces, at the same time providing fast navigation in narrow passages and cluttered environment, as with space discretization based methods.

## 2 Motion Planning Overview and Method Description

### 2.1 Motion Planning Overview

Most of up-to-date motion planning algorithms can be divided into two groups by the type of expansion. The first group is where nodes to be expanded are first selected from the search tree, and then expanded. Expansion is done by integrating some control input  $u(t)$  for a period of time from the selected state in a tree. Priority queue is used in the most cases. The second group is where a new candidate state is first sampled in a configuration space, and then based on that information, an existing node is selected, then  $u(t)$  is found, and the new state is added to the search tree. In this case nodes to be expanded are chosen based on the nearest neighbor search, like in RRT.

These two approaches have different behavior in road-like environment. If we represent a road as a space bounded by walls from the both sides, then the first type of algorithms would expand faster in forward direction because space is limited. But sampling based algorithms would have many collision check failures, which would lead to slow expansion.

But the first type approach, from the other side, has the following disadvantage: due to the large range of road curvature change, discretization of the control input  $u(t)$  should be high enough to fit the shape of the road curve.

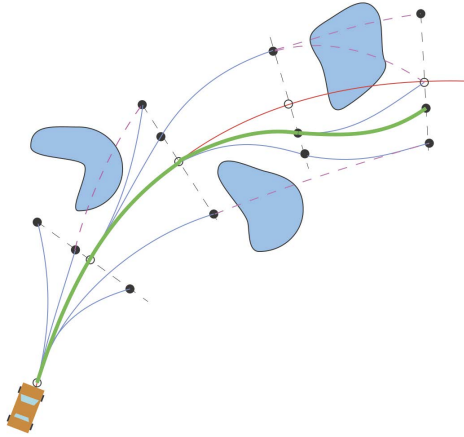
### 2.2 Target Environment and Conditions

For the competition, we participated in, the main task was to follow a reference path, and avoid obstacles if they exist. Reference path was provided. Obstacles were not

complicated enough to require backward motion planning or intricate manoeuvres, and there is no such obstacle that requires driving in the direction opposite to the reference path.

These conditions allowed us to make simplifications to our algorithm. Our algorithm works a growing tree, with a propagating front. At every cycle we process a propagating front. Taking into account above-mentioned conditions we assume that normals to the reference path would be good candidates for a propagated front, as it is seen on Fig. 1. The completeness of the planner will be discussed later.

In general our method is not limited to such of representation of a wave front. But for the simplicity of implementation we let it be so by now.



**Fig. 1.** Search tree growth is performed in steps, target points are sampled at normals of the reference path at reference way points

### 2.3 Planning Algorithm

By the expansion approach our planning method is close to a well-known RRT algorithm. Initial and goal states in the configuration space  $\mathcal{C}$  given as an input. A search tree is grown from the initial state.

Unlike in RRT algorithm where candidate states are sampled from a configuration space in every cycle, we make a tree expansion in steps to provide more deterministic behavior, so it is better suited for real-time planning. At every step we propagate the expansion front of our tree by some distance. As we mentioned before, for the reasons of simplicity we have selected normals to the reference path to be candidates of a tree front. So at every cycle we make sampling from the normal taken at the next reference point.

After a new candidate is sampled, the nearest neighbor state from the tree is chosen, a trajectory linking two states is calculated, and a collision check is performed. If collision



check passes successfully then the new state is connected to the tree. At every cycle, for every propagation front we perform a number of sampling, enough to provide smooth motion. And also we require enough samples to give a good representation of an actual propagated front.

It is necessary to distinguish a candidate propagated front, which is determined heuristically, and an actual propagated front, which depends on an actual environment and is represented by a set of states.

### 3 Sampling Method

#### 3.1 Issue with Sampling

In previous section a simple sampling strategy, sufficient for motion planning of a vehicle, was described. However such a simple strategy may sometimes show a poor performance because. Tree expansion is computationally expensive; it includes two operations, which is nearest neighbor search, and collision check.

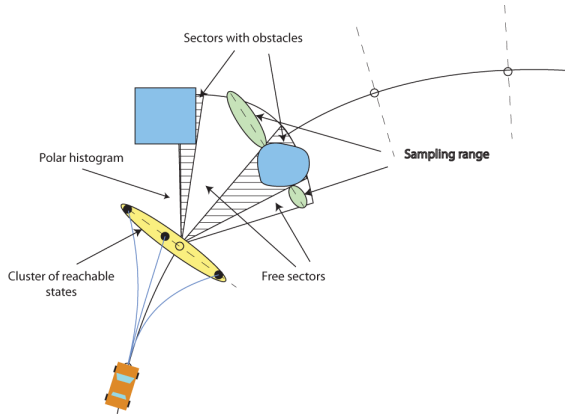
Problem with the nearest neighbor search is that for planning with differential constraints, Euclidean distance in configuration space is not enough, and algorithms which provide  $\mathcal{O}(n \log n)$  look up doesn't work. So in real environment we have to deal with  $\mathcal{O}(n^2)$  lookups. Collision check is a costly operation in general.

We have investigated the candidate samples, which cause most of the collision failures in an original algorithm, and detected a road pattern where algorithm performed poorly. This was a path bounded by walls from both side, like on Fig. 3. The problem is that naive sampling from the normals may pick candidate states from the area behind the wall. And the narrower the road, the more probably such a failure state would be sampled.

#### 3.2 Polar Histogram Based Sampling

To solve the problem of inefficient sampling our task was to find a probability distribution function that would give better failure rate. As we mentioned before, we distinguish candidate propagation front and actual propagation front. The latter one is represented by a set of states, connected to the tree. At this step we find clusters of those states. This clustering is performed to determine discrete obstacles on the road, i.e. obstacles are discontinuities in an actually propagated front. By making clustering we find continuous segments of a front, and then choose the center of a cluster to be a good point for visibility lookup. We can justify this approach because if we were located between two obstacles, then the widest view angle would be from the point in the middle between two obstacles.

After the viewpoint is determined we build a polar histogram to determine a probability distribution function for sampling. So the output of a histogram is an approximation of  $P\{\text{successful motion} | \text{current position} = \text{view point}\}$ . One of the good approximation is an intersection of histogram sectors with candidate propagation front, as shown in Fig. 2. But this approach is too limiting. So the edges of intersection segments can be expanded by some margin.



**Fig. 2.** Clustering of actually propagated front states, and a polar histogram, built from the center of a cluster

Such an approximation of probability of successful motion, is justified by the fact that at a high speed vehicle is more probable to move forward in the direction of road, and if such a motion is not feasible, then the speed of a vehicle can be reduced, and thus some more time is reserved to investigate less probably chosen directions.

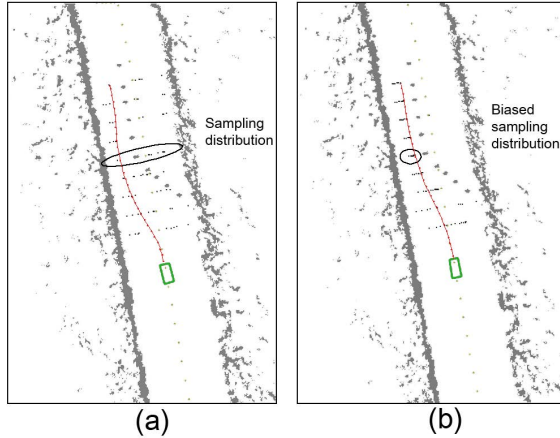
### 3.3 Generalization and Completeness

It may be noticed that in such an implementation, the following assumption was made. Since road width is not large, propagated front states clusters are also not wide in comparison to the distance to the next candidate front. To generalize the method for wider applications we can define a maximum size of a cluster, and divide large clusters into small subclusters, and perform histogram based lookup from the centers of the latter.

The second issue that can be noted is that points are not sampled in between the reference points. Such an algorithm may work if required maneuvers are larger than distance between propagated fronts, but this problem is solved incrementally in a following way. If propagation fails, then a reference point is chosen in the middle between the current reference point and the next failed reference point, and sampling is made on the normal to it. Correspondingly, maximum width of a cluster should also be reduced by two. Following these steps iteratively we can cover the whole space with desired density.

In our implementation candidate propagation fronts are represented by normal to the reference path, but in general they can be constructed using heuristic function, or be constructed interactively online, from the clustering information.

Our method works because two-dimensional workspace can be a good approximation of a projection of higher dimensional configuration space. This gives us opportunity to perform preliminary approximate lookups using fast algorithms like vector field histogram in two-dimensional space. The innovation of our method is that we combine sampling based approach with wave front propagation approach using clustering in a reduced space.



**Fig. 3.** Comparison of sampling distribution (black dots) of the algorithm with uniform sampling (a), and polar histogram based sampling (b). Passing artificially made narrow path.

## 4 Experiment

We implemented the proposed algorithm and tested it on our Hyundai SantaFe SUV based autonomous vehicle, driving in real environment. We have experimentally proved the efficiency of our algorithm. Seen on Fig. 5 is a comparison of a naive sampling and sampling based on our method. As it is seen sampled candidate states (black dots), are not located beyond the reachability of a vehicle. We provide a comparison graph of algorithm execution time over an experimental track, seen on Fig. 3.

### 4.1 Comparison to Existing Approaches

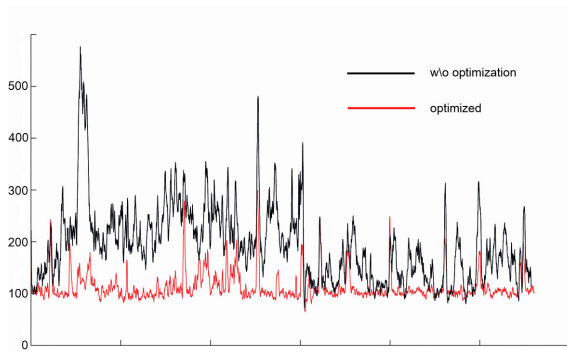
To estimate efficiency of our algorithm to the existing ones, we made a comparative analysis. In comparison to original simple RRT with uniform distribution in configuration space our algorithm performs better because we use some a priori known information about environment, and limited the sampling range. We also used the fact that vehicle travels faster in longitudinal direction than in lateral, especially at high speed. The strongest factor in favor of our algorithm is that it performs better in narrow passages.

In comparison to space discretization methods our algorithm performs better, because sampling defines the discretization of the space and by increasing sampling density in lateral direction of car motion, and decreasing in longitudinal direction we avoid large amount of discretized cells. Another factor in favor of our algorithm is that obstacles are present in robot workspace, and we make histogram lookups in a workspace, while sampling is performed in higher dimensional hyperspace, where sampling based algorithms perform better.

There is also another class of path following algorithms that beat our algorithm in terms of computation time, for example spatiotemporal state lattices [8] approach, but



**Fig. 4.** Hyundai SantaFe SUV based autonomous vehicle



**Fig. 5.** Algorithm execution time(ms). Depicted by black is algorithm with nave sampling, by red is the proposed method.

such algorithms have a pre-processing stage, and the reference path should be known a priori. In case of our algorithm, the reference path can be changed dynamically for example from more abstract higher level planner.

## 5 Conclusion and Further Work

In this paper we have presented a new approach for sampling based motion planning, which allows us to drive our autonomous vehicle at high speed with low computation power requirements.

The method provides a sampling probability distribution function for more efficient search tree expansion by utilizing some assumptions and approximations about car motion. We experimentally prove the efficiency of proposed method by comparing a planner with nave sampling and the polar histogram based sampling. Comparison study over existing approaches is provided as well.

Our planning algorithm was tested and proved well performing in real driving environment. The peculiarity of real environment is that it is cluttered and consists mostly of narrow passages, from the other side planning with differential constraints requires higher order configuration space.

The proposed method is a synergy of RRT-like sampling based approach, which performs well in high dimensional hyperspaces and has a very flexible space discretization, and Dijkstra-like approaches based on wave front propagation, which perform well in narrow passages. The innovation of our method is that we combine sampling based approach with wave front propagation approach using clustering in a reduced space.

Presented motion planner was used by our team during Hyundai-Kia Autonomous Vehicle Competition 2010. It let our vehicle successfully pass all planning related missions.

**Acknowledgement.** The Authors are grateful of the following support: Ministry of Knowledge and Economy of Korea grant as part of the Development of a Tele-Service Engine and Tele-Robot Systems with Multi-Lateral Feedback project.

## References

1. Dolgov, D., Thrun, S., Montemerlo, M., Diebel, J.: Practical Search Techniques in Path Planning for Autonomous Driving? In: Proceedings of the First International Symposium on Search Techniques in Artificial Intelligence and Robotics (STAIR 2008) (June 2008)
2. Duda, R., Hart, P., Stork, D.: Pattern classification, 2nd edn. Wiley, New York (2001) ISBN 0-471-05669-3
3. Frazzoli, E., Dahleh, M., Feron, E.: Real-time motion planning for agile autonomous vehicles. *AIAA Journal of Guidance and Control* 25(1), 116–129 (2002)
4. Konolige, K.: A gradient method for realtime robot control. In: Proceedings of the IEEE International Conference on Intelligent Robots and Systems, IROS (2000)
5. Kuwata, Y., Fiore, G., Teo, J., Frazzoli, E., How, J.: Motion planning for urban driving using RRT. In: Proc. IROS, pp. 1681–1686 (2008)
6. LaValle, S., Kuffner, J.: Randomized kinodynamic planning. *International Journal of Robotics Research* 20(5), 378–400 (2001)
7. LaValle, S.: *Planning Algorithms*. Cambridge University Press, Cambridge (2006), <http://planning.cs.uiuc.edu/>
8. Ziegler, J., Stiller, C.: Spatiotemporal state lattices for fast trajectory planning in dynamic on-road driving scenarios. In: Proc. IROS, pp. 1879–1884 (2009)

# Gaze Control-Based Navigation Architecture for Humanoid Robots in a Dynamic Environment

Jeong-Ki Yoo and Jong-Hwan Kim

Department of Electrical Engineering, KAIST,  
335 Gwahangno, Yuseong-gu, Daejeon 305-701,  
Republic of Korea  
{jkyoo, johkim}@rit.kaist.ac.kr

**Abstract.** Due to the limited information from the environment using a local vision sensor, gaze control research is very important for humanoid robots. In addition, multiple objectives for navigation have interactive relationships among them. From this point of view, this paper proposes a gaze control-based navigation architecture using fuzzy integral and fuzzy measure for humanoid robots. Four criteria are employed along with their partial evaluation functions in order to determine the final gaze direction. By employing fuzzy integral approach for the global evaluation for candidate gaze directions, effective gaze control considering the interactive phenomena among criteria is accomplished and verified through a simulation using a developed simulator for HanSaRam-IX (HSR-IX).

**Keywords:** gaze control, Choquet fuzzy integral, preference-based selection algorithm, univector field method.

## 1 Introduction

Researches for humanoid robots have been started from walking pattern generation researches [1,2]. Based on the improvement of such researches, the extended research issues such as navigation, human-robot interaction, active vision-based 3D object localization, etc. have been performed [3-5]. In particular, gaze control issues were emerged to be important for the robust navigation due to the limited capacity of visual perception of a humanoid robot [6]. In this manner, information-centered approach for gaze control was proposed [7,8]. Even though these various issues for navigation are closely dependent each other, the integration aspect has not been fully considered so far. Based on the importance of gathering surrounding information, this paper proposes a gaze control-based navigation architecture for humanoid robots dealing with navigation in a dynamic environment. Four criteria are employed to consider the local map confidence area, self-localization uncertainty, obstacle-centered surrounding situation, and the capacity to gaze at the waypoint direction along with their corresponding partial evaluation functions and representative gaze directions. As mentioned above, user's preference and interactive relationships among criteria are considered by employing fuzzy measure and fuzzy integral [9]. The effectiveness of the proposed architecture is verified through a computer simulation using the developed simulator for the small-sized humanoid robot, HSR-IX. This paper is organized as follows. Section 2 presents

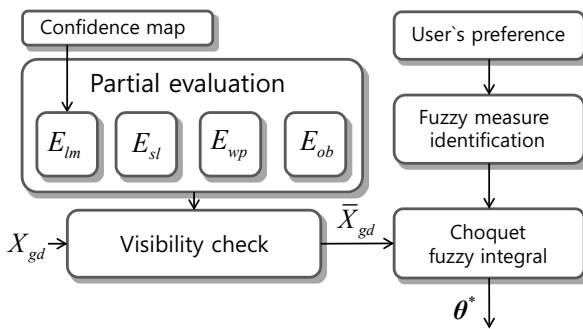
the proposed gaze control-based navigation architecture, four criteria for gaze control and their corresponding partial evaluation functions. Simulation environment and results are presented in Section 3 and concluding remarks follow in Section 4.

## 2 Gaze Control-Based Navigation Architecture

This section describes the gaze control-based navigation architecture for humanoid robots employing fuzzy measure and fuzzy integral for a preference-based gaze selection algorithm incorporated with the navigation architecture.

### 2.1 Proposed Gaze Control Architecture

In order to determine the most appropriate gaze direction considering user’s preference and the interactive phenomena among criteria, various types of information such as distance and velocity of surrounding obstacles, the localization errors of surrounding obstacles and the robot itself, etc. have to be considered simultaneously. For those purpose, the gaze control-based navigation architecture, as shown in Fig. 1 is proposed for humanoid robots with four criteria considering above information and their corresponding partial evaluation functions,  $E_\alpha, \alpha = lm, sl, wp, ob$ . Considering a set of candidate gaze directions,  $X_{gd}$ , defined by the capacity of local vision system, the partial evaluation values over criteria are assigned to each visibility-checked candidate gaze direction in  $\bar{X}_{gd}$  if the corresponding representative gaze directions are visible through the visibility check process. The user-defined preference is represented by the fuzzy measures [10]. The global evaluation of  $\bar{\theta}_i^{pt}$  in  $\bar{X}_{gd}$  is calculated using the Choquet fuzzy integral of partial evaluation values corresponding to visible representative gaze directions for  $\bar{\theta}_i^{pt}$  with respect to their fuzzy measures. Then, the candidate gaze direction having the maximum global evaluation value is selected as the final gaze direction  $\theta^*$ .



**Fig. 1.** Gaze control-based navigation architecture.  $X_{gd}$  is a candidate gaze direction set, and  $\theta^*$  is the final gaze direction.

## 2.2 Criteria for Gaze Control

The following describes the four criteria and their corresponding partial evaluation functions along with their representative gaze directions in detail.

**Local Map Confidence Area-Based Criterion ( $C_{lm}$ ).** This criterion drives gaze selection tendency to un-explored or out-of-date explored directions by managing once explored gaze directions using the confidence values assigned to each grid of the local map [11-13]. Thus, the corresponding partial evaluation function estimates the necessity to gaze at unconfident area according to their confidence-based evaluations. The representative gaze direction for this criteria is generated to gaze at the mean direction of unconfident sample points.

Using the sampled points of robot-centered occupancy grid map, the partial evaluation for  $C_{lm}$  is defined as the ratio of the number of unconfident sample points  $N_{up}$  to the number of sample points  $N_{sp}$  excluding the current gaze area as follows:

$$E_{lm} = n(\{\mathbf{p}_i^s | K(\mathbf{p}_i^s) < \lambda_{cf}, \mathbf{p}_i^s \in R_{s-g}\}) / n(\{\mathbf{p}_i^s | \mathbf{p}_i^s \in R_{s-g}\}) \quad (1)$$

where  $\lambda_{cf}$  is a user-defined confidence threshold for counting unconfident sample points,  $\theta_c^{pt}$  is the current gaze direction,  $R_{s-g}$  is the sampling area excluding the current gaze area  $R_g^c$ ,  $\mathbf{p}_i^s$  denotes the  $i$ th sample point in  $R_{s-g}$ , the function  $K(\cdot)$  returns the confidence value of the corresponding sample point, and the function  $n(A)$  counts the number of elements in a set  $A$ .

**Waypoint-Based Criterion ( $C_{wp}$ ).** The gaze control should be performed to keep the tendency of maintaining its gaze direction toward the destination direction. Thus, the representative gaze direction for this criterion is defined as the direction of waypoint. In this manner, the partial evaluation for this criterion is defined to evaluate the capacity to gaze at the waypoint direction. Using the obstacle-free distance  $d_{of}$  which is the maximum obstacle-free distance from the robot, the partial evaluation function for  $C_{wp}$  is defined as follows:

$$E_{wp}(d_{of}) = \begin{cases} d_{of}/d_{of}^{max} & \text{if } d_{of} \leq d_{of}^{max} \\ 1 & \text{otherwise} \end{cases} \quad (2)$$

**Self-localization-Based Criterion ( $C_{sl}$ ).** For the localization of the robot and the detected obstacles, this paper uses the Unscented Kalman filter-based SLAM, and provides the error covariance matrix  $Q_r$  of the robot position [14]. The magnitude of  $Q_r$ ,  $\sigma_r$  is used to represent the localization uncertainty of the robot. To reduce the localization error, the robot has to gaze at the surrounding static obstacles as often as possible to update their positions. In this manner, the representative gaze directions for this criterion are defined as the directions of surrounding static obstacles. The partial evaluation function is defined as follows:

$$E_{sl}(\sigma_r) = \begin{cases} \sigma_r/\tau_{sl} & \text{if } \sigma_r \leq \tau_{sl} \\ 1 & \text{otherwise} \end{cases} \quad (3)$$

where  $\tau_{sl}$  is a user-defined threshold for the maximum magnitude of  $\sigma_r$ .



**Obstacle-Based Criterion ( $C_{ob}$ ).** The movement of surrounding obstacles has to be considered with the top priority for the dynamic obstacle avoidance problem [6]. Since the objective of this criterion focuses on the obstacle avoidance in a dynamic environment, this criterion uses representative gaze directions gazing at surrounding obstacles including dynamic ones. To evaluate the necessity to gaze at the surrounding obstacles, the partial evaluation function for this criterion is defined as

$$E_{ob}(\sigma_{o_i}) = \begin{cases} \alpha_{o_i}/T_o & \text{if } \alpha_{o_i} \leq T_o \\ 1 & \text{otherwise} \end{cases} \quad (4)$$

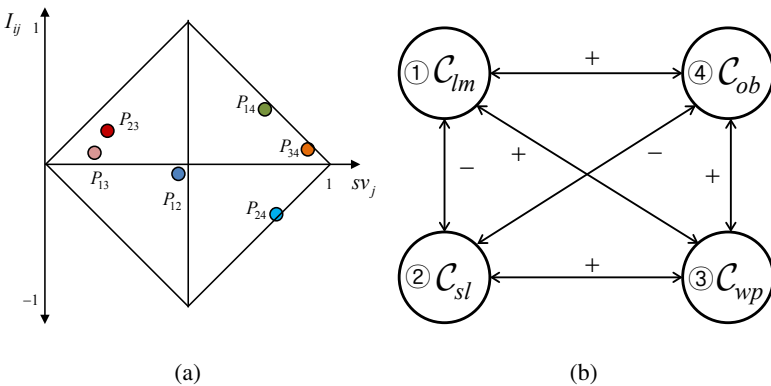
with

$$\alpha_{o_i} = \sigma_{o_i} \omega_{ob} (\omega_{ro} {}^R\hat{V}_{o_i} + \omega_{vo}) \|\mathbf{v}_{o_i}\|_2 / 2 \|{}^R\mathbf{p}_{o_i}\|_2,$$

where  $\alpha_{o_i}$  denotes the arranged terms related to the velocity and position of the  $i$ th obstacle,  $\mathbf{v}_{o_i}$  means the absolute velocity of the  $i$ th obstacle,  $\omega_{ob}$  is a weight adjusting the applicable range of this partial evaluation function, and  $\omega_{ro}$  and  $\omega_{vo}$  are user-defined weights managing relative ratio between two velocity-related terms,  $\xi_{o_i}$  and  $\|\mathbf{v}_{o_i}\|_2$ .

### 2.3 Final Gaze Direction Selection

The global evaluation is performed for each of the candidate gaze directions  $\bar{\theta}_i^{pt}$  in  $\bar{X}_{gd}$  using the determined fuzzy measures and partial evaluation values. User’s preference is considered through fuzzy measures which are calculated by Grabish’s graphical interpretation and identification method [9,10,15]. Fig. 2 shows user’s preference for the four criteria. For the  $i$ th criterion,  $\bar{C}_i, i = 1, \dots, 4$ , in the criteria set  $\mathbf{C}_g = \{C_{lm}, C_{sl}, C_{wp}, C_{ob}\}$ ,  $P_{ij}, i, j = 1, \dots, 4, i \neq j$ , means user-defined preference points comparing the  $i$ th criterion with the  $j$ th one. Table 1 is the calculated values of  $c_{ij}$  and  $\xi_{ij}$ , which are derived from the values of  $sv_i$  and  $I_{ij}$  as described in [10]. Using this result, the fuzzy measure values are finally identified as in Table 2.



**Fig. 2.** Setting up the relative importance factors for the criteria. (a) User-defined diamond pairwise comparison diagram. (b) Interaction diagram of four criteria. In (b), ‘+’ sign denotes the positive interaction (negative correlation), and vice versa.

**Table 1.** Weights of criteria pairs and their calculated interaction degree values

i vs. j	Horizontal			Vertical	
	$sv_i$	$sv_j$	$c_{ij}$	$I_{ij}$	$\xi_{ij}$
1 vs. 2	0.5490	0.4510	1.2174	-0.0902	0.5458
1 vs. 3	0.8358	0.1642	5.0909	0.0328	0.4707
1 vs. 4	0.2732	0.7268	0.3758	0.4917	0.0955
2 vs. 3	0.8070	0.1930	4.1818	0.2573	0.3230
2 vs. 4	0.2359	0.1930	0.3087	-0.4246	0.7436
3 vs. 4	0.0688	0.9313	0.0738	0.1238	0.1295

**Table 2.** Fuzzy measures

A	$g(A)$	A	$g(A)$
{}	0.0000	{4}	0.2750
{1}	0.0453	{1,4}	0.7494
{2}	0.1980	{2,4}	0.4589
{1,2}	0.2409	{1,2,4}	0.9089
{3}	0.0087	{3,4}	0.3644
{1,3}	0.0544	{1,3,4}	0.8454
{2,3}	0.2063	{2,3,4}	0.5455
{1,2,3}	0.2496	{1,2,3,4}	1.0000

If a representative gaze direction is not located within the visible gaze area of  $\hat{\theta}_i^{pt}$ , its corresponding partial evaluation value are excluded from the ones which are to be used for the global evaluation for  $\bar{\theta}_i^{pt}$ . Thus, the relevant partial evaluation values for criteria are used to obtain  $\bar{\theta}_i^{pt}$  if their corresponding representative gaze directions are visible by  $\bar{\theta}_i^{pt}$  using the following visibility check function  $V_j(\cdot)$ :

$$V_j(\bar{\theta}_i^{pt}, \hat{\theta}_\alpha^{ptj}) = \begin{cases} E_\alpha & \text{if } \bar{\theta}_i^{pt} \in G(\hat{\theta}_\alpha^{ptj}) \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where the function  $G(\cdot)$  generates a gaze area for the input gaze direction, and  $\hat{\theta}_\alpha^{ptj}$  denotes the  $j$ th representative gaze direction for  $F_\alpha$ . If there are multiple representative gaze directions for one criterion as the cases of  $C_{sl}$  and  $C_{ob}$ , the corresponding partial evaluation values for one criterion over  $\bar{\theta}_i^{pt}$  are added up and bounded by 1.0 using the following the normalizing function  $U_\alpha(\cdot)$ :

$$U_\alpha(\bar{\theta}_i^{pt}) = \min \left( \sum_{j=1}^{N_\alpha} V_j(\bar{\theta}_i^{pt}, \hat{\theta}_\alpha^{ptj}), 1.0 \right), \tag{6}$$

where  $N_\alpha$  is the number of representative gaze directions for  $C_\alpha$ . Then, the global evaluation for  $\bar{\theta}_i^{pt}$  considering four criteria is computed by the Choquet fuzzy integral as follows:

$$E_{global}(\bar{\theta}_i^{pt}) = \sum_{i=1}^4 \{U_i^*(\bar{\theta}_i^{pt}) - U_{i-1}^*(\bar{\theta}_i^{pt})\}g(E_i^C), \tag{7}$$

where  $U_i^*(\cdot)$ ,  $i = 1, \dots, 4$ , are reordered partial evaluation values for criteria in ascending order, and  $E_i^C = \{C_i^*, \dots, C_4^*\}$  is a corresponding reordered criteria set excluding lower  $i - 1$  criteria. This calculation is performed for all of the candidate gaze directions in  $\bar{X}_{gd}$ . If there are multiple directions with the same maximum global evaluation value in  $E_{global}$ , the mean gaze direction corresponding to those maximum ones is selected as the final one  $\theta^*$ .

## 2.4 Coordination with the Navigation Framework

The proposed gaze control-based navigation architecture uses modified univector field-based in order to avoid collisions against static and moving obstacles [16-18]. By using dynamically generated virtual obstacle and velocity modification schemes, the proposed gaze control-based navigation architecture copes with dynamic obstacle avoidance problem. Since the results of UKF-SLAM are used for the obstacle avoidance and gaze control, the whole architecture can efficiently deal with gaze control along with navigation simultaneously.

## 3 Simulation

This section describes the developed simulation environment and results. By using the developed simulator, the proposed architecture was verified through the simulation in a dynamic environment.

### 3.1 Simulation Environment

For the simulation, OpenInventor-based navigation simulator was used for the simulation as shown in Fig. 3 [2]. The simulator uses the same frame design files and walking pattern generation algorithm with ones used for the small-sized humanoid robot, HSR-IX.

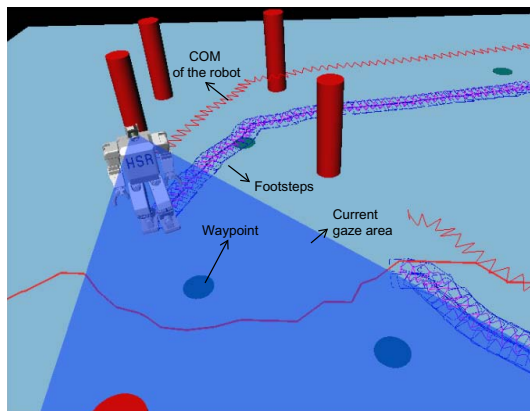
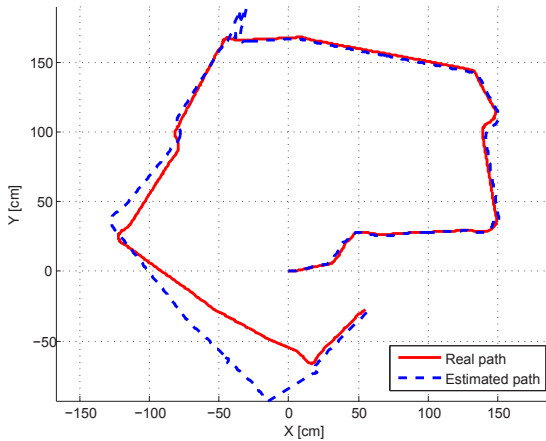


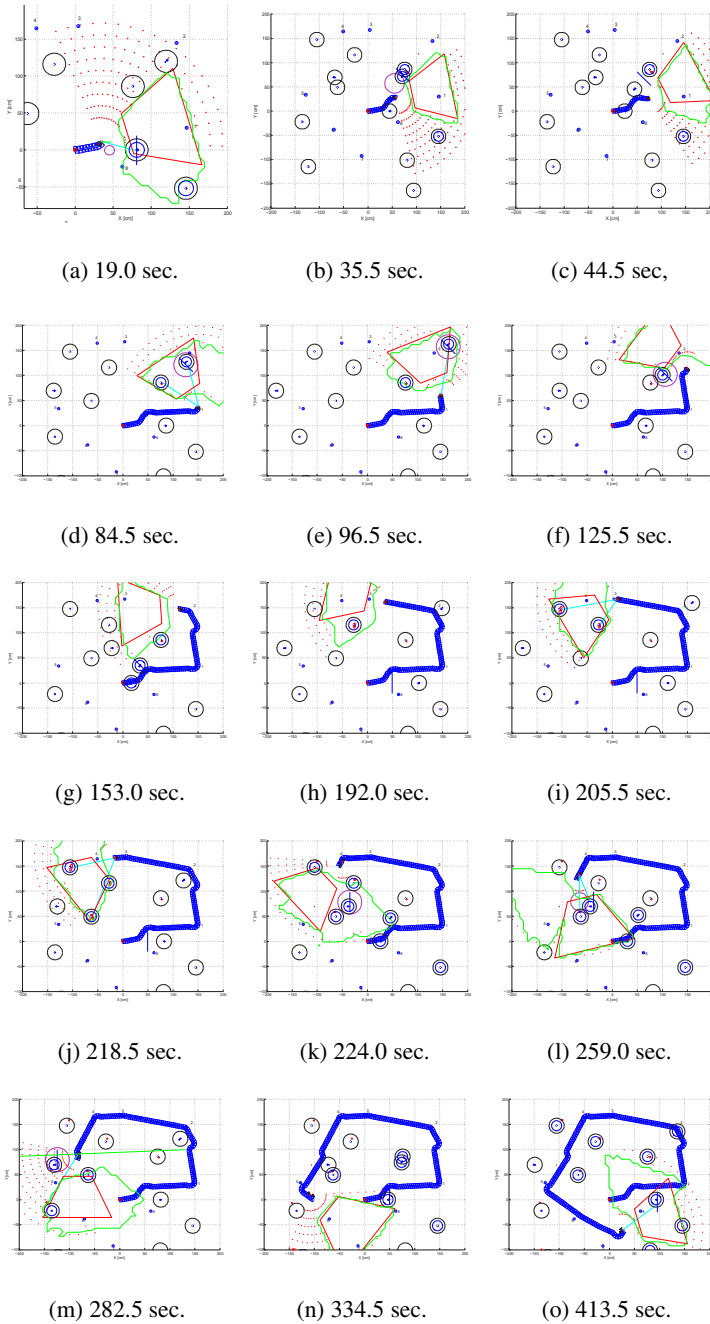
Fig. 3. HSR-IX simulator

### 3.2 Simulation Results

In this simulation, eight waypoints were traversed in order. 10 obstacles including three dynamic ones were set up. The static ones among them were located randomly, on the other hand, moving ones were assigned not to traverse the paths of the others. To show the capability of the collision avoidance scheme against moving obstacles, one of the three moving obstacles was assigned to move toward the robot at the beginning of the simulation. Fig. 5 shows the simulation result graph, where the red rectangle denotes the current Center-of-Mass (CoM) of the robot, rectangles following the footstep denote footstep history, double circles denote detected obstacles, the trapezoid in front of the robot depicts the current gaze area, and dots around the robot denote unconfident sample points used for calculating  $E_{lm}$  in Section 2.2. Fig. 4 shows the true and estimated paths which are calculated by the process of SLAM. Fig. 5 shows the process of the simulation. After avoiding the first approaching moving obstacle from 19.5 sec. to 44.5 sec., The robot distributed its gaze direction toward surrounding obstacles in order to decrease the localization uncertainty of the obstacles and itself during the traverse process between waypoints. At 259.0 sec., the robot successfully detected and avoided the collision against the approaching moving obstacle by the virtue of the proposed architecture.



**Fig. 4.** Real and estimated paths of the robot



**Fig. 5.** Sequential simulation results by the proposed architecture for 413.5 ms.

## 4 Conclusions

This paper proposed the gaze control-based navigation architecture for humanoid robots using the fuzzy integral approach. For the effective distribution of gaze direction and dynamic obstacle avoidance, four criteria were employed along with their corresponding partial evaluation functions and representative gaze directions. User's preference was considered by adopting fuzzy measures of criteria, and the global evaluations of all the candidate gaze directions were calculated through the Choquet fuzzy integral. The effectiveness of the proposed architecture was demonstrated by the simulation. In the simulation, humanoid robot could effectively deal with obstacle avoidance and gaze control in a dynamic environment by using the proposed architecture.

**Acknowledgment.** This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2012-0000150).

## References

1. Akachi, K., Kaneko, K., Kanehira, N., Ota, S., Miyamori, G., Hirata, M., Kajita, S., Kanehiro, F.: Development of humanoid robot HRP-3P. In: Proc. IEEE-RAS Int. Conf. Humanoid Robots, Tsukuba, Japan, pp. 50–55 (December 2005)
2. Yoo, J.-K., Lee, B.-J., Kim, J.-H.: Recent progress and development of the humanoid robot HanSaRam. *Robotics and Autonomous Systems* 57(10), 973–981 (2009)
3. Gutmann, J.-S., Fukuchi, M., Fujita, M.: 3D perception and environment map generation for humanoid robot navigation. *Int. J. Robot. Res.* 27(10), 1117–1134 (2008)
4. Asfour, T., Azad, P., Vahrenkamp, N., Regenstien, K., Bierbaum, A., Welke, K., Schöder, J., Dillmann, R.: Toward humanoid manipulation in human-centred environments. *Robot. Auton. Syst.* 56(1), 54–65 (2008)
5. Andreopoulos, A., Hasler, S., Wersing, H., Janssen, H., Tsotsos, J.K., Körner, E.: Active 3D object localization using a humanoid robot. *IEEE Trans. Robot.* 27(1) (February 2011)
6. Henderson, J.M.: Human gaze control during real-world scene perception. *TRENDS in Cognitive Sciences* 7(11), 498–504 (2003)
7. Seara, J.F., Schmidt, G.: Intelligent gaze control for vision-guided humanoid walking: methodological aspects. *Robot. Auton. Syst.* 48(4), 231–248 (2004)
8. Lidoris, G., Kuhlentz, K., Wollherr, D., Buss, M.: Information-based gaze direction planning algorithm for SLAM. In: 2006 6th IEEE-RAS Int. Conf. Humanoid Robots, Genova, Italy, pp. 302–307 (December 2006)
9. Takahagi, E.: A fuzzy measure identification method by diamond pairwise comparisons and  $\phi_s$  transformation. *Fuzzy Optim. and Making* 7(3), 219–232 (2008)
10. Grabisch, M.: A graphical interpretation of the Choquet integral. *IEEE Trans. on Fuzzy Syst.* 8(5), 627–631 (2000)
11. Posner, M.I., et al.: Inhibition of return: neural basis and function. *Cogn. Neuropsychol.* 2, 211–228 (1985)
12. Leek, E.C., Reppa, I., Tipper, S.P.: Inhibition of return for objects and locations in static displays. *Percept. Psychophys.* 65(3), 388–395 (2003)
13. Thrun, S.: Learning occupancy grid maps with forward sensor models. *Auton. Robots* 15(2), 111–127 (2003)

14. Bailey, T.: Mobile robot localization and mapping in extensive outdoor environments. *Doct. Thesis, Univ. Sydney, Australia* (2003)
15. Sugeno, M.: *Theory of fuzzy integrals and its applications*. *Doct. Thesis, Tokyo Institute of Technology* (1974)
16. Kim, Y.-J., Kim, J.-H., Kim, D.-S.: Evolutionary programming-based univector field navigation method for fast mobile robots. *IEEE Trans. Syst., Man, Cybern. B* 31(3), 450–458 (2001)
17. Lee, B.-J., Stonier, D., Kim, Y.-D., Yoo, J.-K., Kim, J.-H.: Modifiable walking pattern of a humanoid robot by using allowable ZMP variation. *IEEE Trans. Robotics* 24(4), 917–925 (2008)
18. Yoo, J.-K., Kim, J.-H.: Navigation framework for humanoid robots integrating gaze control and modified-univector field method to avoid dynamic obstacles. In: *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Taipei, Taiwan*, pp. 1683–1689 (October 2010)

# Comparison between Photo Interrupter and Giant Magnetoresistive Sensor for Auto Focusing System in the Digital Camera

Sakura Sikander, Han Nam Lee, and Hee-Je Kim

Dept. of Electrical Engineering, Pusan National University  
Busan 609-735, Republic of Korea  
sakura.sikander@gmail.com, heeje@pusan.ac.kr

**Abstract.** Photo interrupter (PI) sensors are used to focus the lens in the optical devices such as digital camera, close circuit camera etc. It is cheap and easy to control. Despite of the advantages it has some limitations such as it is difficult to assemble, needs small tolerance and lacks high speed and accuracy. Magnetoresistive (MR) especially giant magnetoresistive (GMR) sensor provides the solution of PI sensor's weak points. Furthermore, using GMR sensor shows better performances than PI sensor for the focusing system.

**Keywords:** photo interrupter, giant magnetoresistive sensor, auto focus, digital camera.

## 1 Introduction

Most of the recent mobile devices like cell phones, laptops, digital cameras are designed for better speed. It doesn't matter how big or what purpose it is used for, it should be fast, precise, reliable and cheap. To keep pace with the market and customer need, the production method also should be simple and cost effective. All these devices are based on sensors which are the key components that need rapid modification.

In optical applications like digital camera, close circuit camera the key component which needs continuous modification is the auto focus (AF) system to make the camera faster. So the sensor involved in AF is very important. AF means proper positioning of the lens so that the focus is sharp. Thus the key task of the sensor is to detect the position of the lens and control the lens movement. Normally for focusing photo interrupter (PI) sensor is used.

PI sensor has three tiny wings inside in it which spins between the emitter and detector inside the device. The detector counts how many times the wings interrupt the light and detects the position of the lens for AF. The PI sensor is easy to design and also cheaper than other sensors. But one of the major limitations is making precise molds of the three wings which have a significant impact on the precession of the position detection. A little tolerance in such a small scale wing can incorporate error in the sensor. Thus while assembling the sensor precession is interrupted due to



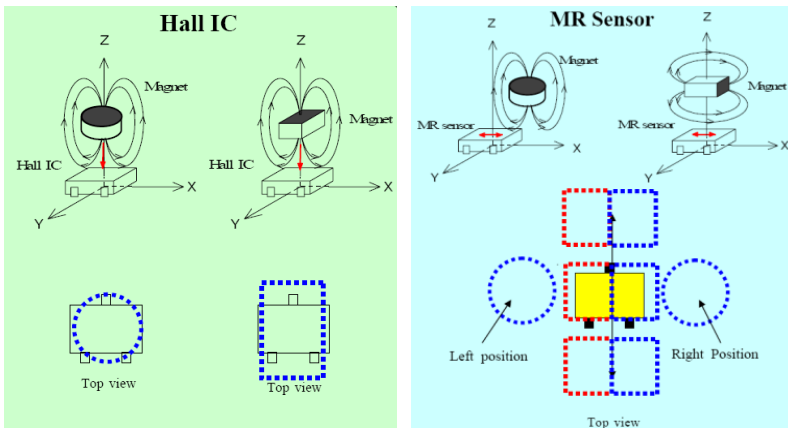
this reason. Another issue is when surface mounting the sensor in the reflow machine the precise positioning of the sensor isn't possible because the solder is liquid for certain period of time. So improper positioning of the sensor causes improper wing alignment which leads to precession problem. Thus it is very important to choose such a sensor which is independent of mounting, position and movement.

In this paper to find a solution of these drawbacks we have compared several sensors including PI sensor and concluded finding a sensor with a superior performance.

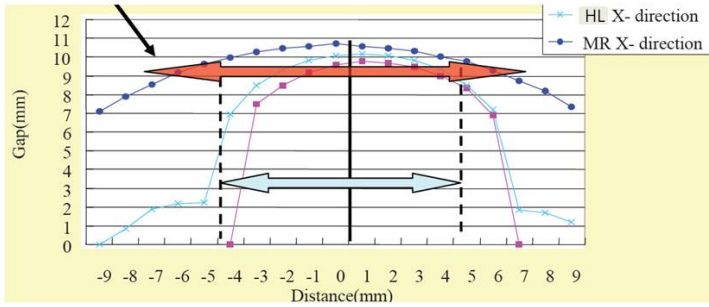
## 2 Selection of the Sensor

In this paper we will concentrate on the AF mechanism of the digital camera which has moving barrels controlled with motors to focus the lens. So controlling the barrel position with the sensor properly is our concern. To overcome the earlier mentioned drawbacks in the PI sensor we will focus on a non-contact sensor. So we are interested in the sensor that uses magnetic signals. In that case hall sensor and magnetoresistive sensor is a good choice.

As shown in figure 1 we can see that hall sensor detects vertical magnetic component where else the magnetoresistive (MR) sensor detects the horizontal component. So if the target object is not in vertical position then the sensor will not detect properly and will create a weaker signal than normal position which makes it a poor sensing device compared with the MR sensor. So assembly of the hall sensor will cause loss of accuracy and effectiveness of the sensor because the sensor barrel is round in shape where the magnetic strip will be wound around which means the magnetic signal may not be vertical properly and also improper assembly will affect the signal.



**Fig. 1.** Magnetic field orientation of a Hall sensor (*left*) and a MR sensor (*right*)



**Fig. 2.** Range of the sensors

Figure 2 shows the range of the two sensors. In terms of range also the MR sensor has a significant advantage which means MR sensor has larger tolerance for assembling. Compared to sensitivity and signal level giant magnetoresistive (GMR) sensor has the same principle but has higher sensitivity than the MR sensor [1]. GMR sensor has more effect for thin multi-layer structures [2]. From formula (1) we can find the MR ratio where  $R_{\max}$  and  $R_{\min}$  are the maximum and minimum magneto resistance respectively. GMR produces MR ratio of 20-50%, which means that 20-50% of the obtained signal can be collected from the sensor [3] which is very satisfactory.

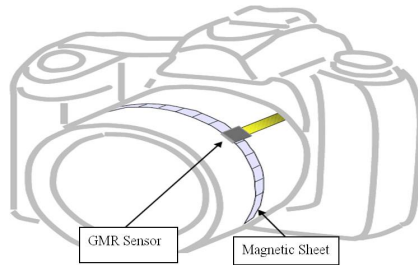
$$MR_{Ratio} = \frac{R_{\max} - R_{\min}}{R_{\min}} \quad (1)$$

### 3 Results

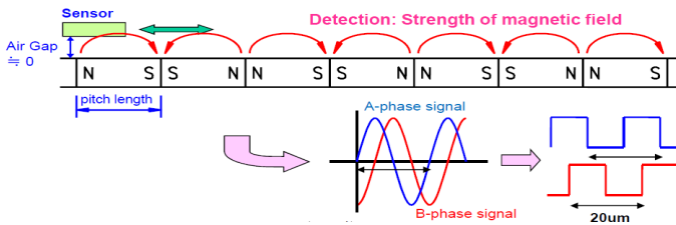
A Magnetic sheet is installed inside the lens barrel of a digital camera and using the rotation of the lens barrel the GMR sensor detects the magnetic field from the magnetic sheet as shown in figure 3.

For our experiment GMR sensor from “Hitachi Metal” was used [4], which generates a pair of signal. The signals are same except the phases are different as shown in figure 4. The bridge circuit’s opposite polarity makes the difference between A and B’s phase. If we multiply A and B better resolution can be obtained. The pitch distance was set 20um for our experiment and was multiplied four times and achieved 5um of resolution. In case of Hitachi Metals GMR sensor system, it is possible to multiply the pitch distance (i.e. 400um) 256 times which shows 1.56um of resolution. A typical PI sensor has a resolution of 1mm. If Hitachi GMR sensor system is compared with the PI sensor system, the resolution can be improved at least 10 to 700 times. In our experiment with our designed circuit we had about 200 times better resolution than the PI sensor system.

Our circuit was designed to measure the signal for the experiment using formula (2) where  $V_{A-phase}$  and  $V_{B-Phase}$  are phase voltage of A and B signals respectively.



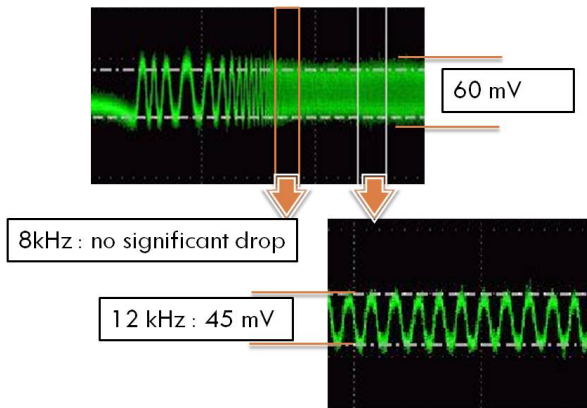
**Fig. 3.** Position of the GMR sensor and magnetic sheet



**Fig. 4.** GMR sensor signal with magnetic sheet

$$\theta = \tan^{-1} \left( \frac{V_{B-Phase}}{V_{A-Phase}} \right) \quad (2)$$

The output signal using our circuit is shown in figure 5. The camera lens barrel rotated at a speed of 120 rpm with GMR sensor system having 20um of pitch. For output signal there was no significant attenuation at 8 kHz from 60mV signal. At 12 kHz the signal attenuated 25% and dropped to 45mV, which is still larger than the PI sensor signal at this speed. The PI sensor had the same output at 1 kHz. The minimum detection voltage for our design was 37mV.



**Fig. 5.** GMR sensor output frequency

For a sensor its repeatability, durability and accuracy are very important factors. For production and manufacturing it is also very important to know which parameter will effect its performance most in the production line. So the output of the GMR sensor was analyzed by installing the sensor with some installation error. The test set up shown in figure 6 was used for the analysis to rotate the lens barrel. In the experiment clockwise (CW) and counter clockwise (CCW) rotation had a smooth pattern shown in figure 7 which justified there were no additional noises due to the spinning pattern in the output. From figure 8 we can see the change in the output due to different physical parameter (i.e. pitch angle, gap, azimuth, offset, overlap and roll angle) change. Yellow, red and blue line in the figure corresponds to the signal in CW direction, CCW direction and the difference between CW and CCW signal respectively for phase A and B individually. The green box shows the signal where it is greater than 85mV.

Figure 8 shows that change in azimuth and specially roll angle effects the output significantly. Thus in the assembly line change in roll angle and azimuth should be given a special attention. However minimum detection voltage was 37mV for the setup so the angle change didn't matter that much in the experiment. From the analysis normal and worst cases were detected which are shown in table 1. To make the experiment more reliable these cases were used in the durability test.

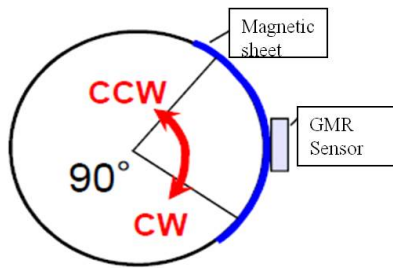


Fig. 6. GMR sensor test setup

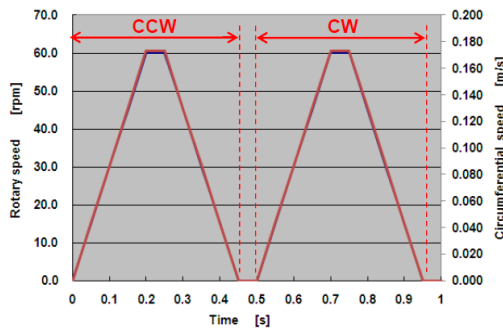


Fig. 7. Rotary speed and cycle time of the lens barrel

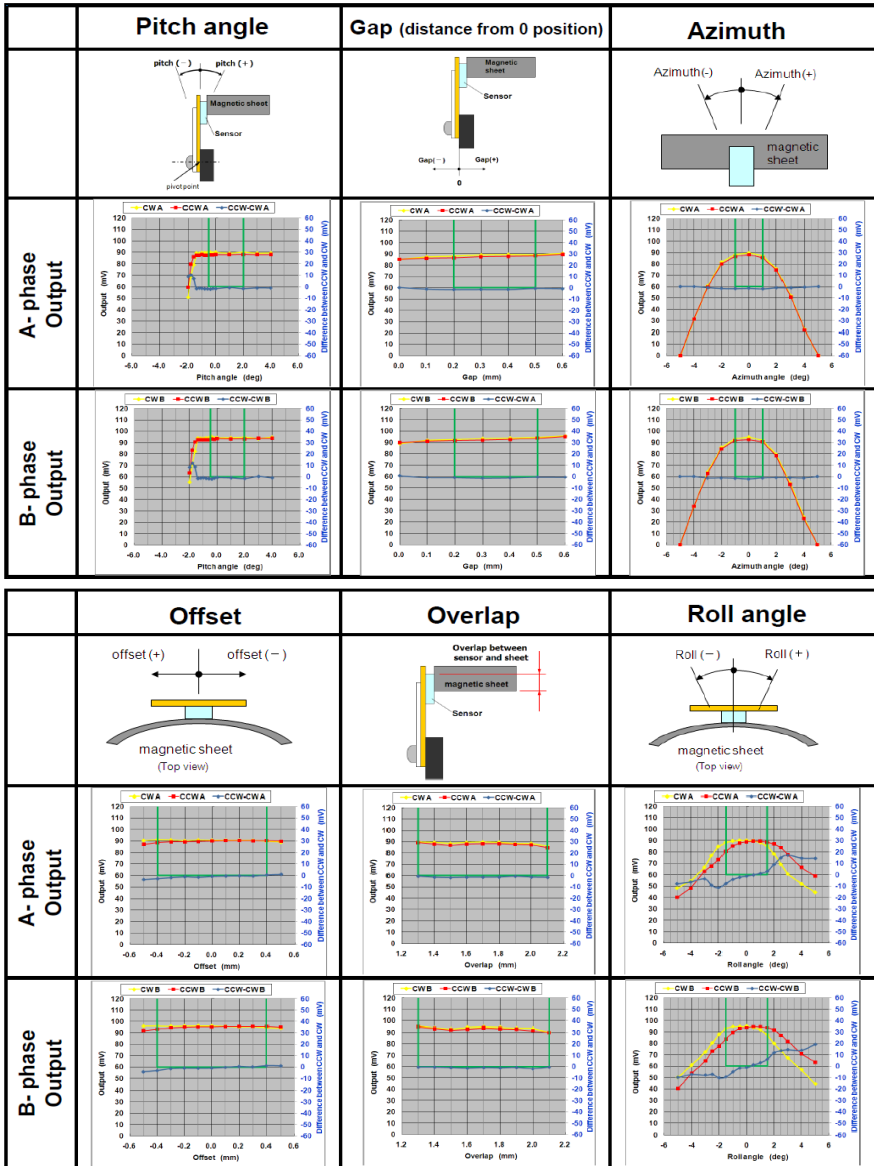


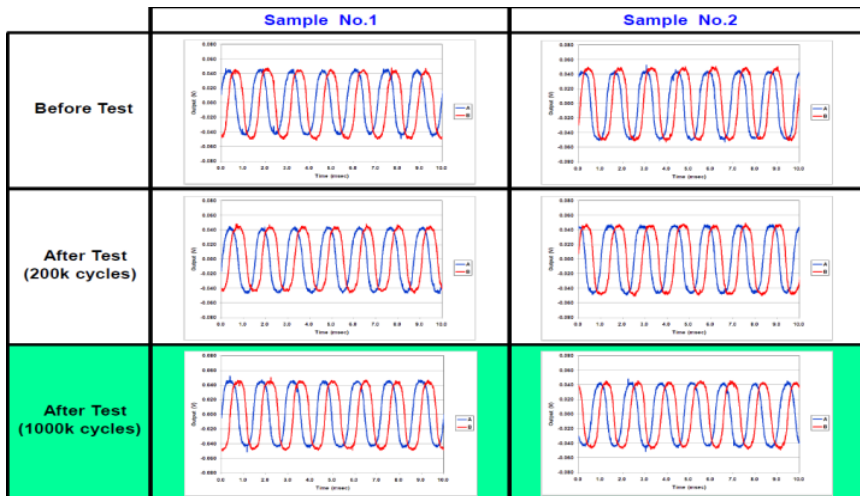
Fig. 8. Output of the GMR sensor for different installation error

Using the conditions from table 1 the durability test was accomplished. The durability test result is shown in figure 9 where red and blue line show the signals for A and B phase respectively. We did the test up to 1000,000 cycles. The PI sensors durability is around 200,000 cycles where else in our experiment the GMR sensor

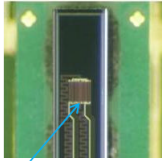
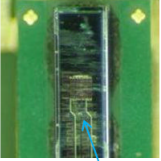
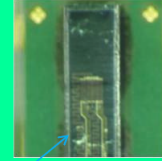
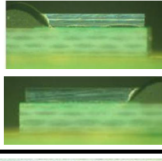
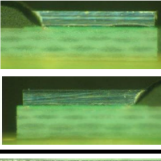

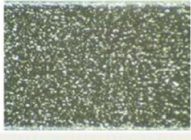
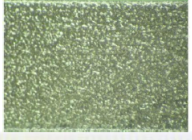
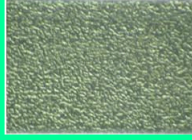
didn't have any significant output change even after 1000,000 cycles of operation. So it is evident that GMR sensors reliability and durability is superior to the PI sensor. Because the GMR sensor is a non-contact type sensor the physical condition also remained well even after 1000,000 cycles of operation which is shown in figure 10. The GMR sensor lifetime was calculated using the Arrhenius equation and found 39,000 hours.

**Table 1.** Durability test conditions

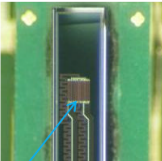
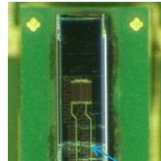
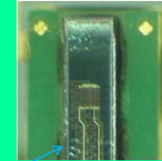
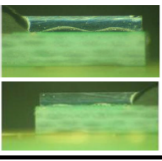
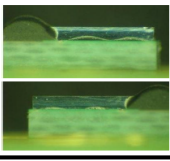
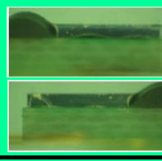
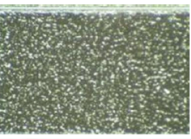
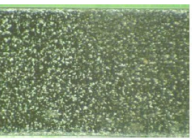
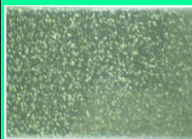
Sample #	Setting	Pitch	Gap	Azimuth	Offset	Overlap	Roll
		deg	mm	mm	deg	mm	deg
#1	Normal	0.0	0.0	0.35	0.0	1.7	0.0
#2	Worst	2.0	1.0	0.35	1.5	2.1	1.5



**Fig. 9.** Result of the durability test of the GMR sensor

Sample No.1	Before Test	After Test (200k cycles)	After Test (1000k cycles)	result
Surface of Sensor	 element	 White area is lubricant, not scratch		No scratch No Crack
Side of sensor				No scratch No Crack
Surface of scale				No scratch No Crack

(a) For normal condition

Sample No.2	Before Test	After Test (200k cycles)	After Test (1000k cycles)	result
Surface of Sensor	 element	 White area is lubricant, not scratch		No scratch No Crack
Side of sensor				No scratch No Crack
Surface of scale				No scratch No Crack

(b) For worst condition

**Fig. 10.** Physical condition of the GMR sensor after the durability test

## 4 Conclusion

Sensor plays a key role in the AF system. All the test results show that the GMR sensor can play far more superior role than the PI sensor for the AF system.

The GMR sensor has larger range which will make it less sensitive to the assembly error and give higher accuracy. Also small defects in the assembly will not cause any error in the sensing. GMR sensor has higher reliability and durability than the PI sensor and the life time is also significantly higher than the PI sensor. Though the PI sensor is cheaper than the GMR sensor but due to higher life time, reliability and durability the total cost will decrease. Compared to the PI sensor the GMR sensor is a better solution for digital cameras AF mechanism.

**Acknowledgements.** This research was supported by the MKE(The Ministry of Knowledge Economy), Korea, under the Human Resources Development Program for robotics support program supervised by the NIPA(National IT Industry Promotion Agency) (NIPA-2011-C7000-1001-0009).

## References

1. Smith, C.H., Schneider, R.W.: Low-Field Magnetic Sensing with GMR Sensors. Sensors EXPO (1999)
2. Baibich, M.N., Broto, J.M., Fert, A., Nguyen Van Dau, F., Petroff, F.: Giant Magnetoresistance of (001)Fe/(001)Cr Magnetic Superlattices. Phys. Rev. Lett. 61, 2472–2475 (1988)
3. Jander, A., Smith, C., Schneider, R.: Magnetoresistive Sensors for Nondestructive Evaluation. In: SPIE International Symposium, vol. 5770, pp. 1–13 (2005)
4. Data Specification from Hitachi Metals (2011)



# Natural Terrain Detection and SLAM Using LIDAR for an UGV

Kuk Cho<sup>1</sup>, SeungHo Baeg<sup>2</sup>, and Sangdeok Park<sup>2</sup>

<sup>1</sup> Dept. of Intelligence Robot Engineering, University of Science and Technology  
176 Gajung-dong, 217 Gajungro Yuseong-gu Deajeon, 305-350, Korea  
googi33@ust.ac.kr

<sup>2</sup> Dept. of Applied Robot Technology, KITECH  
1271-18, Sa-3 dong, Sangrok-gu, Ansan, 426-791, Korea  
{shbaeg, sdpark}@kitech.re.kr

**Abstract.** This paper describes natural terrain detection and SLAM algorithms using a LIDAR sensor for an unmanned ground vehicle. We describe how features are detected from natural terrain, and then localize the vehicle's position and compose a map with the detected features. The LIDAR is equipped on the experimental vehicle to scan natural terrain. The scan data include many kinds of natural disturbances on uneven terrain: a banded tree, a branch of a tree, non-uniform size of bushes, and undefined or unexpected objects. We apply a RANSAC (RANdom SAMple Consensus) algorithm to discriminate ground point cloud data and object point cloud data, and then separate bush points and tree points by a combination of two algorithms, GMM (Gaussian Mixture Model) and EM (Expectation Maximization). The GMM and EM algorithms are for extracting features and classifying groups, respectively. We propose a double FCM (Fuzzy C-mean clustering) algorithm to robustly estimate the number of trees and their positions. The Extended Kalman Filter applied to simultaneous localization and mapping (EKF-SLAM) is composed of the extracted tree features. The *mahalanobis* distance is applied to retain consistency for feature correspondence, which is used data association. Finally, we show the results obtained from implementation of the approach on a tree-covered mountain.

**Keywords:** object detection, SLAM, natural terrain, extended Kalman filter, data association.

## 1 Introduction

Unmanned systems have been studied in various fields. UGVs (Unmanned ground Vehicles), UAVs (Unmanned Aerial Vehicles), and UUVs (Unmanned Underwater Vehicles) are respectively operated on the ground, in air, and underwater. Since the Grand Challenge event, in which UGVs follow a designed route in the desert, research in the field of UGVs has focused intensely on achieving missions. In the case of urban area research, which has also been widely studied since the Urban Challenge event, vehicles encounter various sizes of objects and use different speeds during the

route. Another research field that has recently seen much attention is unstructured environment recognition – specifically natural terrain [1, 2, 3].

To achieve even general missions in tree-filled and bush-filled mountains, an UGV should recognize the vehicle's position and the positions of surrounding objects, i.e. localization of the vehicle and mapping with objects. An UGV driving in natural terrain requires environment recognition, which should be based on feature object detection. The features can be trees, bushes, canopy, shrubs, and so on.

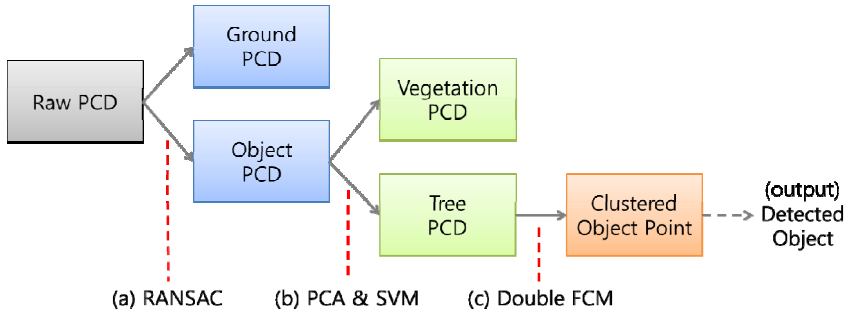
In a dense forest or a mountainous area, GPS cannot receive a sufficient signal from satellites to estimate precise positions due to blockage of the signal. Even data from a high-precision INS-GPS are degraded by uncertain measurement. Image sensors are especially sensitive to light. In a forest, light is frequently changed by the leaves of trees. An image sensor cannot guarantee performance of environment recognition in either day or night. However, LRF (Laser Range Finder) and LIDAR (Light Detection And Ranging) sensors are not sensitive to light and are robust to various external conditions [3, 4]. LRF and LIDAR devices are thus primary equipment to acquire and analyze environment information. Environment resources for detection and tracking include either artificial landmarks or natural landmarks. There are no artificial landmarks and no unique landmarks, however, in natural terrain. Only numerous trees, which have different sizes and are not uniformly round, are available for landmarks. Bushes are similarly not uniform size and have different heights. Besides, the sensor sometimes provides some part of obstacle scan data.

Numerous studies on the use of cameras and LIDAR for environment recognition have been reported. However, they have only been applied to recognize traversability on a given trail. Two methods are depending on available for measurement devices. In 2D LRF studies, it is assumed that trees are round and have only one trunk. In 3D LIDAR studies, features are extracted using the nearest neighbor point. A Gaussian Mixture Model (GMM) is then made using the Expectation-Maximization (EM) algorithm to classify the object. To recognize a point cloud data group, Principle Component Analysis (PCA) and Support Vector Machine (SVM) are used [5, 6]. 3D cluttered range data processing is restricted with an uneven environment and by processing time.

In this paper, we describe both natural terrain detection for feature extraction and SLAM for vehicle localization and object mapping using a LIDAR sensor for an unmanned ground vehicle. In section 2, we propose a method to extract the significant objects from natural terrain of point cloud data. Section 3 describes how to apply the extended Kalman filter with a simultaneous localization and mapping algorithm. In Section 4, we show two implementation results for an object detection method and an EKF-SLAM method.

## 2 Object Detection

Object detection is a process of a point cloud classification, as seen in figure 1 [7]. We acquire the point cloud data by scanning a downward-looking multi-layer LIDAR sensor in natural terrain. The process is a feature extraction procedure from raw point cloud data that include both significant information and insignificant information.



**Fig. 1.** The general point cloud processing illustrates how features detected from the environment scan data (raw point cloud data) are converted into a map state as landmarks

Finally, the object number and its position are extracted. In this section, we classify feature landmarks from natural terrain 3D scanning data.

### 2.1 Terrain and Object Discrimination

First, it is assumed that the scanned point cloud data consist of two parts: a terrain part and an object part. On the ground, terrain comprises pebble, soil, sand, etc. Over the ground, objects comprise tree, bushes, rocks, etc. The point cloud data are composed mostly of terrain points. To reduce data processing and computing time, we discriminate between terrain point sand object points with a RANSAC algorithm, as in figure 1-(a) [8].

Let  $\mathbf{X} = \{\mathbf{x}_i | \{x_1, y_1, z_1\}, \dots, \{x_N, y_N, z_N\}\}$  be the  $N$  number of raw point cloud data.  $\hat{\mathbf{x}}_m = \{\{x_i, y_i, z_i\}: i = 1, \dots, M\}$  are the uniformly selected  $M$  random samples in the point cloud data. We assume that a scanned ground model is defined as a plane:

$$\hat{\eta} = f(\hat{\mathbf{x}}_m) = \{ax_m + by_m + cz_m + d\}, \{a, b, c, d\} \in \mathbb{R} \tag{1}$$

where  $\{a, b, c, d\} \in \mathbb{R}$ . The evaluation function ( $\epsilon$ ) is the distance between the candidate ground model and point cloud data. The distance is less than  $\omega_{in}$ , inliner. If the number of ground inliner points is greater than the previous candidate point number, the parameters and candidate point number are stored. After the iteration process reaches a certain time, the process is finished, as delineated in algorithm 1.

#### Algorithm 1

Let  $\mathbf{X} = \{\mathbf{x}_i | \{x_1, y_1, z_1\}, \dots, \{x_N, y_N, z_N\}\}$  be  $N$  point cloud raw data

**Repeat**

1. Select  $m$ -points

$$\text{point } \hat{\mathbf{x}}_{m,m} \sim u(0, N) \begin{cases} 1/N & \text{for } x \in [0, N] \\ 0 & \text{otherwise} \end{cases}$$

2. Estimate the model parameters ( $\hat{\eta} = f(\hat{\mathbf{x}}_m)$ ) with selected  $m$ -points.
3. Evaluate the candidate ground model with the evaluation function ( $\varepsilon$ ) within inliner point threshold ( $< \omega_{in}$ ) to all of  $N$ -number of point cloud data  $\{i = 1, \dots, N\}$ .

$$\varepsilon_i = \frac{ax_i + by_i + cz_i + d}{\sqrt{a^2 + b^2 + c^2}} < \omega_{in}$$

- A. If the number of ground inliner points is more than the remaining parameter's inliner points, keep the parameter and update inliner points.

**Until** the maximum number of iterations  $k$  is reached.

## 2.2 Vegetation and Object Points Discrimination

The object point cloud data consist of vegetation point cloud data and object point cloud data, as in Figure 1(b). The vegetation and objects represent bushes and trees, respectively. After extracting the features from the object point cloud data, we applied a Gaussian Mixture Model (GMM) using Expectation Maximization (EM) [5] and a Bayesian classifier to discriminate the tree objects and bush objects. First, we made a covariance matrix with the nearest neighbor points for each point, and then extracted the feature as an eigenvalue of the matrix. The nearest neighbor points represent point cloud data in a certain range. We made a group for training and a classifier by a GMM with EM. The covariance matrix for  $N$  3D-point set  $\mathbf{X}_i = \{x_i, y_i, z_i\}$  with  $\bar{\mathbf{X}} = 1/N \sum \mathbf{X}_i$  is defined as:

$$\frac{1}{N} \sum (\mathbf{X}_i - \bar{\mathbf{X}})(\mathbf{X}_i - \bar{\mathbf{X}})^T \quad (2)$$

The covariance matrix is decomposed into principal components. We define eigenvalues  $e_0, e_1, e_2$  as the eigenvectors corresponding to the eigenvalues  $\lambda_0, \lambda_1, \lambda_2$ , where  $\lambda_0 \geq \lambda_1 \geq \lambda_2$  respectively. The bush point cloud data are  $\lambda_0 \cong \lambda_1 \cong \lambda_2$ . In the case of tree point cloud data, the eigenvalues will be  $\lambda_0, \lambda_1 \gg \lambda_2$  [5]. Second, we use a parametric model for a classifier by a Bayes classifier. The classification model employs a pre-defined data set. The class parametric Gaussian model is expressed as a mean and covariance ( $\mu^k, \Sigma^k$ ) of the  $k$ -th class. The  $k$ -th class has a probability density function  $p(\mathbf{x}|C_k)$ .  $C_k$  represents each class: the vegetation group and the object group. We assume that each class has the same portion of the entire point cloud data.

$$y(x) = \begin{cases} 1 & \text{if } p(\mathbf{x}|C_1) > p(\mathbf{x}|C_2) \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

Each class has a different mean ( $\mu^k$ ) and covariance ( $\Sigma^k$ ). Using the Bayes classifier, we can compose the decision rule as

$$y(\mathbf{x}) = \arg \max_k \left\{ (\mathbf{x} - \mu_k)(\Sigma_k)^{-1} (\mathbf{x} - \mu_k)^T \right\} \quad (4)$$

The binary classifier separates the vegetation points from the object points.

### 2.3 Object Clustering

Following vegetation and object point discrimination, the object points can be utilized as landmarks for localization. One object structure is described with many object points. The cluster method makes objects with dispersion point clouds. Finally, we can recognize how objects are in the scan scene and where the objects exist in the scene. We applied the fuzzy c-means (FCM) algorithm, which finds the solution by iterated calculation [9, 10]. First, the basic FCM algorithm can be summarized as follows.

Let  $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$  be the  $n$ -number of given point cloud data. Each point cloud datum at the  $k$ -th point data represents a 3D coordinate,  $\mathbf{x}_k = \{x_k, y_k, z_k\}$ . Let  $U_{cn}$  be the matrix of the partition data,  $Z^{c \times n}$ .  $c$  is the number of clusters and  $2 \leq c < n$ . The fuzzy c-partition space for  $\mathbf{X}$  is the set

$$M_{fcm} = \{U \in U_{cn} : u_{ik} \in [0, 1], \sum_{i=1}^c u_{ik} = 1, 0 < \sum_{k=1}^n u_{ik} < n\} \quad (5)$$

where  $u_{ik}$  is the membership value of  $x_k$  in cluster  $i$  and the value exists up to the number of clusters. The FCM algorithm is for finding an optimal fuzzy c-partition. In other words, the minimizing object function is described. It is associated with the generalized least-squared errors function as follows:

$$J_m(U, V; X) = \sum_{k=1}^n \sum_{i=1}^c (u_{ik})^m \|x_k - v_i\|^2 \quad (6)$$

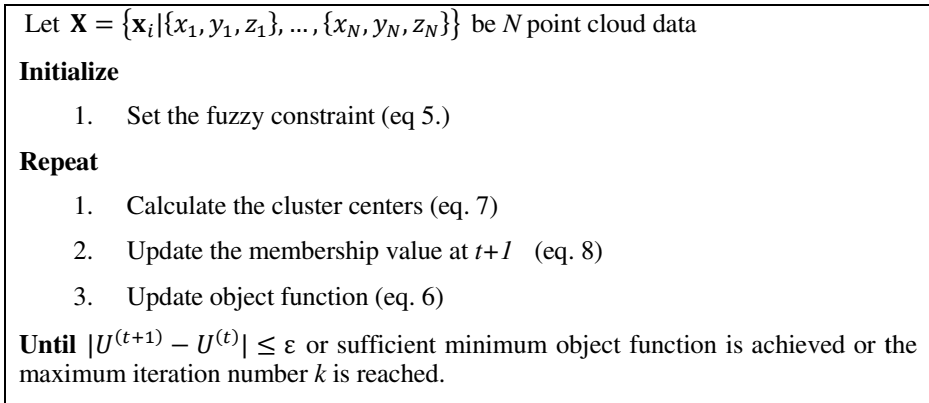
where  $V = \{v_1, v_2, \dots, v_c\}$  is a matrix of unknown cluster centers. To initialize the algorithm, a fuzzy c-partition  $U^0$  is generated randomly and the iteration number represents  $t$ . The iterative process works as follows. Given the membership values ( $u_{ik}^{(t)}$ ) and point cloud data ( $\mathbf{x}_k$ ), the cluster centers  $v_i^{(t)}$  are estimated by

$$v_i^{(t)} = \frac{\sum_{k=1}^n (u_{ik}^{(t)})^m \mathbf{x}_k}{\sum_{k=1}^n (u_{ik}^{(t)})^m} \quad (7)$$

Given the new cluster centers  $v_i^{(t)}$  at (7), the membership value is updated to the next step ( $t + 1$ ) as

$$u_{ik}^{(t+1)} = \left[ \sum_{j=1}^c \left( \frac{\|x_k - v_j^{(t)}\|^2}{\|x_k - v_i^{(t)}\|^2} \right)^{\frac{2}{m-1}} \right]^{-1} \quad (8)$$

There are three requirements to finish the iteration: the transition of c-partition on iteration, a sufficient minimum of object functions, and maximum iteration. The FCM method is explained as follows:

**Algorithm 2**

**Fig. 2.** The double fuzzy c-mean clustering diagram

For enhanced clustering, we apply double fuzzy-c mean clustering, as in figure 2. There are many false detections with one clustering processing. The first FCM clustering focuses on reducing noise and then the second FCM clustering makes extract clusters. The first clustering under-fits the object parameter and the second clustering over-fits an appropriate parameter to extract the object number and position.

### 3 EKF SLAM

In this section, we use the Extended Kalman Filter approach for simultaneous localization and mapping [11]. To find the correspondence feature, the *mahalanobis* distance method, NIS (normalized innovation squared), is applied. The vehicle state matrix and the covariance matrix are represented by the current position and direction with cross-correlation on a Cartesian coordinate frame, as depicted Fig. 3. The map expression consists of distinguishable and static landmarks corresponding with the tree positions. The covariance matrix represents cross-correlation between features. We can make an augmented state matrix with the vehicle state matrix and the feature map state matrix. Also, the vehicle and map have a correlation. The EKF-SLAM is briefly summarized as the following algorithm.

**Algorithm 3.** The EKF SLAM algorithm

Predict :

$$\hat{\mathbf{x}}_k^- = f(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1})$$

$$\mathbf{P}_k^- = \mathbf{A}\mathbf{P}_{k-1}\mathbf{A}^T + \mathbf{W}\mathbf{Q}_{k-1}\mathbf{W}^T$$

Data Association :

$$M_{ij} = v_{ij}^T \mathbf{S}_{ij}^{-1} v_{ij} < \gamma$$

Update :

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}^T (\mathbf{H}\mathbf{P}_k^- \mathbf{H}^T + \mathbf{R})^{-1}$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k (\mathbf{z}_k - h(\hat{\mathbf{x}}_k^-))$$

$$\mathbf{P}_k = \mathbf{P}_k^- - \mathbf{K}_k \mathbf{H}\mathbf{P}_k^-$$

Augment :

$$\hat{\mathbf{x}}_k = g(\hat{\mathbf{x}}_k, z_k)$$

$$\mathbf{P}_k = g(\mathbf{P}_k, z_k)$$

**3.1 Motion Model**

The prediction process requires motion modeling about the vehicle movement. It estimates the dead-reckoning compared with the previous pose. The map features are not considerable in this process. We make a bicycle type model and choose a real-wheel driving type car model, as seen in figure 3.

$$\hat{\mathbf{x}}_k^- = f(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}) = \hat{\mathbf{x}}_{k-1} + \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \\ \frac{\tan(\phi)}{L} \end{bmatrix} v \cdot \Delta t, \mathbf{u}_{k-1} = \begin{bmatrix} \phi \\ v \end{bmatrix} \tag{9}$$

where  $x$  and  $y$  are the  $x$ -axis translation (forward) and  $y$ -axis translation (side).  $\theta$  and  $\phi$  are the body angle and steering angle, respectively.  $L$  is the body length and  $v$  is the velocity of the vehicle. The transition of the vehicle position is that the state estimate is changed with the state covariance matrix and cross-correlation matrix. An estimate of the vehicle position with covariance is commonly obtained using odometry and a vehicle model as follows:

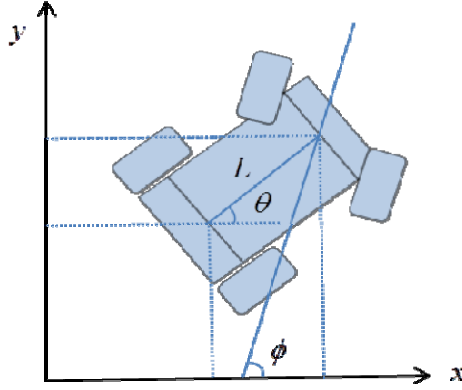


Fig. 3. Vehicle coordination on Cartesian coordinates

$$\mathbf{P}_k^- = \mathbf{A}\mathbf{P}_{k-1}\mathbf{A}^T + \mathbf{W}\mathbf{Q}_{k-1}\mathbf{W}^T$$

$$\mathbf{A} = \frac{\partial f}{\partial \mathbf{x}} = \begin{bmatrix} 1 & 0 & -\sin(\theta) \cdot v \cdot \Delta t \\ 0 & 1 & \cos(\theta) \cdot v \cdot \Delta t \\ 0 & 0 & 1 \end{bmatrix} \tag{10}$$

$$\mathbf{W} = \frac{\partial f}{\partial \mathbf{u}} = \begin{bmatrix} 0 & \cos(\theta) \cdot v \cdot \Delta t \\ 0 & \sin(\theta) \cdot v \cdot \Delta t \\ \frac{v}{L} \cdot \sec^2(\phi) \cdot \Delta t & \frac{\tan(\phi)}{L} \cdot \Delta t \end{bmatrix}$$

### 3.2 Sensor Model

The LIDAR sensor scans the front of the vehicle on the natural terrain. From the environment, the features are extracted based on the object detection algorithm, which is explained in the previous section. The observed features are expressed with range and bearings. The augmented state matrix and the measurement are as given below:

$$\mathbf{z} = \begin{bmatrix} r \\ \theta \end{bmatrix}, \quad \mathbf{R} = \begin{bmatrix} \sigma_r^2 & \sigma_{r\theta}^2 \\ \sigma_{r\theta}^2 & \sigma_\theta^2 \end{bmatrix} \tag{11}$$

where  $r$  and  $\theta$  describe the range and bearings of the observed features, respectively.  $\mathbf{R}$  is the observation covariance. The estimate sensor parameter is given by the following equation.



$$\hat{\mathbf{z}}_i = \mathbf{h}_i(\hat{\mathbf{x}}_a) = \begin{bmatrix} \sqrt{(\hat{x}_i - \hat{x}_v)^2 + (\hat{y}_i - \hat{y}_v)^2} \\ \arctan\left(\frac{\hat{y}_i - \hat{y}_v}{\hat{x}_i - \hat{x}_v}\right) - \hat{\phi}_v \end{bmatrix} \quad (12)$$

where  $\hat{x}_i$  and  $\hat{y}_i$  are the extracted feature position in the state vector.  $\hat{x}_v$ ,  $\hat{y}_v$ , and  $\hat{\phi}_v$  are the vehicle's current x and y position and direction.

### 3.3 Data Association

Data association is the process whereby newly observed features are associated with previously known features. The feature are described by their position. The new and previously known features are assumed to have Gaussian characteristics. It is possible that the newly observed features correspond to known features. We should consider a reasonable neighborhood of new observed features. In the observation space, association validation is conducted. A validation gate dictates the correspondence between a measurement ( $\mathbf{z}_i$ ) and a predicted observation ( $\hat{\mathbf{z}}_j = h(\hat{\mathbf{x}}_j)$ ) for the target  $\mathbf{x}_j$ .

The most common statistical validation gate is based on the *normalized innovation squared* (NIS), also known as the Mahlanobis distance. Given an observation innovation ( $v_{ij} = \mathbf{z}_i - h(\hat{\mathbf{x}}_j)$ ) with covariance  $\mathbf{S}_{ij}$ , the NIS is defined as

$$M_{ij} = v_{ij}^T \mathbf{S}_{ij}^{-1} v_{ij} < \gamma \quad (13)$$

## 4 Experiment

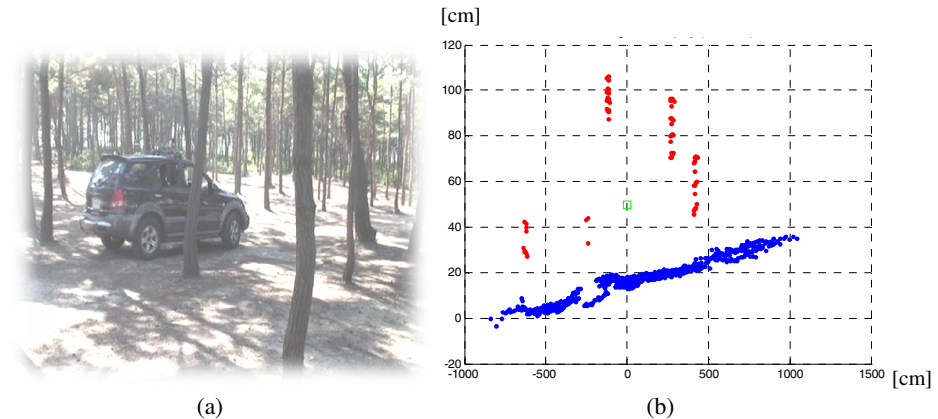
In this section, we describe the implemented system and environment with experimental results. The experimental vehicle is equipped with a laser scanner sensor, SICK LD-MRS. It has a 4-ch scan laser. All devices for navigation of the vehicle are installed and use process distributed computing: a camera, a LIDAR, a GPS, and an INS. Each system communicates through Ether-Cat communication in real-time. Figure 4 shows the experimental vehicle and natural terrain environment. The process is executed with one frame point cloud scan data.

In the detection part of the process, the clustering part requires the distinguished features. The estimating cluster is an important part to determine system performance for simultaneous localization and mapping for an unmanned ground vehicle. We compare three clustering results: general fuzzy c-mean clustering, k-mean clustering, and double fuzzy c-mean clustering. The clustering error arises from scan disturbance: banded trees, non-uniform size trees, non-uniform size trees, and canopy height.

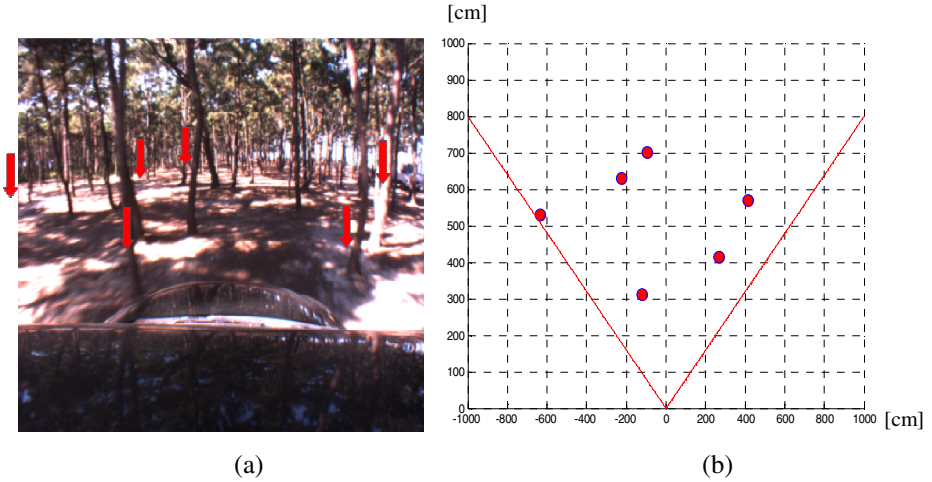
**Table 1.** The object detection results compared to clustering methods

Method	True	False	Percent
Single FCM Clustering	145	65	69%
K-Mean Clustering	155	55	73%
Double FCM Clustering	194	16	92%

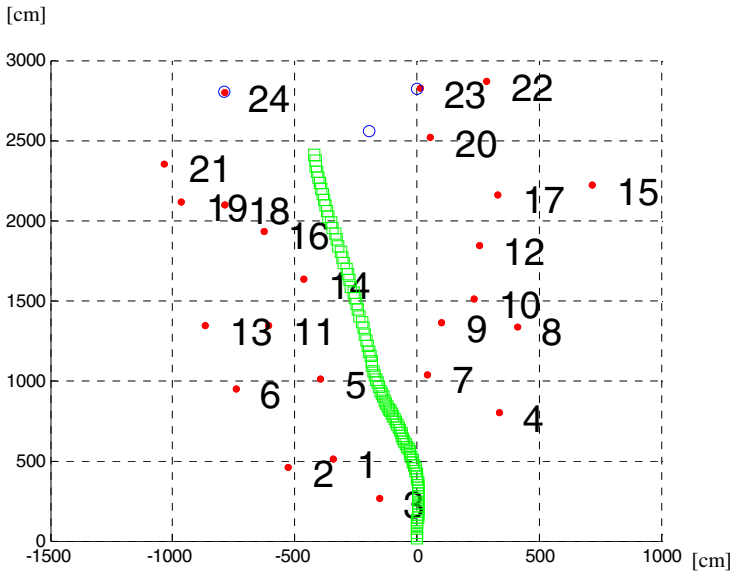
Figure 4(b) shows the raw scan point cloud data of one frame, which is scanned by 4-ch LIDAR on the vehicle. The figure 4(b) shows the x-z axis. The blue point is ground and the red points are tree points, and the green point is the vehicle's position. The color describes the results of the detection from extracted object features and classification of tree point cloud data as landmark features. Figure 5 shows the extracted tree features and corresponding trees on the image in the local top view. In figure 5(a), the red arrows describe the detected trees with scanning point cloud data. In figure 5(b), the red circles represent the extracted tree features. The red line represents the field of view of the scan sensor. Figure 6 shows the tree feature extraction result, which is based on the map generation. Figure 7 represents the covariance of the vehicle pose: the position and direction. It is going increase before loop closing. When the vehicle pose rapidly declines, a closed loop occurs, as seen in Figure 7. It does not require significant resources to decrease the covariance. Figure 8 shows point cloud feature data for all frames: green color points are the moving path of the vehicle and red color points are feature objects. The figure includes a lot of disturbance because of the canopy in front of the vehicle, the banded trees, and the separated branches, which deleteriously affect the feature clustering number.



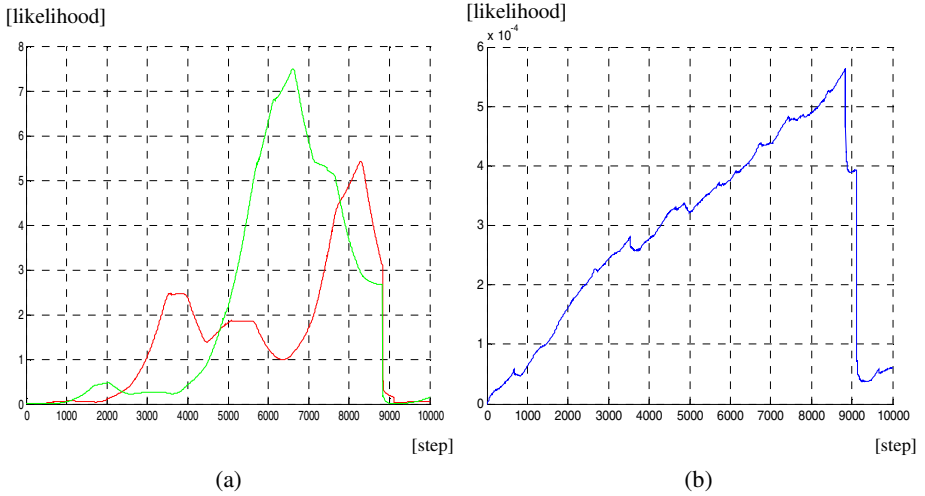
**Fig. 4.** (a) Experimental vehicle and natural terrain environment, and (b) The local side view (x-z axis) of raw point cloud data of one frame scanned by 4ch. LIDAR on the vehicle. The blue points are ground scan points. The red points are tree objects. Green points are the vehicle's position.



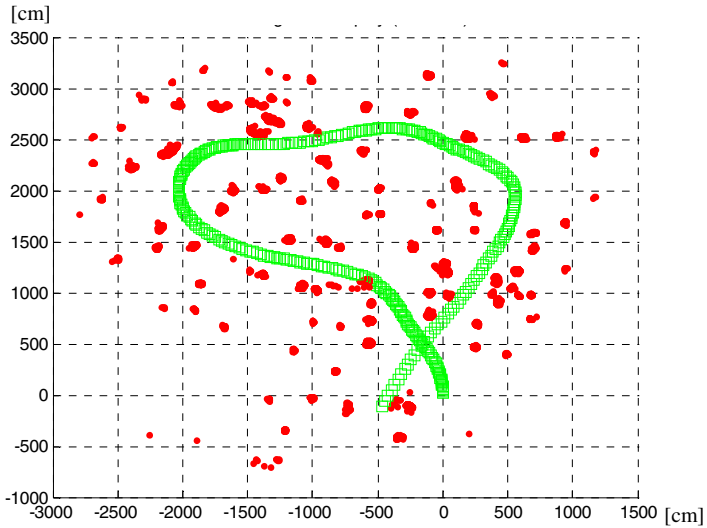
**Fig. 5.** The extracted tree features and corresponding trees on the image in the local top view. (a) In the synchronized camera image, the red arrows describe the detected trees with scanning point cloud data. (b) The red circles represent the extracted tree features. The red line represents the field of view of the scan sensor.



**Fig. 6.** Feature Extraction Results, red marks are the tree features on the Cartesian coordinates, a set of text of the red mark are its identification number. The blue mark is the current scanning object. The green mark is the trajectory of the vehicle.



**Fig. 7.** The likelihood of the vehicle position. (a) the green line is likelihood of the x-axis, the red line is likelihood of the y-axis, and (b) the line is likelihood of the direction. When the loop closing on approx. 8800 steps, the likelihood rapidly falls.



**Fig. 8.** Point cloud feature data for all frames, green color points are the moving path of the vehicle and red color points are feature objects.

## 5 Conclusion

In this paper, we describe an EKF-SLAM method with feature extraction on natural terrain using LIDAR. The LIDAR is a 4-channel device and we process each scan frame. The 4-channel LIDAR is a downward-looking installation on the vehicle and it provides 3D point cloud data. In the case of detection, each discrimination procedure classifies points to significant and non-significant points for SLAM. The proposed method is better than the conventional single fuzzy c-mean clustering and k-mean clustering. In the case of SLAM, the map is composed of previous extracted features and it shows a closed loop. For data association, a significant issue that must be overcome is the presence of various disturbances. In the future, we will increase discriminate classes: trees, bushes, canopy, etc. The terrain influences the vehicle's localization with disturbance and slip. A critical problem of the EKF-SLAM framework is missing correspondence with feature landmarks. To reduce missed correspondence, we should increase the map consistency with map joining skill.

## References

1. Uumson, C., et al.: Autonomous driving in urban environments: Boss and the Urban Challenge. *Journal of Field Robotics-Part I* 25(8), 425–466 (2008)
2. Crane, C., et al.: Team Gator Nation's Autonomous Vehicle Development for the 2007 DARPA Urban Challenge. *Journal of Aerospace Computing, Information, and Communication* 4, 1059–1085 (2007)
3. Joel, A., Dean, E., Terence, S., Mike, A., Michael, O.: Autonomous Navigation of an Unmanned Ground Vehicle in Unstructured Forest Terrain. In: ECSIS Symp. Learning and Adaptive Behaviors for Robotic Systems, LAB-RS 2008, pp. 103–108 (2008)
4. Wellington, C.: Learning a Terrain Model for Autonomous Navigation in Rough Terrain. Ph.D dissertation (2005)
5. Vandapel, N., Huber, D.F., Kapuria, A., Hebert, M.: Natural Terrain Classification using 3-D Ladar Data. In: *Proceeding of IEEE Int. Conf. on Robotics and Automation*, pp. 5117–5112 (2004)
6. Lalonde, J.F., Vandapel, N., Hebert, M.: Data Structure for Efficient Dynamic Processing in 3-D. *International Journal of Robotics Research* 26(8), 777–796 (2007)
7. Cho, K., Baeg, S.H., Park, S.D.: Multiple object detection and tracking on the uneven terrain using multiple lidar for UGV. In: *Proceeding SPIE Defense Sensor and Security*, pp. 26–30 (2012)
8. Fischler, M.A., Bolles, R.C.: Random Sample Consensus: "A paradigm for modelfitting with applications to image analysis and automated cartography". *Communications of the ACM* 24(6), 381–395 (1981)
9. Nascimeto, S., Mirkin, B., Moura-Pires, F.: A Fuzzy Clustering Model of Data and Fuzzy C-means. In: *Proc. 19th IEEE Int. Conf. Fuzzy Syst.*, vol. 1, pp. 302–307 (2000)
10. Pal, N.R., Pal, K., Keller, J.M., Bezdek, J.C.: A possibilistic fuzzy c-means clustering algorithm. *IEEE Trans. Fuzzy Syst.* 13(4), 517–530 (2005)
11. Bailey, T.: Mobile Robot Localisation and Mapping in Extensive Outdoor Environments., Ph.D dissertation (2002)

# Indoor Flight Testing and Controller Design of Bioinspired Ornithopter

Jun-Seong Lee and Jae-Hung Han

Dept. of Aerospace Engineering, KAIST  
291 Daehak-ro, Yuseong-gu,  
Daejeon 305-701, Korea  
{jsl1009, jaehunghan}@kaist.ac.kr

**Abstract.** Indoor flight testing of a bioinspired ornithopter is conducted in this study and the dominant flight state variables such as body pitch angle, forward flight speed, altitude, wings and tail motions of the freely flying ornithopter are simultaneously measured by using a three-dimensional visual tracking system. A control-oriented system model of the ornithopter in trimmed level flight is established based on the recorded inputs and outputs dataset and the system matrices are fitted in a least-squares sense. To reduce the amplitude of the ornithopter body oscillations, the identified linear time-invariant system model is formulated to a disturbance-rejection problem and an optimal controller minimizing the quadratic performance index is designed. The continuous wing motion defined as the known disturbance deteriorates the pitch balance with respect to the center of gravity; however, the designed feedforward and feedback controller periodically activates the ornithopter tail and successfully reduces the magnitudes of the body oscillations.

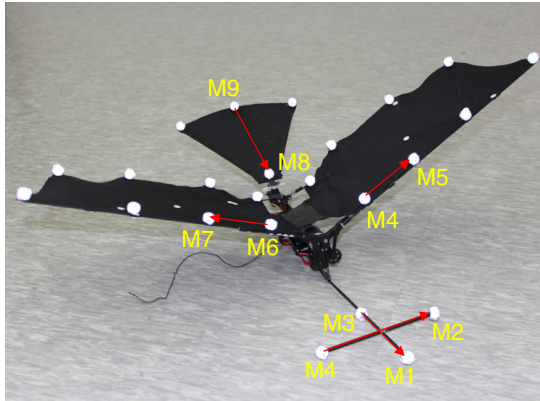
**Keywords:** flapping-wing flight, ornithopter, feedback and feedforward control, visual tracking system, indoor flight testing.

## 1 Introduction

A bioinspired ornithopter is an engineering realization of nature's flyers with insect-like flexible flapping-wings and a bird-like tail. *Nano hummingbird* (Aerovironment, Inc.) and *SmartBird* (Festo, AG & Co. KG) are recently developed to mimic the flight of insects and birds, respectively; however, the ornithopter herein is much close to *Cybird* (Neuros, Co. Ltd) or Kinkade's *Slowhawk* [1-5].

For a trimmed level flight of the ornithopter, the continuous flapping-wing motion generates the positive net thrust and the ornithopter body is accelerated to gain the forward flight speed. The pitching moment induced by aerodynamic loads on tail makes the body have the positive pitch angle, and the flexible wings are aeroelastically deformed and twisted in spanwise direction to have the positive mean angle of attack. Thus, the ornithopter has an aerodynamic lift to compensate its own weight resulting from the lifting surfaces with the positive angle of attack [6-8].

In the cycle-averaged mean sense, the forces and moments around the ornithopter are statically balanced in both translational and rotational directions, and the pre-described wings and tail motions with aeroelastic responses make the ornithopter have a trimmed level flight without active control of the tail. The previous numerical and experimental studies showed that the forward flight ornithopter has distinct trim conditions corresponding to the fixed wingbeat frequencies and tail trim angles [8-9]. In other words, the cycle-averaged mean values of flight state variables are mostly constant during the trim flight without any active feedback control. The aeroelasticity of wings and tail is known to passively reject the external disturbances and sustain the original flight trajectories in longitudinal direction. Limit-cycle oscillation (LCO) is one of the unique trim characteristics of flapping-wing flight [8, 10]. Note that the body movements of hovering flapping-wing flyers are quite different from those of the forward flight ornithopters; the disturbances on body are inherently damped out due to the symmetric and continuous flapping-wing motions in hover [11]. In case of the ornithopters, the existence of such passive damping has not been experimentally found yet.

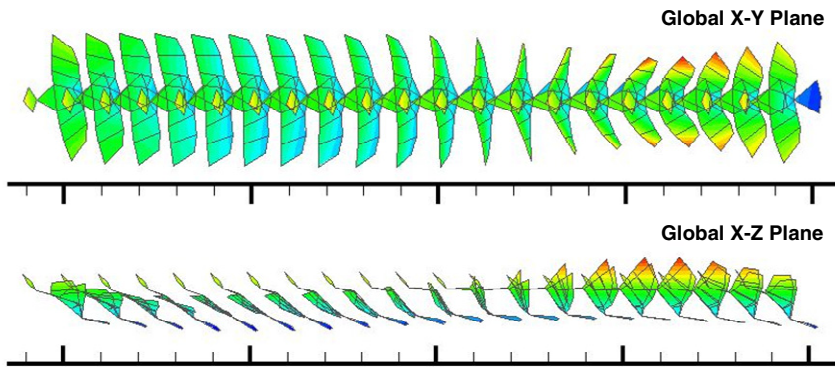


**Fig. 1.** Tested ornithopter platform and the retro-reflective marker placements for visual tracking system

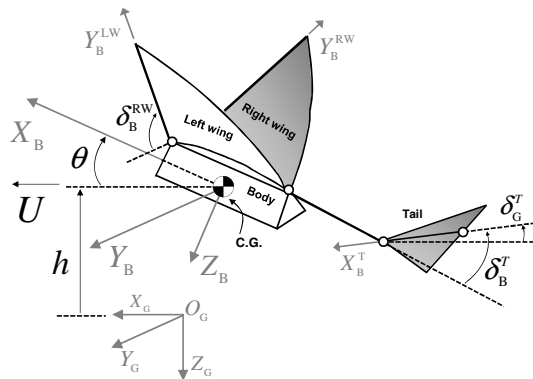
In this study, indoor flight testing of the ornithopter was performed and the flight data were recorded by using a three-dimensional visual tracking system. Based on the acquired inputs/outputs dataset, a control-oriented system model in linear time-invariant form was established. The oscillatory behavior of the ornithopter body somehow enhances the flight stability; however, the oscillations should be minimized to collect the best visual information and prevent the malfunction of other sensitive on-board sensors such as inertial measurement unit, GPS, microprocessor. A linear quadratic controller was designed to reduce the amplitude of the ornithopter body oscillations; the combined feedforward and feedback controller periodically activated the tail so that the body pitching was successfully suppressed.

## 2 Indoor Flight Testing of Bioinspired Ornithopter

KARPE stands for KAIST Arena with Real-time Positioning Environment, which is similar to the RAVEN of MIT ACL [12] and the Micro Air Vehicle Integration and Application Research Institute ( $\mu$ AVIARI) at Wright Patterson Air Force Base [13]. KARPE consists of a three-dimensional visual tracking system with 12 strobe IR cameras (Eagle digital, Motion Analysis Corp.). The cameras are installed in an indoor room capturing a volume of 12 m long, 9 m wide, and 5 m height. The system provides sub-mm spatial resolution with a 12 mm diameter retro-reflective marker (sphere-shaped) and the uncertainty of orientation angles such as body pitch angle, tail elevation angle, wings and tail motion are also examined as 0.5 to 1.0 % of the measured values. Traces of the attached marker sets can be recorded and transmitted almost real-time to the NI's PXIE controller or dSPACE's DSP controller for the feedback control applications [7].



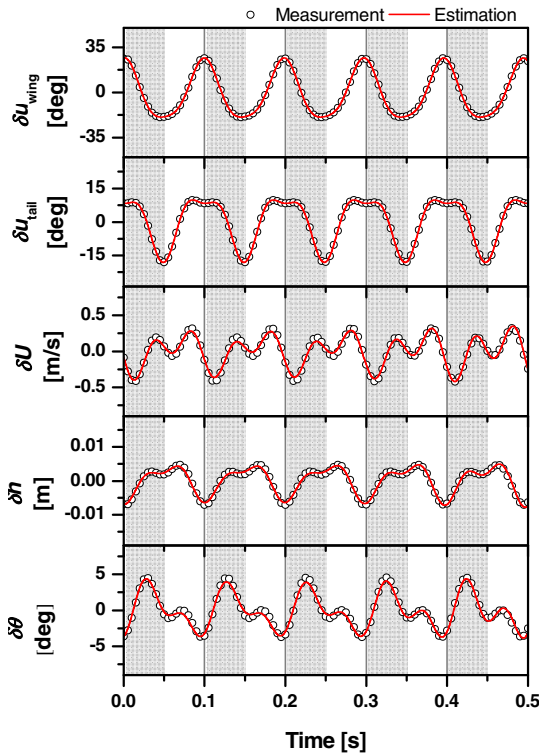
**Fig. 2.** Dorsal and left-side view of freely flying ornithopter at wingbeat frequency 10Hz and forward flight speed 4.2m/s (frame rate of strobe cameras: 200 fps)



**Fig. 3.** Definition of flight state variable and coordinate system for free flight testing of the ornithopter



Fig. 1 shows the ornithopter platform used for the indoor free flight testing; 20 markers were installed to the wings, tail and body. The overall flight behavior including the detailed aeroelastic deformations on wings and tail is illustrated in Fig. 2. Note that the contour color represents the spatial displacement in global Z direction. The visual tracking system is quite promising to obtain the flight state variables without weight penalty and changes of flight dynamics. To construct a control-oriented system model of the freely flying ornithopter, 9 markers (M1~M9) were chosen and the dominant flight state variables such as the forward flight speed  $U$ , the altitude  $h$ , the body pitch angle  $\theta$ , the flapping-wing motion  $\delta_B^{RW} (= u_{wing})$ , and the tail elevation angle  $\delta_B^T (= u_{tail})$  were calculated from the vectors defined in Figs. 1 and 3. To achieve the straight and level flight in KARPE, the ornithopter was hand-launched with the fixed wingbeat frequency of 10 Hz and the tail elevation angle of  $33^\circ$ . After the several times of free flight trials, the cycle-averaged mean of the dominant flight state variables were found as: the mean forward flight speed of 4.2m/s, the mean body pitch angle of  $47.3^\circ$ , and the mean altitude of 1.97m.



**Fig. 4.** Time history of zero-mean time-varying dominant flight state variables of freely flying ornithopter (Symbol: measurement data, Solid line: Estimation results)

The ornithopter flight dynamics can be separated into longitudinal and lateral planes; it can also be categorized as the cycle-averaged mean flight dynamics and the zero-mean, time-varying flight dynamics, as described in Ref. [14]. Thus, the measured dataset were offset by the cycle-averaged means and the zero-mean, time-varying flight state variables are prepared for identifying the control system model as in Fig. 4. More details of flight dynamic characteristics of the freely flying ornithopter can be found in Ref. [7].

### 3 System Identification and Controller Design of Bioinspired Ornithopter

The cycle-averaged mean flight dynamics of the forward flight ornithopter is determined by the constant values of wingbeat frequency and the tail elevation angle. As explained earlier, the mean flight dynamics of the ornithopter shows a stable LCO, believed to be partially induced by the aeroelasticity of wings and tail. This paper focuses more on the zero-mean, time-varying flight dynamics than the cycle-averaged one because the suppression of body oscillation is the control objective. In the previous numerical study on the suppression of the time-varying dynamics [14], the prediction error method was introduced to model the forward flight ornithopter. The modeling error and sensor noise were considered as the Gaussian noise and the linear-quadratic Gaussian regulator was applied to construct the feedback controller. Herein, the system matrices were fitted in least-squares sense and due to the advantage of the visual tracking system, all of the interested state variables were measured with a negligible magnitude of noise and a reasonable sampling rate.

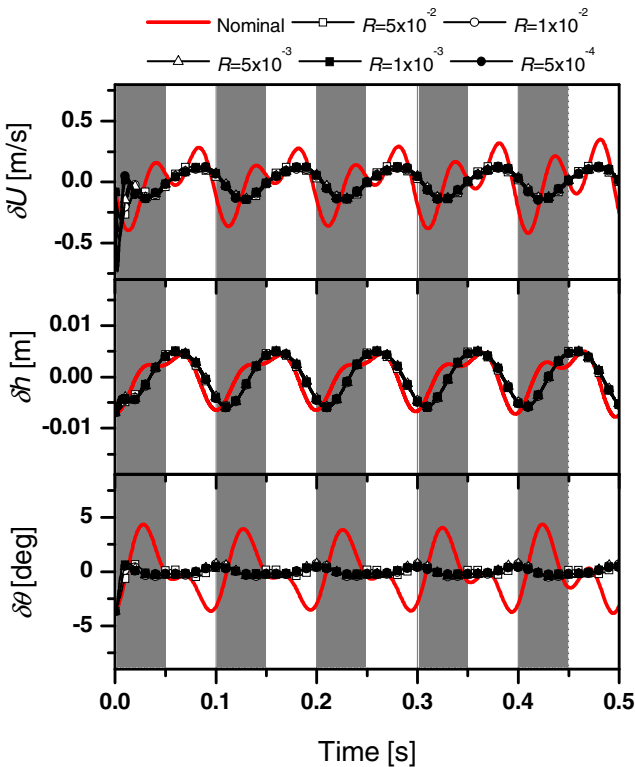
Let the identified system model have the form:

$$\delta \mathbf{x}_{3 \times 1}[k+1] = \hat{\Phi}_{3 \times 3} \delta \mathbf{x}_{3 \times 1}[k] + \hat{\Gamma}_{3 \times 2} \delta \mathbf{u}_{2 \times 1}[k] \quad (1)$$

where

$$\begin{aligned} \delta \mathbf{x}_{3 \times 1}[k] &= [\delta U[k] \quad \delta h[k] \quad \delta \theta[k]]^T \\ \delta \mathbf{u}_{2 \times 1}[k] &= [\delta u_{\text{wing}}[k] \quad \delta u_{\text{tail}}[k]]^T \\ \hat{\Phi}_{3 \times 3} &= \begin{bmatrix} +9.96 \times 10^{-1} & 3.27 \times 10^0 & 2.55 \times 10^{-3} \\ -7.07 \times 10^{-4} & 9.85 \times 10^{-1} & 2.17 \times 10^{-6} \\ -5.29 \times 10^{-1} & 1.93 \times 10^{+1} & 9.91 \times 10^{-1} \end{bmatrix} \\ \hat{\Gamma}_{3 \times 2} &= \begin{bmatrix} \hat{\gamma}_{3 \times 1}^{\text{wing}} & \hat{\gamma}_{3 \times 1}^{\text{tail}} \end{bmatrix} \\ &= \begin{bmatrix} +2.89 \times 10^{-4} & -6.17 \times 10^{-6} & +4.61 \times 10^{-3} \\ +4.99 \times 10^{-4} & -6.84 \times 10^{-7} & -8.46 \times 10^{-5} \end{bmatrix}^T \end{aligned} \quad (2)$$

The responses of the identified system are plotted in Fig. 4 (red solid line) with the measured data (symbol). The time histories of two dataset were sufficiently similar to each other. The eigenvalues of  $\hat{\Phi}_{3 \times 3}$  indicated that the identified system was stable and the controllability and observability were also satisfied. For a given system model, various control methods can be applied to achieve the desired performance of the control system [15-18]. In this study, the disturbance-rejection problem was formulated by using the identified system model in Eq. (1) while the flapping-wing motion was considered as the known disturbance and the tail was the only control input of the control system. Even the wing motion deteriorates the body dynamics, flapping-wing motion should be continued during the trimmed level flight. Due to the limited degrees of freedom, the modification of wing kinematics is very difficult for the ornithopter. Thus, active control of tail motion is quite natural rather than the change of wing motion during the trim flight.



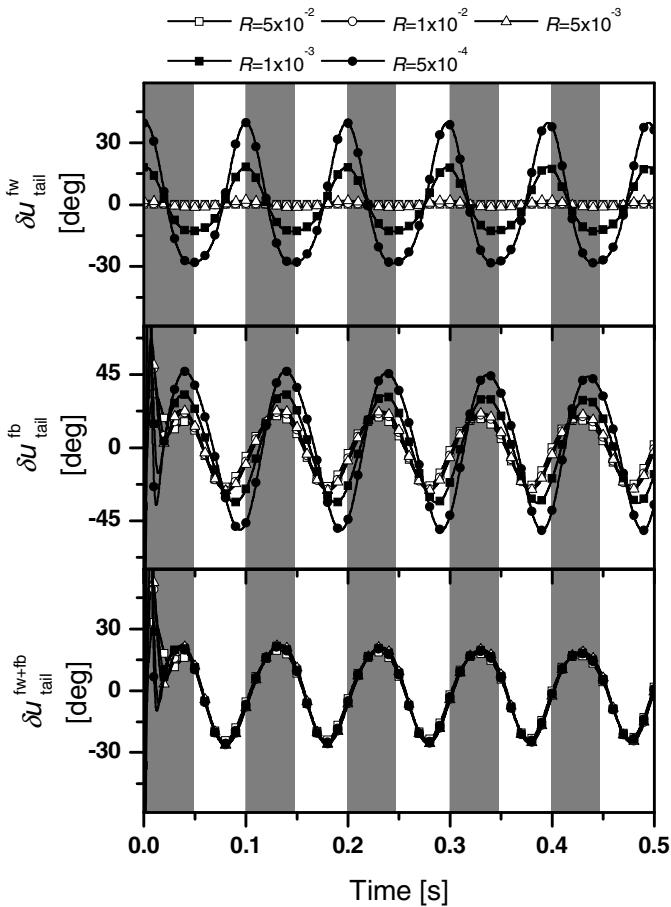
**Fig. 5.** Time histories of the suppressed zero-mean time-varying dominant flight state variables (Simulation results, Symbol: control activated cases, Solid line: nominal system, control deactivated case)

Let  $w[k]$  is the known disturbance of the system and the input matrix  $\hat{\Gamma}_{3 \times 2}$  in Eq. (2) is divided as in Eq. (3).

$$\delta \mathbf{x}_{3 \times 1}[k+1] = \hat{\Phi}_{3 \times 3} \delta \mathbf{x}_{3 \times 1}[k] + \hat{\gamma}_{3 \times 1}^{\text{tail}} \delta u_{\text{tail}}^*[k] + \underbrace{\hat{\gamma}_{3 \times 1}^{\text{wing}} \delta u_{\text{wing}}[k]}_{w[k]} \quad (3)$$

To reduce the magnitude of the oscillations, the performance index to be minimized is defined in Eq. (4).

$$J = \int_0^{\infty} (\delta \mathbf{x}_{3 \times 1}^T Q_{3 \times 3} \delta \mathbf{x}_{3 \times 1} + \delta u_{\text{tail}}^T R_{1 \times 1} \delta u_{\text{tail}}) d\tau \quad (4)$$



**Fig. 6.** Time histories of the feedforward, feedback and combined feedforward and feedback control inputs of tail (Simulation results)

Applying the linear quadratic control framework [19], the optimal control solution of the disturbance-rejection problem consists of feedforward control,  $\delta u_{tail}^{fw}$  and feedback control,  $\delta u_{tail}^{fb}$ .  $\delta u_{tail}^{fw}$  is the control effort on the known disturbance rejection and  $\delta u_{tail}^{fb}$  corresponds to the state regulation via full-state feedback (Eqs. (5) and (6)).

$$\delta u_{tail}^* = \delta u_{tail}^{fw} + \delta u_{tail}^{fb} \tag{5}$$

where

$$\begin{aligned} \delta u_{tail}^{fb} &= -K_{LQR} \delta \mathbf{x}_{3d} = -\{R_{bd}^{-1} (\hat{\gamma}_{3d}^{tail})^T \bar{P}_{3d}\} \delta \mathbf{x}_{3d} \\ \delta u_{tail}^{fw} &= -R_{bd}^{-1} (\hat{\gamma}_{3d}^{tail})^T \text{Re}[-\{\hat{\Phi}_{3d} - \hat{\gamma}_{3d}^{tail} K_{LQR} + j\omega \mathbf{I}_{3d}\}^{-1}] \bar{P}_{3d} w[k] \end{aligned} \tag{6}$$

For the steady-state case, the matrix Riccati equation in Eq. (7) has time-invariant solution,  $\bar{P}_{3d}$  and is easily calculated for obtaining the optimal control gain,  $K_{LQR}$ .

$$\hat{\Phi}_{3d}^T \bar{P}_{3d} + \bar{P}_{3d} \hat{\Phi}_{3d} + Q_{3d} - \bar{P}_{3d} \hat{\gamma}_{3d}^{tail} R_{bd}^{-1} (\hat{\gamma}_{3d}^{tail})^T \bar{P}_{3d} = 0 \tag{7}$$

The weighting matrices,  $Q_{3d}$  and  $R_{bd}$  were tuned to achieve the reduction of states  $\delta \mathbf{x}_{3d}$  and the values were finally found as Eq. (8). The range of tail motion was adjusted by the five different weighting values of  $R_{bd}$  ( $= R_j, j = 1 \sim 5$ ).

$$\begin{aligned} Q_{3d} &= \text{diag}([10 \quad 10 \quad 20]) \\ R_{bd} &= 5 \times 10^{-2}, 1 \times 10^{-2}, 5 \times 10^{-3}, 1 \times 10^{-3}, 5 \times 10^{-4} \end{aligned} \tag{8}$$

As shown in Fig. 5, the oscillatory behaviors of the states were successfully minimized and 86% of the suppression of body pitch angle,  $\delta \theta$  was found at  $R_{bd} = 5 \times 10^{-4}$ . Table 1 summarizes the control results in the root-mean-squared (RMS) sense and the time history of feedforward, feedback and combined of both control inputs were shown in Fig 6. The changes of the weighting value,  $R_{bd}$  can be used to adjust the portion of the control efforts between feedforward and feedback controls; however, the combined control efforts,  $\delta u_{tail}^{fw+fb}$ , were almost the same regardless of the variation in  $R_{bd}$  as shown in Fig 6. Note that the tail motion in Figs. 4 and 6 has different phase angles with respect to the wing motion and the simple sweeping of the tail motion might be helpful to reduce the magnitude of body oscillations of the ornithopter [15].

**Table 1.** Summary of suppression of flight state variables

	<i>Nominal</i>	$R_1$	$R_2$	$R_3$	$R_4$	$R_5$
$\delta U$ [m/s]	0.20			0.10		
$\delta h$ [m]				3.8		
$\delta \theta$ [deg]	2.31	0.45	0.49	0.48	0.39	0.33

## 4 Conclusion

An indoor flight testing of a bioinspired ornithopter was conducted at KARPE. The dominant flight state variables of the freely flying ornithopter were measured and the control-oriented system model of the ornithopter in trimmed level flight was established in the least-squares sense. By applying the linear-quadratic control framework, the oscillation of the body pitch angle due to the continuous wing motion was successfully minimized by introducing the active control of the tail. The resultant control input of tail was quite different from the one of the nominal system; it had a dominant single frequency, periodic shape and a certain phase angle with respect to the wing motion. In future work, the designed controller is going to be implemented in the same environment, KARPE and the automation of the indoor ornithopter flight will be experimentally made.

**Acknowledgements.** This work was supported by Mid-career Researcher Program (2011-0015569) through NRF grant funded by the MEST. The first author would like to thank the Brain Korea 21 Project of 2012.

## References

1. Hylton, T., Martin, C., Tun, R., Castelli, V.: The DARPA Nano Air Vehicle Program. In: Proc. of 50th AIAA Aerospace Science Meeting, Nashville, TN, AIAA 2012-0583, January 06-12 (2012)
2. Keennon, M., Klingebiel, K., Won, H., Audriukov, A.: Development of the Nano Hummingbird: A Tailless flapping wing micro air vehicle. In: Proc. of 50th AIAA Aerospace Science Meeting, Nashville, TN, AIAA 2012-0588, January 06-12 (2012)
3. Festo, AG & Co. KG, Brochure of SmartBird (April 2011), <http://www.festo.com/>
4. Han, J.-H., Lee, J.-S., Kim, D.-K.: Bio-inspired flapping UAV design: A university perspective. In: Proc. SPIE, vol. 7295, pp. 729511–7295112 (2009)
5. Park, J.H., Yoon, K.-J.: Designing a biomimetic ornithopter capable of sustained and controlled flight. *J. Bionic Eng.* 7(1), 39–47 (2008)
6. Pfeiffer, A.T., Lee, J.-S., Han, J.-H., Baier, H.: Ornithopter flight simulation based on flexible multi-body dynamics. *J. Bionic Eng.* 7(1), 102–111 (2010)
7. Lee, J.-S., Han, J.-H.: Experimental study on the flight dynamics of bioinspired ornithopter: Free flight testing and wind tunnel testing. *Smart Mater. Struct.* 21(9), Article No. 094023 (11pp) (2012)
8. Lee, J.-S., Kim, J.-K., Kim, D.-K., Han, J.-H.: Longitudinal flight dynamics of bioinspired ornithopter considering fluid-structure interaction. *J. Guid. Control. Dynam.* 34(3), 667–677 (2011)
9. Grauer, J., Hubbard Jr., J.: Inertial measurements from flight data of a flapping-wing ornithopter. *J. Guid. Control. Dynam.* 32(1), 326–331 (2009)
10. Kim, J.-K., Lee, J.-S., Han, J.-H.: Passive longitudinal stability of ornithopter flight. *J. Guid. Control. Dynam.* 35(2), 669–673 (2012)
11. Cheng, B., Deng, X.: Translational and rotational damping of flapping flight and its dynamics and stability at hovering. *IEEE T. Robot.* 27(5), 849–864 (2011)

12. How, J., Bethke, B., Frank, A., Dale, D., Vian, J.: Real-time indoor autonomous vehicle test environment. *IEEE Contr. Syst. Mag.* 28(2), 51–64 (2008)
13. Maniar, G., Randall, R., Shkarayev, S.: Kinematics of Free-flight Ornithopters. In: *Proc. of 50th AIAA Aerospace Science Meeting*, Nashville, TN, AIAA 2012-0666, January 06-12 (2012)
14. Kim, J.-K., Lee, J.-S., Han, J.-H.: Limit-cycle oscillation suppression of ornithopter longitudinal flight dynamics. In: *Proc. of AIAA GNC Conference*, Portland, OR, AIAA 2011-6404, August 08-11 (2011)
15. Lee, J.-S., Kim, J.-K., Han, J.-H., Ellington, C.P.: Periodic tail motion linked to wing motion affects the longitudinal stability of ornithopter flight. *J. Bionic. Eng.* 9(1), 18–28 (2012)
16. Grauer, J., Hubbard Jr., J.: Modeling of ornithopter flight dynamics for state estimation and control. In: *Proc. of America Control Conf.*, Baltimore, MD, June 30 –July 02 (2010)
17. Dietl, J., Garcia, E.: Ornithopter control with periodic infinite horizon controllers. *J. Guid. Control. Dynam.* 34(5), 1412–1421 (2011)
18. Krashanitsa, R., Silin, D., Shkarayev, S., Abate, G.: Flight dynamics of a flapping-wing air vehicle. *Int. J. MAV.* 1(1), 35–49 (2009)
19. Dorato, P., Abdallah, C.T., Cerone, V.: *Linear-Quadratic Control: An Introduction*, pp. 43–62. Prentice-Hall, Inc. (1995)

# Effect of Passive Body Deformation of Hawkmoth on Flight Stability

Ryusuke Noda, Masateru Maeda, and Hao Liu

Graduate School of Engineering, Chiba University  
1-33 Yayoi-cho, Inage-ku,  
Chiba 263-8522, Japan  
hliu@faculty.chiba-u.jp

**Abstract.** In this study, the effect of passive body deformation on flight stability during insect flapping flight is investigated numerically. We developed a flexible body dynamic solver for a three-dimensional flexible beam model and coupled it with an *in-house* fluid dynamics solver. With this integrated model, hawkmoth free flights are simulated and analyzed systematically with six cases, in which the joint stiffness between thorax and abdomen varied from extremely rigid to very flexible. Our results indicate that the passive body deformation works likely altering the aerodynamic torque, the body attitude and the flight trajectory. We further found that the most stable flight can be achieved by a moderate joint stiffness, in which the body attitude remains approximately around the initial angle of 40 degree. This points to the importance that the flexible body and its passive deformation during flapping-wing flight are capable to enhance stable flight and flight control.

**Keywords:** insect flight, flexible body dynamics, stability.

## 1 Introduction

The effect of wing deformation of insects on their flight ability has attracted attentions from researchers in either biology [1-3] or engineering [4-7]. The work on the body deformation, however, is rare [8, 15], and with our knowledge, the work on the passive body deformation in terms of flight stability or performance is not published. One of the difficulties in treating flexibility in flying insects is that fluid dynamics and passive structural deformation couple each other: fluid force makes the shape of the body changed; deformed body generates altered fluid force. We developed a flexible body dynamics solver where insect body is modeled as a three dimensional beam. By coupling the solver with an existing fluid dynamics solver, numerical simulation of a hovering hawkmoth with flexible body is achieved. We tested several values of elasticity (from rigid to very flexible) for the joint between thorax and abdomen and saw how it affects aerodynamic force/torque generation and especially history of the flier's attitude. The result showed that in the most flexible case the body angle increased 45 degrees in five wingbeat cycles, while in the stiffer cases the trends were similar to that in the rigid case.



## 2 Method

### 2.1 Insect Model

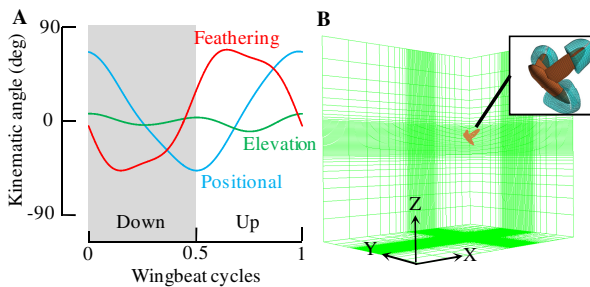
Nakata [9] showed that the larger the insect, the smaller the wing dimensionless stiffness. It is not so unnatural to assume this trend also applies for the body. Therefore we used a hawkmoth as a model insect in the current study, where large body deformation compared to the smaller insects is expected, thus suitable for assess its effect. The morphological and wing kinematics models (Fig. 1) are the same as those in Aono *et al.* [10].

### 2.2 Computational Fluid Dynamics

For the flow field computation around insect body and wings, we used a computational fluid dynamics solver developed by Liu [11]. In the present study the global (background) grid is replaced with the Cartesian structured grid and the body grid block is now a local grid. The major parameters and the grid used in the fluid dynamics computation are shown in Table 1 and Fig. 1, respectively. The number of grid used ( $i \times j \times k$ ) are: background grid,  $61 \times 61 \times 57$ ; body grid,  $37 \times 37 \times 9$ ; each wing grid,  $37 \times 33 \times 11$ .

**Table 1.** Parameters for fluid dynamics simulation

Reynolds number	Re	5400
Reduced frequency	k	0.35
Mean chord length	$c_m$	18.6 mm
Reference velocity	$U_{ref}$	4.34 m/s
Wingbeat frequency	$f$	26.0 Hz
Initial thorax angle (body angle if rigid body)	$\beta_{ini}$	40.0 deg



**Fig. 1.** Hawkmoth model. (A) Wing kinematics. (B) Computational grid for fluid dynamics simulation with the global coordinates

### 2.3 Flexible Body Dynamics

In order to reproduce the body deformation, we developed a flexible body dynamics solver, where three dimensional flexible beam with 41 elements takes the load (Figs. 2A and 2B). The deformation of the beam is solved with the finite element method (FEM). Note that in the present computation, we only considered the passive body deformation and not the active body deformation which is generated by its muscle. The aerodynamic force acting on the body itself is too small to provoke the deformation. Rather, the aerodynamic forces on the wings are introduced to the two wing base nodes, becoming the main source for the body deformation. The governing equations of FBD solver is written as

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \mathbf{K}_L\mathbf{q} + \mathbf{f}_{\text{int}}(\mathbf{q}) - \mathbf{f}_{\text{ext}} = \mathbf{0} \quad (1)$$

where  $\mathbf{M}$  is the lumped mass matrix;  $\mathbf{C}$  is the damping matrix;  $\mathbf{K}_L$  is the linear stiffness matrix;  $\mathbf{f}_{\text{int}}$  is the nonlinear restoration force;  $\mathbf{f}_{\text{ext}}$  is the external force;  $\mathbf{q}$  is the positions and orientations in the global frame. Note that in this problem, each element changes the position not only by deformations but by whole-body motion. Therefore the restoration force is expressed as a nonlinear function of the displacement. For the integration, we use Newmark- $\beta$  scheme for nonlinear problems. The tangent stiffness matrix  $\mathbf{K}_N$  is required in the formulation, which is expressed as the partial derivative of  $\mathbf{f}_{\text{int}}$  as

$$\mathbf{K}_N = \partial\mathbf{f}_{\text{int}}/\partial\mathbf{q}. \quad (2)$$

The  $\mathbf{K}_N$  can be divided into two parts as

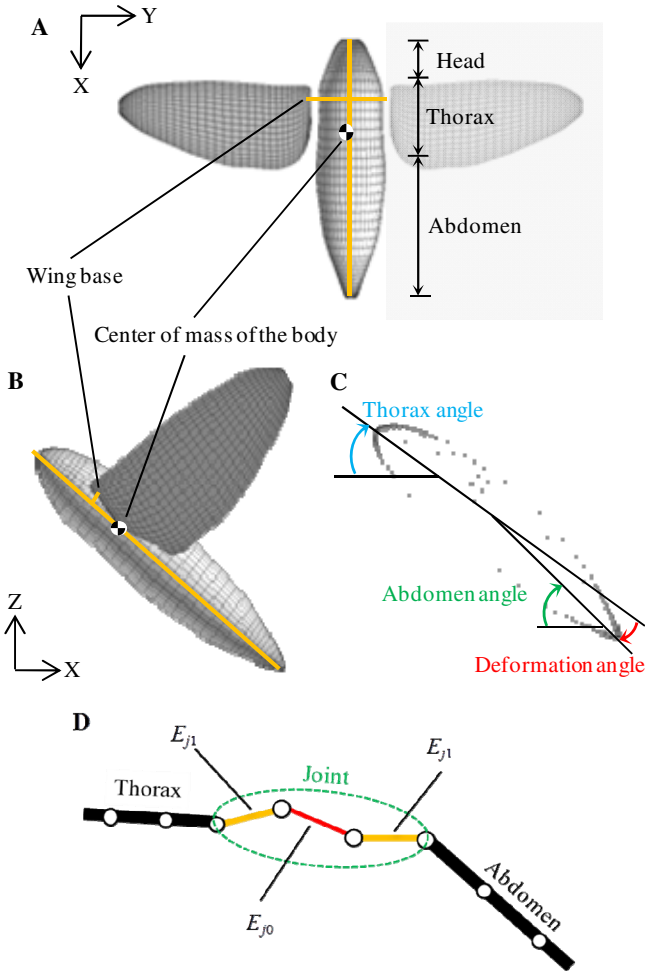
$$\mathbf{K}_N = \mathbf{K}_G + \mathbf{K}_O \quad (3)$$

where  $\mathbf{K}_G$  is the geometric stiffness matrix and  $\mathbf{K}_O$  is the elastic deformation matrix. [12] The flexible body dynamics solver developed was validated with both fixed-end and free-end flexible beam benchmark problems [13].

The fluid dynamics solver and the flexible body dynamics solver need to exchange information to achieve the computation. This process is called coupling and we used the following manner [14]. (1) Predict the displacement of each node at time  $t_{n+1}$  from the information (displacement, velocity and acceleration) at time  $t_n$ ; (2) Update the position of the fluid grid based on the increments of each node; (3) Transfer fluid force to the structure; and (4) advance the time in the FBD solver.

The major parameters used in the flexible body dynamics computation are shown in Table 2 and 3. We have chosen the location for the center of mass of the body as follows. The mass of each block (head, thorax, abdomen, and wings) was assigned according to [15]. Next, the density distribution to the nodes in each block was determined so that the rigid case (described below) can maintain relatively stable hovering. Therefore caution has to be taken that the density distribution may not reflect that of the real hawkmoth. In order to see the effect of the different body flexibility on the flight dynamics, several different Young's moduli at the three joint elements between thorax and abdomen ( $E_{j0}$  and  $E_{j1}$ , where  $E_{j1} := 2E_{j0}$ . see Fig. 2D) were chosen. The Young's modulus  $E_{j0} = 10^{11}$  pascals was selected to represent a

rigid body model. In addition,  $E_{j0} = 10^7, 10^5, 5 \times 10^4, 4 \times 10^4,$  and  $3 \times 10^4$  pascals were tested. The values of  $E_{j0}$  were determined so that the resultant deformation angles (Fig. 2C) fall within an experimental value of approximately five degrees [8]. In addition, we assumed a uniform Young's modulus distribution across each cross section.



**Fig. 2.** Flexible body model. (A) and (B) A flexible beam (orange) is placed in the middle of the insect grid which is used for the fluid dynamics simulation (gray). Note the grid is also used for calculating the moment of inertia. (C) Definitions of thorax, abdomen, and deformation angles. Note thorax and abdomen angles are measured from the horizon. (D) Schematic of the structural elements around the thorax-abdomen joint. Young's modulus of the middle joint element is termed  $E_{j0}$ ; each side joint element is  $E_{j1}$ , defined as  $E_{j1} := 2E_{j0}$ . All the other elements are given the same (high) Young's modulus.

**Table 2.** Mass of the hawkmoth model [15]

Body mass		
head	$m_h$	107 mg
throat	$m_t$	596 mg
abdomen	$m_a$	818 mg
Wing mass (one wing)	$m_w$	35 mg

**Table 3.** Number of nodes and Young's modulus for each structural block

	Number of elements	Young's modulus (Pa)
Head	8	$10^{11}$
Thorax	7	$10^{11}$
Joint between thorax and abdomen	3	* $E_{jo}, E_{j1}$
Abdomen	21	$10^{11}$
Left wing base	1	$10^{11}$
Right wing base	1	$10^{11}$

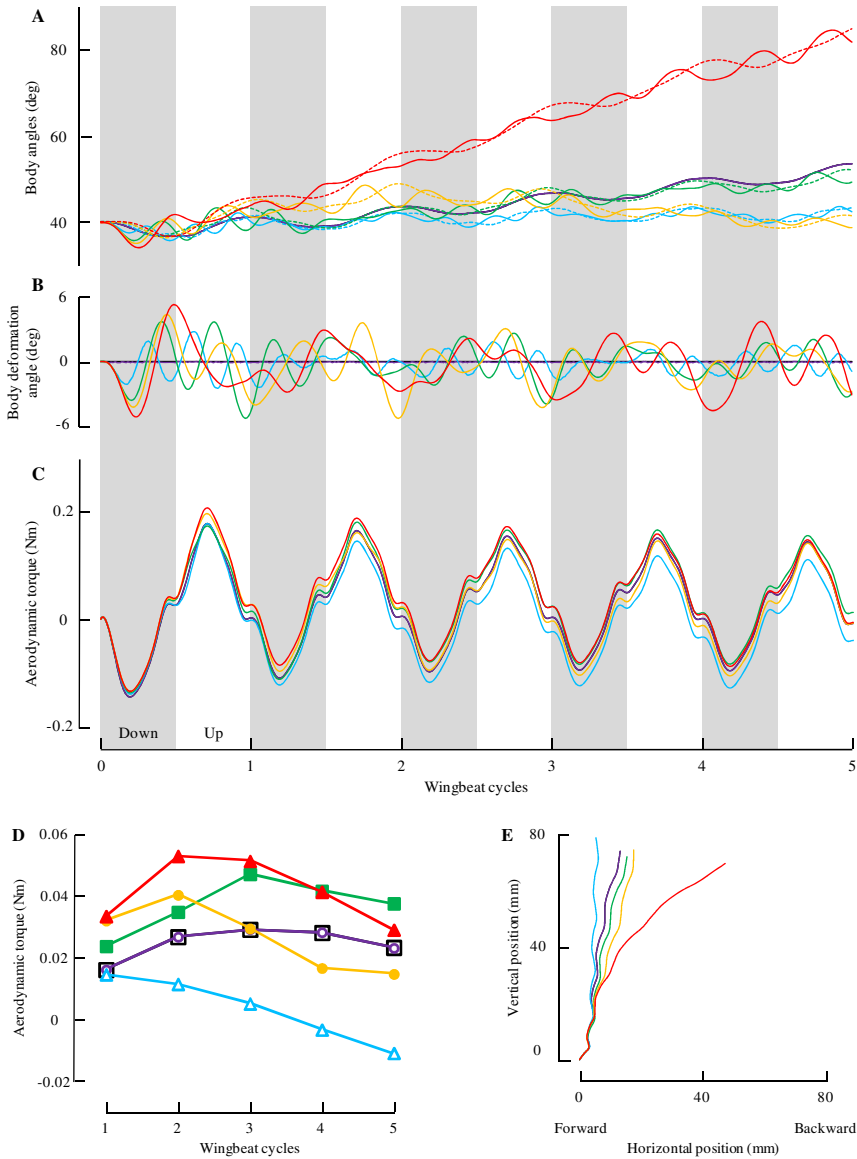
\* See the description in the main text and Fig. 2.

### 3 Results

Our simulation (summarized in Fig. 3) shows that passive body deformation alters aerodynamic torque, body attitude and flight trajectory. There was almost no difference between the most rigid two cases ( $E_{jo} = 1 \times 10^{11}$  Pa and  $E_{jo} = 1 \times 10^7$  Pa). The next rigid case ( $E_{jo} = 1 \times 10^5$  Pa) performed the most stable flight, where body attitude remained around initial angle of 40 degrees (Fig. 3A) while horizontal deviation was minimal (Fig. 3E). The most flexible case ( $E_{jo} = 3 \times 10^4$  Pa) exhibited the most unstable flight, where body attitude reached 80 degrees nose-up while the body moved backward. The intermediate flexibility cases ( $E_{jo} = 4 \times 10^4$  Pa and  $E_{jo} = 5 \times 10^4$  Pa) showed moderately stable trends of body attitudes and flight trajectories, similar to those of more rigid cases. The body deformation angle (Fig. 3B) is calculated by subtracting abdomen angle (dotted line in Fig. 3A) from thorax angle (solid line in Fig. 3A).

### 4 Discussion

Although the link is not sufficiently clear, from our simulation, especially from the result of the most flexible case ( $E_{jo} = 3 \times 10^4$  Pa, red lines in Fig. 3) it is assumed that the body deformation (Fig. 3B) leads to the change in the aerodynamic torque



**Fig. 3.** Time series of simulated flight. (A) Thorax angle (solid lines) and abdomen angle (dotted lines). (B) Body deformation angle. (C) Y component (pitching) of aerodynamic torque around body center of mass. (D) Y component (pitching) of wingbeat cycle-averaged aerodynamic torque around body center of mass. (E) Location of body center of mass in the XZ-plane, starting from origin at time  $t = 0$ . Key to colors: red, Young's modulus of the joint between throax and abdomen  $E = 3 \times 10^4$  Pa; orange,  $E = 4 \times 10^4$  Pa; green,  $E = 5 \times 10^4$  Pa; blue,  $E = 1 \times 10^5$  Pa; purple,  $E = 1 \times 10^7$  Pa; black,  $E = 1 \times 10^{11}$  Pa.

(Figs. 3C and 3D), which determines the overall body attitude (Fig. 3A) and flight trajectory (Fig. 3E). One possible story is that the reduction in the moment of inertia in the  $y$ -(pitching) axis due to the body deformation led to the increase in the pitching angular velocity, thus increased wing velocity and aerodynamic force, and finally aerodynamic torque. Another possibility includes the reaction of the abdomen rotation revolved the thorax, thus wing base, in the opposite direction of the abdomen motion, which resulted in the increase in the wing velocity. Since there are several factors affecting the aerodynamic force generation, some manipulation may be necessary to separate out which has the largest impact. For example, preparative computation for achieving the “trimmed hovering” by choosing the appropriate wing kinematics by the aid of optimization [16] or via flight control (Gao Na, unpublished work) would be useful.

In addition, we have also tested the effect of the location of the body center of mass (not shown), where it was revealed that the relative location of the body center of mass against wing bases has a large impact on the body attitude and flight trajectory. For the insect flight research, this means the precise measurement of the body center of mass is inevitable. For a flapping-wing micro air vehicle (fMAV) without tail (like insects rather than birds), this means that the designer needs to carefully choose the location of the body center of mass.

**Acknowledgements.** R. N. thanks Toshiyuki Nakata for checking and commenting on the flexible body dynamics program.

## References

1. Combes, S.A., Daniel, T.L.: Into thin air: contributions of aerodynamic and inertial-elastic forces to wing bending in the hawkmoth *Manduca sexta*. *J. Exp. Biol.* 206, 2999–30006 (2003)
2. Zhao, L., Deng, X., Sane, S.P.: Modulation of leading edge vorticity and aerodynamic forces in flexible flapping wings. *Bioinsp. Biomim.* 6, 036007 (2011)
3. Nakata, T., Liu, H.: Aerodynamic performance of a hovering hawkmoth with flexible wings: a computational approach. *Proc. R. Soc. B* 279, 722–731 (2011)
4. Shyy, W., Aono, H., Chimakurthi, S.K., Trizila, P., Kang, C.-K., Cesnik, C.E.S., Liu, H.: Recent progress in flapping wing aerodynamics and aeroelasticity. *Prog. Aerospace Sci.* 46, 284–327 (2010)
5. Nakata, T., Liu, H., Tanaka, Y., Nishihashi, N., Wang, X., Sato, A.: Aerodynamics of a bio-inspired flexible flapping-wing micro air vehicle. *Bioinspir. Biomim.* 6, 045002 (2011)
6. Mahardika, N., Viet, N.Q., Park, H.C.: Effect of outer wing separation on lift and thrust generation in a flapping wing. *Bioinsp. Biomim.* 6, 036006 (2011)
7. Tanaka, H., Shimoyama, I.: Forward flight of swallowtail butterfly with simple flapping motion. *Bioinsp. Biomim.* 5, 026003 (2010)
8. Hinterwirth, A.J., Daniel, T.L.: Antennae in the hawkmoth *Manduca sexta* (Lepidoptera, Sphingidae) mediate abdominal flexion in response to mechanical stimuli. *J. Comp. Physiol. A* 196, 947–956 (2010)
9. Nakata, T.: Simulation-based study on aerodynamic performance of flexible flapping wings. Chiba University PhD thesis (2012)

10. Aono, H., Shyy, W., Liu, H.: Near wake vortex dynamics of a hovering hawkmoth. *Acta Mech. Sin.* 25, 23–36 (2009)
11. Liu, H.: Integrated modeling of insect flight: From morphology, kinematics to aerodynamics. *J. Comput. Phys.* 228, 439–459 (2009)
12. Chan, S.L., Chui, P.P.T.: *Non-Linear Static Analysis of steel Frames with Semi-Rigid Connections*. Elsevier Science Ltd., London (2000)
13. Simo, J.C., Vu-Quoc, L.: On the dynamics in space of rods undergoing large motions - A geometrically exact approach. *Comput. Methods Appl. Mech. Eng.* 66, 125–161 (1988)
14. Piperno, S., Farhat, C.: Partitioned procedures for the transient solution of coupled aeroelastic problems - part ii: energy transfer analysis and three-dimensional applications. *Comput. Methods Appl. Mech. Eng.* 190, 3147–3170 (2001)
15. Hedrick, T.L., Daniel, T.L.: Flight control in the hawkmoth *Manduca sexta*: the inverse problem of hovering. *J. Exp. Biol.* 209, 3110–3114 (2006)
16. Wu, J.H., Zhang, Y.L., Sun, M.: Hovering of model insects: simulation by coupling equations of motion with Navier-Stokes equations. *J. Exp. Biol.* 212, 3313–3329 (2009)

## **Chapter III**

# **Service Robots for Home and Healthcare**

Recently the service robots are coming near to our daily lives, such as, home assistant and health care robots. Depending on the service, the mechanical design of the service robot is optimized and implemented. A robot manipulator with five degrees of freedom for hand-over-hand guidance training of laparoscopic surgery is presented [1]. Probabilistic decision making for multi-modal service robots from human demonstrations is an interesting approach for the service robot implementation [2].

There are several trials for the surgical robots: The ARAKNES platform has been introduced as a case study of safety in the design of surgical robot [3]. Master console haptic handle using cable conduit mechanism for robot assisted laparoscopy is designed to overcome the deficiency of visual feedback [4]. For microsurgery, fusion of inertial measurements and vision feedback has been proposed. A high speed monovision camera on an optical surgical microscope and acceleration measurements from an intelligent handheld instrument are utilized to provide real-time enhanced micrometer scale positioning for a microsurgery [5]. The design concept of a novel vaginal hysterectomy robot which is composed of three compound robots is discussed with two prototypes and ideas for future development [6].

For the service robots, the recognition and decision are critical factors for the safe and convenient interactions to humans. To improve the patient safety, relevant applications of simulations are introduced and classified for training and robot design usages [7]. Also for the patient safety in the robotic surgery, role of holographic and stereovision displays is presented for the high quality 3D images [8]. Through the experience of SAFROS project, a methodology frame work for the definition of patient safety measures in robotic surgery has been demonstrated during the pre- and intra-operative phase of robotic interventions [9].

Visual and tactile sensors are very widely utilized for the service robots, even though there are many research issues in both of the two sensors. For the working space of the robot system is overlapped with the patient and the surgical personnel, a partially autonomous system with the guaranteed collision-free path-planning with a real-time monitoring system is proposed [10]. Intention sharing of assistant robot for elderly people has been discussed to improve the interactive cooperation between human and robot [11].



## List of Contributed Papers

1. Mechanism of a Learning Robot Manipulator for Laparoscopic Surgical Training
2. Learning probabilistic decision making by a service robot with generalization of user demonstrations and interactive refinement
3. A Case Study of Safety in the Design of Surgical Robots: The ARAKNES platform
4. Design of Master Console Haptic Handle for Robot Assisted Laparoscopy
5. Fusion of Inertial Measurements and Vision Feedback for Microsurgery
6. A New Concept for a “Vaginal Hysterectomy” Robot
7. Classification of Modeling for Versatile Simulation Goals in Robotic Surgery
8. Role of Holographic Displays and Stereovision Displays in Patient Safety and Robotic Surgery
9. A methodological framework for the definition of patient safety measures in robotic surgery: the experience of SAFROS project
10. System concept for collision-free robot assisted surgery using real-time sensing
11. Human-Robot Interaction-based Intention Sharing of Assistant Robot for Elderly People

# Mechanism of a Learning Robot Manipulator for Laparoscopic Surgical Training

Tao Yang<sup>1</sup>, Jiang Liu<sup>1</sup>, Weimin Huang<sup>1</sup>, Liangjing Yang<sup>2</sup>, Chee Kong Chui<sup>2</sup>,  
Marcelo H. Ang Jr.<sup>2</sup>, Yi Su<sup>3</sup>, and Stephen K.Y. Chang<sup>4</sup>

<sup>1</sup> Dept. of Computer Graphics and Interface,  
Dept. of Computer Vision and Image Understanding  
Institute for Infocomm Research, Singapore

{tyang, jliu, wmhuang}@i2r.a-star.edu.sg

<sup>2</sup> Dept. of Mechanical Engineering, National University of Singapore, Singapore

{mpeyl, mpecck, mpeangh}@nus.edu.sg

<sup>3</sup> Dept. of Computing Science, Institute of High Performance Computing, Singapore

suyi@ihpc.a-star.edu.sg

<sup>4</sup> Dept. of Surgery National University Hospital, Singapore

cfscky@nus.edu

**Abstract.** This paper presents a robot manipulator for hand-over-hand guidance training of laparoscopic surgery. Details of the mechanical design, kinematic analysis and control mechanism of the robot are presented. The robot records motion of surgical tool manipulated by master surgeon, and provides physical guidance to the novice based on the recorded motion. The robotic manipulator can accurately reproduce the five degree of freedom manipulation of laparoscopic instrument during surgery. A hybrid spherical mechanism is applied for decoupling and reproducing the motion of surgical tool to facilitate implementation of control mechanism. The manipulators for left and right hands are capable of precise execution of a recorded trajectory with observed maximum error of 2.12 mm and 2 mm respectively during an experiment on user interaction.

**Keywords:** Laparoscopy, Surgical training, Robot manipulator.

## 1 Introduction

Laparoscopic surgery with small incisions on the patient as the main treatment approach has become a preference for many types of surgeries. Ninety five percent of cholecystectomy is performed laparoscopically as reported in [1]. Laparoscopic surgery imposes high visual and physical constrains. Surgeons are subjected to demanding visual and physical constrains due to the nature of laparoscopic surgery [2]. Training is crucial for surgeons to obtain the necessary level of proficiency in performing laparoscopic surgeries safely and effectively [3]. Current laparoscopic training equipments range from physical box trainers to virtual simulators [4, 5]. While there are different advantages associated with each of the training methods, none of them mimic the conventional ‘hand-over-hand’ guidance that surgeons use to

teach the novice surgeons by holding and guiding novice surgeon's hands to perform certain tasks or corrections in order to train the motor skills. Researchers have explored the application of robotic assistance in teaching calligraphy and trajectories guidance [6, 7] to train motor skills. However, robotic assistance for the honing of laparoscopic motor skill, to our knowledge, has not been attempted.

In this paper, we propose a robotic platform to deliver robotic assisted laparoscopic skill training. The focus is on the design and development of the robot apparatus capable of active guidance and interactive surgical training. Its mode of operation and training methods are described in Section 2. In Section 3, the design and development of the robotic apparatus are discussed. Kinematics and control implementation are discussed in Section 4. Subsequently, Section 5 discusses the performance of the robotic platform through experimental validation. The paper concludes in Section 6 with a summary of the contributions of this work and our future direction.

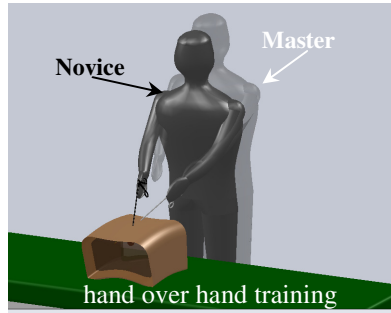
## 2 Overview of Training Methods

'Master-apprentice' and 'hand-over-hand' guidance training strategies are reliable and effective techniques in laparoscopic surgical training, especially for difficult surgical scenarios. Figure 1 illustrates the strategy of 'hand over hand' in conventional training. During the conventional training, sometimes the master surgeon holds the novice's hands to guide the novice to perform certain tasks, sometime the master surgeon allows the novice to perform certain tasks but the master surgeon is always ready to provide correction to the novice's manipulation. This is because that verbal description of motion is always vague and slow. The proposed platform is designed to facilitate training with the learning methods to achieve similar outcome as that of 'hand over hand' training model. This laparoscopic training robot has five degree of freedom on each manipulator that can control the orientation and position of its associated laparoscopic instrument. Figure 2 illustrates the overall structure of the laparoscopic surgical training robot system. It consists of an actuated mechanism that acts as a human machine interface (HMI) to interact with the virtual environment as shown in Figure 2b. While graphical rendering provides visual feedback, haptic feedback to user is achieved through the control of the actuated manipulator for parasitic force compensation and shared control guidance.

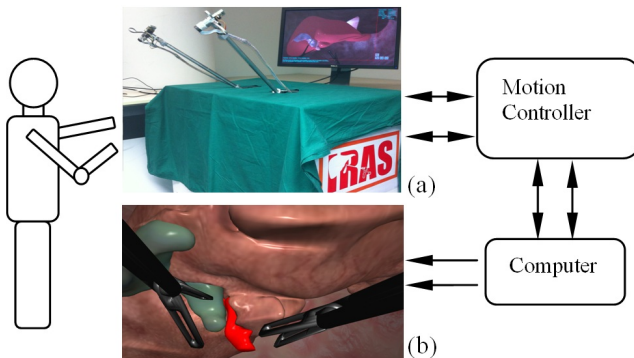
In proposed training model, the master operates on a 3D virtual patient model which is reconstructed from patient's CT images, and has his motion of hands learned and recorded in the training system. There are two guiding approaches. These approaches include complete guidance which is implemented through supervised control scheme, and haptic cue guidance from shared control and human-robot collaborative control scheme.

In complete guidance, the novice is experiencing the tool manipulative motion of a master surgeon kinesthetically by holding onto the surgical instrument that follows the recorded tool manipulative trajectory of the master surgeon. This provides a deeper appreciation to the master surgeon's motion than mere visual and didactic guidance. In haptic cue guidance, the novice is allowed to operate on the patient specific anatomical model with some degree of motion guidance from the robotic device based on his own knowledge. Haptic is implied through the robot manipulator

if the novice's operation deviates from master surgeon's operation. It is advantageous that the novice can be trained via 'hand over hand' method without the master being physically present. Although there has not been any conclusive evidence of benefits to laparoscopic training through kinesthetic guidance from recorded motion, subjects appear to perform tasks better after going through it as suggested in [8].



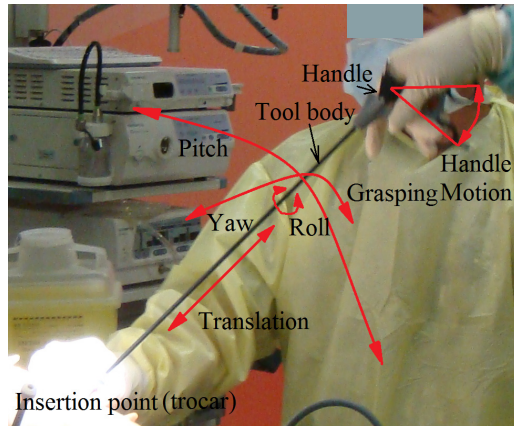
**Fig. 1.** Illustration of hand over hand training strategy



**Fig. 2.** Structure of laparoscopic surgical training robot system. (a) Laparoscopic surgical training robot. (b) Patient specific anatomical model. User operates on the surgical instruments, the virtual instrument interact with patient's anatomical model. Visual feedback is provided by graphic rendering of virtual anatomical model. The motion controller moves the instruments to compensated friction forces, provide haptic feedback, record the trajectories of surgical instruments/ provide active guidance for novices.

### 3 Design and Development

Laparoscopic instruments are long and slender tools. Its applicator is driven by lever mechanism through a handle. Generally, the mobility of a laparoscopic tool constrained at the insertion point (trocar) includes four degree of freedom namely roll,



**Fig. 3.** Motion of surgical instrument in laparoscopy procedure

pitch, yaw and translation. The control of the handle's grasping motion is another degree of freedom though not contributive to its kinematic configuration. Figure 3 illustrates the mobility of one surgical instrument during the surgical process. The robot apparatus which consists of two manipulators representing the surgical instruments is designed with five degree of freedom to fulfill the required mobility. Details of the kinematics of the mechanism are presented in Section 4.

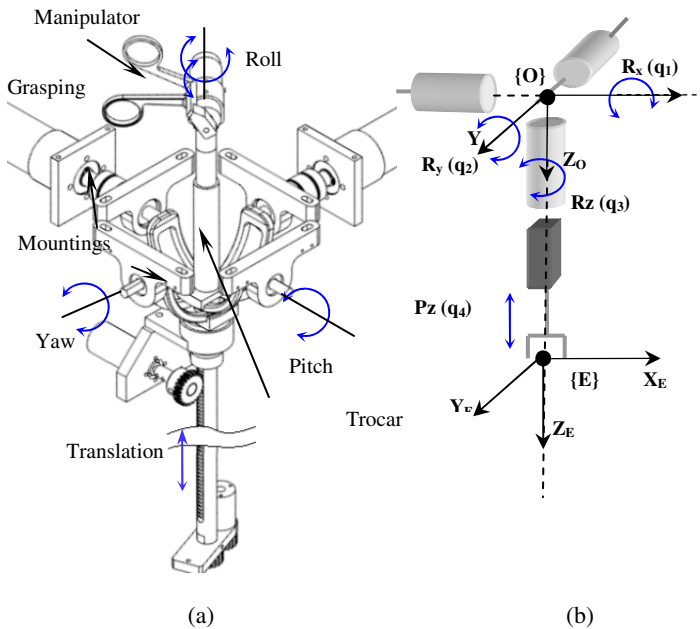
The mechanism, as shown in Figure 4(a), is designed to mimic the kinematic motion of laparoscopic instruments. Spherical mechanism, rack and pinion system, and modified instrument handle, finger detection sensors and foot switch which are highlighted in Figure 5, have been used. The mobility of the mechanism is designed as an exact mimic of the kinematics of laparoscopic procedure. Since the task space of laparoscopic procedure can be readily expressed in spherical coordinates, an ideal mechanism design will be one with axes of control corresponding to the spherical coordinates. The orientation joints should coincident at a point for optimal geometric workspace efficiency [9]. Hence, the hybrid spherical mechanism as shown in Figure 5 is adopted. This hybrid mechanism possesses the advantages of both serial and parallel manipulator as explained in [9]. Similarly, this spherical mechanism is advantageous for both hardware and software performance. Unlike most general manipulators, the spherical mechanism is highly decoupled with each of the actuated axes of control corresponding to task-oriented space coordinates. This direct mapping of joint space to task-oriented space allows high frequency control and enables fast data updating from the actuation and sensory unit to the graphic and haptic rendering module without being burdened by space domain transformation.

In laparoscopic surgery, the instrument is manipulated at substantial moment arm about the insertion point (trocar). This requires high operational torque range for the pitch and yaw axis of control. Although closed-chain mechanism generally provides structural stability, the structural redundancy is workspace inefficient and collision prone with at least two manipulators simulating the laparoscopic instruments.

The argument for parallel mechanism to improve manipulator stiffness is therefore ineffective as it either increases structural link length index or reduces workspace [10]. This may also raise safety concern in user interaction due the high manipulator stiffness associated with parallel linkage configuration. The proposed structure enables a more even and appropriate sizing of actuators for the range of operation configurations.

Apart from kinematic requirements, user centric design attributes like ergonomics and usability are considered. The modified handle is the only physical user interface in the user workspace. All other actuation and control mechanisms are concealed underneath the insertion point (trocar). This user centric configuration produces a more realistic operating environment during training as shown in Figure 2(a).

The range of forces for haptic feedback on each axis is designed for general laparoscopic simulation. Existing literature has shown that the maximum pulling forces along the translation direction, grasping and cutting force are about 17 N, 16 N [11] and 14 N [12] respectively. The actuators are specified base on these guidelines to permit wide range of haptic feedback.



**Fig. 4.** (a) Mechanical mobility of the robot. The travelling limit for pitch, yaw, roll, translation, handle grasping motion are  $120^\circ$ ,  $120^\circ$ ,  $360^\circ$ ,  $350\text{mm}$  and  $60^\circ$  respectively. (b) Kinematic model of surgical instrument.

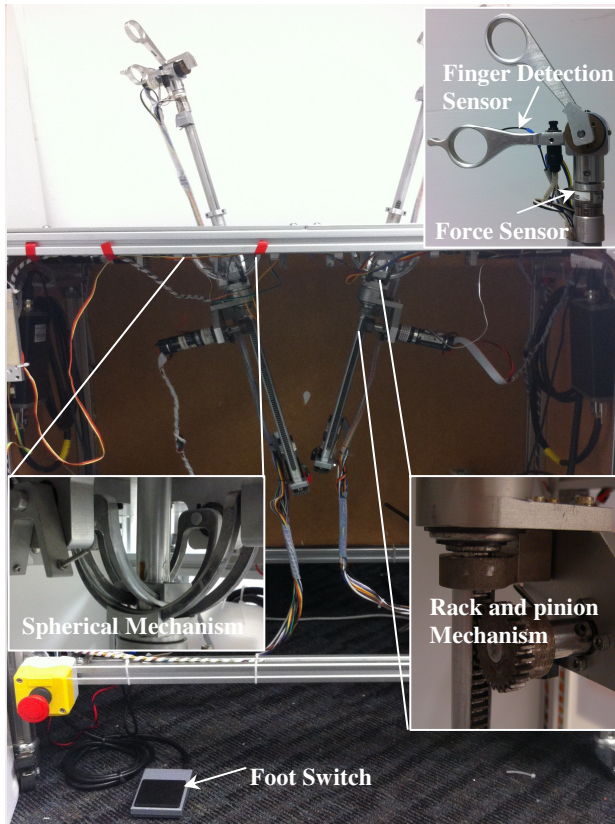


Fig. 5. Details of robotic laparoscopic training device

## 4 Robotic Mechanism

### 4.1 Kinematic Analysis

In laparoscopic procedures where the surgical instrument is constrained along the insertion point (trocar) in its axial direction, the task space configuration can be defined with four degree of freedom through Euler angles, roll, pitch, yaw and translation  $(\alpha, \beta, \gamma, \rho)$ . Figure 4(b) illustrates the frame assignment of the multibody system for kinematic analysis. Actuators are mounted such that their axes of control are aligned to the respective axis of transformation. Hence joint variables  $(q_1, q_2, q_3, q_4)$  correspond to the Euler angle and translation  $(\alpha, \beta, \gamma, \rho)$ .

Homogenous transformation matrix (1) expresses the forward kinematics of Frame  $E$  in Cartesian coordinates,

$${}^0T_E = \begin{bmatrix} \hat{X}_x & \hat{Y}_x & \hat{Z}_x & E_x \\ \hat{X}_y & \hat{Y}_y & \hat{Z}_y & E_y \\ \hat{X}_z & \hat{Y}_z & \hat{Z}_z & E_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s_1s_2s_3 + c_2c_3 & s_1s_2c_3 - c_2s_3 & c_1s_2 & q_4(c_1s_2) \\ c_1s_3 & c_1c_3 & -s_1 & -q_4s_1 \\ s_1c_2s_3 - s_2c_3 & s_1c_2c_3 + s_2s_3 & c_1c_2 & q_4(c_1c_2) \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

where  $s_i, c_i, i=1,2,3$  denote  $\sin(q_i)$  and  $\cos(q_i)$  respectively.

Jacobian matrix to map the joint space  $(q_1, q_2, q_3, q_4)$  into task space is formulated as shown in eq. (2).

$$J_E = \begin{bmatrix} -q_4(s_1s_2) & q_4(c_1c_2) & 0 & c_1s_2 \\ -q_4c_1 & 0 & 0 & -s_1 \\ -q_4(s_1c_2) & -q_4(c_1s_2) & 0 & c_1c_2 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \quad (2)$$

With a given homogeneous matrix acquired from the sensory unit  ${}^0T_E$ , the inverse kinematics is as follows:

$$\begin{aligned} q_1 &= \arctan 2(-E_y, E_z / \cos q_2), \\ q_2 &= \arctan 2(E_x, E_z), \\ q_3 &= \arctan 2(\hat{X}_y, \hat{Y}_y), \\ q_4 &= E_x / (\cos q_1 \cdot \sin q_2). \end{aligned} \quad (3)$$

## 4.2 Control Scheme

The controlling system of the robot is illustrated as shown in Figure 6. Each of the laparoscopy instruments is controlled by one group of motion controller and actuators. Users control the robot through a GUI, the execution commands are sent to two motion controllers and actuators via network controller. The two motion controllers communicate with each to synchronize the motion executed on two groups of actuators.

Figure 7 illustrates the force compensation control scheme implemented on the system. In addition to force feedback mechanism, a dynamic model can be incorporated to compensate undesirable disturbance. The control implementation facilitates the execution of feedback at a rate of 20 kHz to ensure determinism and maintain fidelity. Hence a feedback mechanism is sufficient for the force compensation application.

When the mechanism is driven by user as a passive device, parasitic forces are compensated through force control scheme. The interactive force between user and HMI is acquired by a 6 DOF force sensor installed underneath the handle, as shown in Figure 5. The parasitic force due to the system environment,  $F_O$ , measured by the force sensor in Cartesian coordinates was converted into torque on each joint with



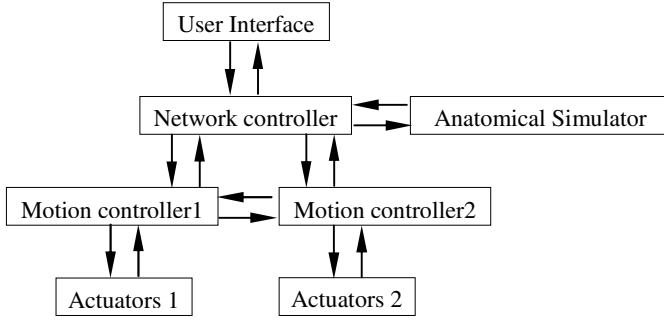


Fig. 6. Robot system control scheme

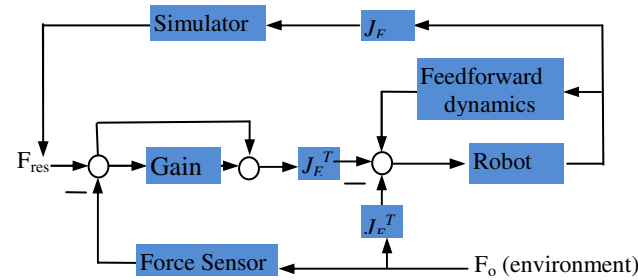


Fig. 7. Impedance control with force feedback

Jacobian matrix (2). Each actuator is commanded with appropriate current to move in corresponding direction according to the direction and magnitude of the torque, and hence reduces parasitic forces measured by the force sensing units.

Current control is implemented to generate the desired force output  $F_{des}$  for the user. The force reference,  $F_{res}$ , as shown in Figure 7 is set as  $F_{des} - F_o$ . The current set point,  $i_{ref}$  is computed by equation (4) as shown below:

$$i_{ref} = K_{\phi}^{-1} J_E^T F_{res} \tag{4}$$

where  $K_{\phi}$  is a 4 by 4 diagonal matrix representing the torque constant of each actuated axis of control,

$J_E$  is the 6 by 4 task space Jacobian matrix,

$F_{res}$  is the desired force/torque vector output in Cartesian coordinates.

In the case of parasitic force compensation without tool- tissue interaction,  $F_{des}$  is equal to zero. Hence, any force reading due to the interaction between user and robot is converted into corresponding current command to drive the actuators to minimize the force reading. Similar method is applied to generate haptic feedback during tool-tissue interaction in the virtual environment with  $F_{des}$  being the required haptic force to output. The computation and actual rendering methodology of the tool-tissue interactive force,  $F_{des}$  is an interesting research topic [13]. It is however beyond the scope of this paper.

For complete guidance, a PID position control is implemented to reproduce the master surgeon's trajectory. All actuators for joints ( $q_1, q_2, q_3, q_4$ ) and handle grasping joint are commanded to move as per the desired trajectory and velocity which acquired from the master surgeon's operation. A position control is appropriate as there is no need for variation in intensity of guidance.

For haptic cue guidance, shared controls of the manipulators are required. Both the user and the predetermined trajectory work as the inputs to the robot to achieve the varying intensity of motion guidance. The force sensor measures the forces  $F_{des}$  that the user experiences, and compensates the parasitic force. At the same time, the current position of the manipulator  $T_n$  is compared with the predetermined trajectory  $T_m$  which acquired from the master surgeon's operation. If the difference is greater than a prescribed value, haptic cue force is provided based on the difference of the two trajectories as follow

$$F_g = K(T_m - T_n) \quad (5)$$

where  $K$  is the coefficient to adjust the force magnitude with respect to trajectory differences.

### 4.3 Control Hardware

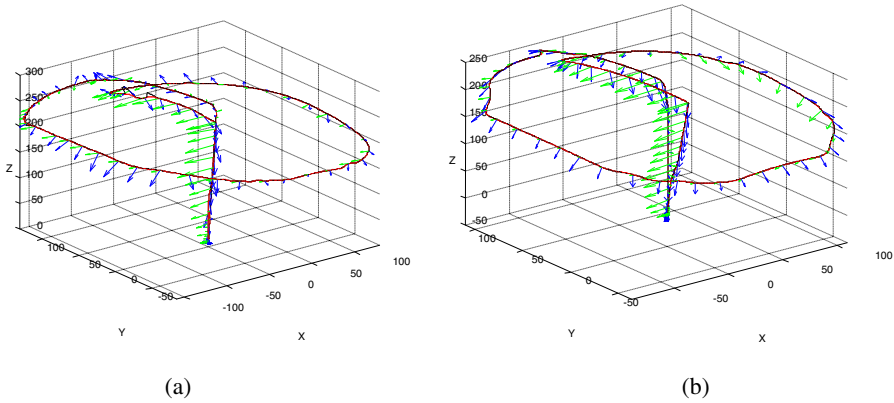
The control hardware platform consists of NI CompactRIO with Xilinx Virtex-5 LX110 reconfigurable I/O FPGA core and real-time embedded controller with 400 MHz processor, 128 MB DRAM memory. Each robotic arm is equipped with six degree of freedom high precision force sensing unit, ATI Nano17, calibrated at a force resolution of 0.0125 N and torque resolution 0.0625 Nmm.

This hardware configuration allows high speed control loop execution, and ensures task determinism for managing communication flow crucial to haptic fidelity and human-machine interface applications in laparoscopic surgical training robot system. The FPGA-based real-time hardware platform is effective in the implementation of reliable controls including force feedback signal processing. The parallelism nature of the FPGA operation mechanism facilitated fast and robust coordination amongst axes simplifying the issue of joint synchronization.

Control operation and computational task were mostly hard programmed in FPGA. This allows minimal delay in the compensation of the parasitic forces. Control of the manipulator is implemented with FPGA at a rate of 20 kHz to ensure determinism and maintain fidelity.

## 5 Experimental Results

The kinematics and dynamics profiles were acquired and analyzed for a given path execution in a specified operational workspace. To evaluate the efficacy of the control mechanism, a recorded trajectory was executed by the robot under condition with and without user interaction. Kinematic trajectories were acquired through the encoder with joint control scheme at frequency of 100 samples/second and subsequently transformed to 3D Cartesian coordinates for analysis. As this computational task is



**Fig. 8.** (a) execution and force on handle of left manipulator, (b) execution and force on handle of right manipulator. Red line is the recorded trajectory, black line is the execution results, blue arrow indicates the force vector on handle, and green arrow indicates the moment vector on handles

independent of the control loop mechanism, it does not create any bottleneck in the control mechanism. The force profile was acquired by the 6-DOF force sensor through FPGA-based DAQ module.

The robot was tested when it was working under complete guidance in the presence of user-system interaction. When there was no interaction with the user, the maximum errors of execution on the left and right manipulators were 2.12 mm and 1.55 mm respectively. The robot was also tested by guiding a user to perform the recorded path. In order to minimize the effect of visual guidance, the trajectory was neither displayed nor known to the user prior to the test. Figure 8 depicts the 3D trajectory and the force profile in Cartesian coordinates when the robot was interacting with a user. The mechanical components and motion control mechanism are capable of precise and accurate execution. The maximum errors of execution on both right and left manipulators are 1.87 mm and 2 mm respectively. The maximum errors of each joint in the left and right manipulators are tabulated in Table 1. This trajectory spanned an approximated  $(0.2 \times 0.2 \times 0.2) \text{ m}^3$  workspace and was subjected to a maximum interaction force range of 3.6 N and torque of 73.7 Nmm.

**Table 1.** maximum error of each joint

Joint	Manipulator	
	Left	Right
Pitch ( $Q_1$ )	0.230°	0.425°
Yaw ( $Q_2$ )	0.241°	0.313°
Roll ( $Q_3$ )	1.66°	1.67°
Translation ( $Q_4$ )	0.524 mm	1.35 mm

## 6 Conclusion

The contribution of this work is on the introduction of a new robot apparatus for hand-over-hand guidance in laparoscopic surgical training. The active robot manipulator designed with appropriate specifications implements an interactive platform that can adequately meet the training needs of laparoscopic surgeries.

The robot is developed with the provision of complete guidance method capable of guiding the novice according to a recorded trajectory. This complete guidance method could provide the novice with deeper appreciation of how an experienced surgeon deals with specified surgical scenarios. Concurrently, we are developing the haptic cue guidance method which allows varying degree of guidance. This will be integrated to the system to facilitate a variant of the hand-over-hand training method.

**Acknowledgements.** This work is partially supported by research grant SERC 092-148-0072 and 102-148-0009 from the Agency for Science, Technology and Research, Singapore

## References

1. Voyles, C.R., et al.: A practical approach to laparoscopic cholecystectomy. *Am. J. Surg.* 161(3), 365–370 (1991)
2. Alberich-Bayarri, A., et al.: Volume Mesh Generation and Finite Element Analysis of Trabecular Bone Magnetic Resonance Images. In: 29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS 2007 (2007)
3. Derossis, A.M., Fau - Fried, G.M., et al: Development of a model for training and evaluation of laparoscopic skills (0002-9610 (Print))
4. Derossis, A.M., et al.: The effect of practice on performance in a laparoscopic simulator. *Surg. Endosc.* 12(9), 1117–1120 (1998)
5. Woodrum, D.T., et al.: Construct validity of the LapSim laparoscopic surgical simulator. *The American Journal of Surgery* 191(1), 28–32 (2006)
6. Teo, C.L., Burdet, E., Lim, H.P.: A robotic teacher of Chinese handwriting. In: Proceedings of 10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems, HAPTICS 2002 (2002)
7. Bluteau, J., et al.: Haptic guidance improves the visuo-manual tracking of trajectories. *PLoS One*, 3(3), e1775 (2008)
8. Lee, C.-S., et al.: Designing an Active Motor Skill Learning Platform with a Robot-Assisted Laparoscopic Trainer. In: 33rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Boston (2011)
9. Vijaykumar, R., Tsai, M., Waldron, K.: Geometric optimization of manipulator structures for working volume and dexterity. In: Proceedings of 1985 IEEE International Conference on Robotics and Automation (1985)
10. Piccin, O., et al.: Kinematic modeling of a 5-DOF parallel mechanism for semi-spherical workspace. *Mechanism and Machine Theory* 44(8), 1485–1496 (2009)

11. Rosen, J., et al.: Surgeon-tool force/torque signatures—evaluation of surgical skills in minimally invasive surgery. *Stud. Health Technol. Inform.* 62, 290–296 (1999)
12. Yang, T., et al.: Modeling cutting force of laparoscopic scissors. In: 2010 3rd International Conference on Biomedical Engineering and Informatics, BMEI (2010)
13. Misra, S., Ramesh, K.T., Okamura, A.M.: Modeling of tool-tissue interactions for computer-based surgical simulation: A literature review. *Presence-Teleoperators and Virtual Environments* 17(5), 463–491 (2008)

# Learning Probabilistic Decision Making by a Service Robot with Generalization of User Demonstrations and Interactive Refinement

Sven R. Schmidt-Rohr, Fabian Romahn, Pascal Meissner,  
Rainer Jäkel, and Rüdiger Dillmann

Institute for Anthropomatics (IFA), Karlsruhe Institute of Technology, Germany  
{srsr, romahn, meissner, jaekel, dillmann}@ira.uka.de

**Abstract.** When learning abstract probabilistic decision making models for multi-modal service robots from human demonstrations, alternative courses of events may be missed by human teachers during demonstrations. We present an active model space exploration approach with generalization of observed action effect knowledge leading to interactive requests of new demonstrations to verify generalizations.

At first, the robot observes several user demonstrations of interacting humans, including dialog, object poses and human body movement. Discretization and analysis then lead to a symbolic-causal model of a demonstrated task in the form of a preliminary *Partially observable Markov decision process*. Based on the transition model generated from demonstrations, new hypotheses of unobserved action effects, generalized transitions, can be derived along with a generalization confidence estimate. To validate generalized transitions which have a strong impact on a decision policy, a request generator proposes further demonstrations to human teachers, used in turn to implicitly verify hypotheses.

The system has been evaluated on a multi-modal service robot with realistic tasks, including furniture manipulation and execution-time interacting humans.

## 1 Introduction

Autonomous abstract decision making for multi-modal service robots needs models of environment causality to generate action policies. Real world environments are partially observable and not fully deterministic. Thus, *partially observable Markov decision processes* (POMDPs) are a suitable representation of an action model. However, efficient policy computation needs models which describe action and observation effects as well as mission goals explicitly.

Comfortable acquisition of these complex models is difficult. Different learning approaches exist: e.g. active learning by trial, interactive reinforcement learning or learning from demonstrations of human teachers. Compared to the first two, the latter approach is efficient as the human has extensive implicit and explicit domain knowledge while natural demonstrations are an efficient way of information transfer.

A disadvantage is the need for a human teachers time, thus imposing the use of a minimum number of demonstrations. Building on a simpler, previous system [1], we

present a methodology which tries to minimize demonstrations needed for the generation of probabilistic planning models by utilizing active model exploration and interactive refinement.

Model representation, the learning process, generalization and request generation are described in sec. 2. Evaluation of the method on a real robot is described in sec. 3 while sec. 4 compares related work and sec. 5 gives a conclusion.

## 2 Model Generation

The aim is to generate a POMDP [2], a tuple  $(S, A, M, T, R, O)$ , for a specific robot service mission from a set of human demonstration sequences. Symbolic sets of states  $S$ , actions  $A$ , observations  $M$  and transition probabilities  $p(s, a, s') \in T$  as well as observation probabilities  $p(s, m) \in O$  and rewards  $r(s, a) \in R$  have to be compiled. Based on a model, an execution-time policy can be computed by approximate value iteration, e.g. SARSOP [3].

Below, the relation between model symbols and real world is discussed in sec. 2.1 and a learning process overview given in sec. 2.2. Subsequently, generalization is presented in sec. 2.2 and request generation in sec. 2.4.

### 2.1 Service Robot Mission Model Representation

State symbols  $S$  are related to real world characteristics by robot perception and corresponding models, reflected by measurement symbols  $M$ .  $S$  and  $M$  are linked by  $O$ . Action symbols  $A$  are linked to actuator components.

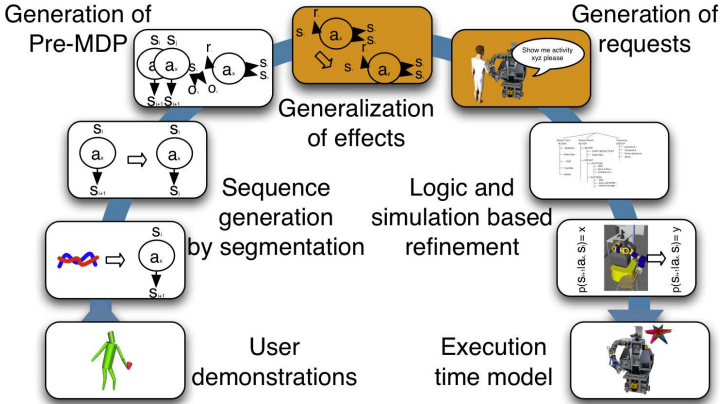
A state  $s$  thus corresponds to a set of properties of the world  $p$  which can potentially be observed by a set of perception components. On a robot with several perception components, a state  $s \in S$  is a class defined over a distinct combination of property sets. Each  $p$  results from a potential observation of a perception component, called modality  $D_l$ . A symbolic modality state  $d_j^l \in D_l$  covers a set of similar potential observations  $p_a, \dots, p_z$ .

$$s_i \in S : d_{x_1}^1 \in D_1 \times \dots \times d_{x_n}^n \in D_n \quad (1)$$

E.g.:  $\{D_1, D_2, D_3\} : \{\text{robot position, human utterance, object poses}\}$ ,  
 $s_i : \text{robotAtTable} \times \text{bringTea} \times \text{teaOnTable}$ .

While related to a factored representation, because of modality interdependencies in  $T$  and  $O$ , it cannot be handled as such. An exemplary transition:  $p(s_i, a_k, s'_j) \in T = 0.6$ ,  $a_k : \text{FetchTea}$ ,  $s'_j : \text{robotAtHuman} \times \text{thankYou} \times \text{teaAtHuman}$ . The granularity of  $S$ ,  $A$  is adjusted by discretization mode.

When learning models from demonstrations for both  $T$  and  $R$ , two main types of model aspects can be distinguished. In  $T$  these are *structural* stochastic transitions modeling agent independent characteristics of a mission on the one hand and *robot capability* stochastic transitions on the other.



**Fig. 1.** The process for generating probabilistic decision making models by learning from human demonstrations (PMPM-PbD). Emphasis of this paper is highlighted.

E.g  $p(\text{robotAtHuman} \times \text{noTeaPresent}, \text{GotoTable}, \text{robotAtTable} \times \text{teaAtTable}) = 0.4$ , structurally modeling the chance to encounter tea at the table and  $p(\text{robotAtTable} \times \text{teaAtTable}, \text{PickTea}, \text{robotAtTable} \times \text{teaInRobotHand}) = 0.8$ , modeling motion planning and execution capabilities of the robot. In  $R$ , the types are *mission goal* positive rewards and *action cost* negative rewards.

**2.2 Process Overview**

Major stages of *Probabilistic mission planning model programming by demonstration* (PMPM-PbD) are depicted in fig. 1 with the focus of this paper highlighted.

*Observation.* utilizes multiple sensors and perception processing components, either robot based - as in the setup, presented in sec. 3 - or parts of a smart room. One or several interacting humans perform natural demonstrations of a mission with differing courses of events. Data of all components, e.g. object localization, is recorded.

*Abstraction.* derives mapping  $P \rightarrow S$  followed by segmentation. As a result, each demonstration is represented as a state-action sequence  $(\dots, s_t, a_t, s_{t+1}, a_{t+1}, \dots)$ . Automatic discretization techniques as clustering in continuous property domains may be used.

*Mapping.* generates a preliminary (PO)MDP model based on several demonstrations. All  $s, a$  in sequences are accounted for. Then, structural transitions  $T_D$  and goal rewards are generated by analysis of sequences. Probabilities  $p(t_i)$  are derived from occurrence frequencies in demonstrations sets. More details about mapping can be found in [1].

*Generalization.* derives transition hypotheses for unobserved transitions related to those appearing in demonstrations. The aim is to maximize information gathered from a limited set of demonstrations as described in sec. 2.3



*Request generation.* takes generalization confidence to rate transitions on necessity to be verified as presented in sec. 2.4. Relevant transitions are assembled into potential courses of events. Based on these sequences, human teachers are given requests to verify hypotheses. Together, generalization and requests perform hypotheses expansion and pruning.

*Refinement.* Several model aspects cannot be learned from demonstrations including observation model, action costs and robot capability transitions. PMPM-PbD uses background knowledge, geometric analysis and learning in dynamics simulation to compute these aspects, transforming a preliminary into a final model. Details are not further relevant for generalization and requests.

*Autonomous execution.* Policy computation generates a policy from this model by approximate value iteration. Finally, the policy is used for autonomous execution-time decision making of the robot performing the mission. An exemplary architecture is described in [4].

## 2.3 Generalization of Transitions

Determining new structural non-zero transition probabilities in a preliminary  $T$  can be interpreted as generating new hypotheses for causal effects, thus *generalizing* observed effects. On the abstract level, modalities  $D_i$  are the main structure suitable for generalization. *Modality states*  $d_x^i \in D_i$ , e.g.  $teaAtTable, teaInRobotHand \in D_{obj} : (object\ poses)$  and corresponding *flat states* (see eqn. 1) reflect this structure. To estimate probabilities for  $T(s_i, a_k, s'_j)$  with previously assigned  $p(s_i, a_k, s'_j) = 0$  - thus unobserved - similarities within  $D_i$  and thus  $S$  have to be exploited. There are three types of generalization:

1. Generalize origin:  $p(s_i, a_k, s'_j) \rightarrow p(s_g, a_k, s'_j)$
2. Generalize effect:  $p(s_i, a_k, s'_j) \rightarrow p(s_i, a_k, s'_g)$
3. Generalize action:  $p(s_i, a_k, s'_j) \rightarrow p(s_i, a_g, s'_j)$

The third type is not investigated further. The other two types imply that when observing transitions for several similar  $s_i$  (1) or  $s'_j$  (2), e.g when those states differ only in a single modality, chances are high that transitions may be non-zero for further similar states. To reflect respective transition similarities, a *transition mask*  $\kappa$  is defined:

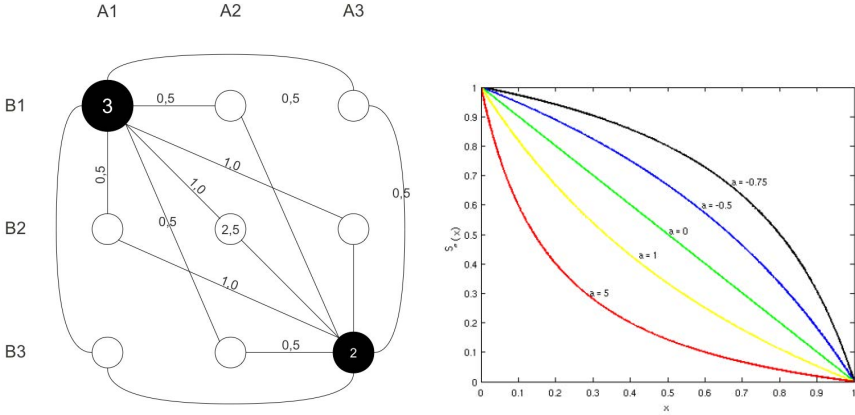
$$A^* = A \cup \{*\}, A^*(a_i^*) = \begin{cases} A & \text{if } a_i^* = * \\ a_i & \text{else} \end{cases} \quad (2)$$

$$D_l^* = D \cup \{*\}, D_l^*(d_l^*) = \begin{cases} D_l & \text{if } d_l^* = * \\ d_l & \text{else} \end{cases} \quad (3)$$

$$S^* = D_1^* \times \dots \times D_n^* \quad (4)$$

$$\kappa(s^*, a^*, s'^*) = \{s^* \in S^*, a^* \in A^*, s'^* \in S^*\} \quad (5)$$

E.g.:  $\kappa_{example1} = \kappa(* \times bringTea, HandTea, robotAtHuman \times thankYou)$  models from where the robot got tea is irrelevant for effect  $s'^*$ , thus a wildcard  $*$  represents the robot location



**Fig. 2.** Transition distance with two modalities A, B, each with three modality states and two observed transitions (black circles with observed frequency) on the left. Sugeno negation for some  $\alpha$  parameters on the right.

modality  $D_{robotloc}$  in  $s^*$ .  $\kappa_{example1}$  covers  $|D_{robotloc}|$  transitions. Masks are used to define the scope of generalization  $\kappa_{scope}$  - e.g. type 1) has wildcards only in  $s^*$ , but not  $s'^*$  - and to describe the resulting, generalized hypotheses set  $\kappa_g$ .

Using masks, a set of hypotheses  $T_g$  for a set of transitions  $T_I$ , can be computed, given the following three elements:

1. A set of input transitions  $T_I$ , generating the hypothesis.
2. A transition mask  $\kappa_{scope}(s^*, a^*, s'^*)$  defining the elements over which may be generalized.
3. A transition mask  $T_g = \kappa_g(s^*, a^*, s'^*)$  containing the resulting generalized set of transitions.

Scope mask  $\kappa_{scope}$  is generated by analyzing onto which modalities actions have an effect in the set  $T_I$  in the types 1) and 2), as  $a$  is not generalized. For all  $t(s, a, s') \in T_I, p(t) > 0$ , each modality is checked for a change and in case  $d_j^i \in s \neq d_k^i \in s'$ , a wildcard is inserted in  $\kappa_{scope}$  for the respective  $D_i$  in  $s^*$  (1) or  $s'^*$  (2).

The resulting set  $\kappa_g$  is derived by concept learning over  $T_I$  with mask  $\kappa_{scope}$ , whereby all  $t \in T_I$  have to be considered as positive examples. Simply put,  $\kappa_g$  is the most specific mask which covers all of  $T_I$  with generalization (wildcards) only allowed, where defined by  $\kappa_{scope}$ .

Each  $t_g \in \kappa_g(s^*, a^*, s'^*)$  is a contender for being non-zero, thus generalization confidence and estimated transition probability are calculated. Confidence is computed using distances between transitions, based on modality state distances. Probabilities can then be computed based on confidences and related transitions.

To compute distances between states  $c(s_1, s_2)$ , modality specific distance metrics  $dc(d_1^i, d_2^i)$  are used, e.g. denoting geometric distance between regions or similarity between utterances or objects. If no specific metric is available, a simple, general metric  $dc_{generic}$  can be used:

$$dc_{generic}(d_j^i, d_k^i) = \begin{cases} 1 & \text{if } d_j^i = d_k^i \\ 0 & \text{else} \end{cases} \quad (6)$$

$$c(s_1, s_2) = \frac{1}{n} \sum_{i=1}^n dc(s_1 : d_j^i, s_2 : d_k^i) \quad (7)$$

$$c(t_1, t_2) = c(s \in t_1, s \in t_2) \text{ type 1), } c(s' \in t_1, s' \in t_2) \text{ type 2)} \quad (8)$$

For each generalized transition  $t_g \in \kappa_g$ , the nearest, observed transition, called *baseline* reference transition  $t_b$  is determined:  $c(t_g, *) = c(t_g, t_b) = \min(c(t_g, t_i), t_i \in T)$ . To compute basic generalization confidence  $gc \in [0, 1]$ ,  $c(t_g, *)$  is coupled with a negation function, e.g. Sugeno-Negation  $N_s(x) = \frac{1-x}{1+\alpha x}$ ,  $\alpha \in (-1, \infty)$ :

$$gc_{basic}(t_g) = \begin{cases} 1 & \text{if } t \text{ was observed} \\ N_s(c(t_g, *)) & \text{else} \end{cases} \quad (9)$$

Especially when using simple distance metrics, further aspects have to be considered when computing generalization confidence values. One such aspect, called *non-observation bias* (nob), reflects a derived transition  $t_g$  being less likely implied by a high number of observations of a reference transition  $t_b$ . Another aspect, called *certainty bias* (ceb) considers that an observed transition  $t_{occ}(s_i, a_k, s'_j)$  with the same origin state and action as  $t_g(s_i, a_k, s' \neq s'_j)$ , has its effect probability reduced by including  $t_g$ . However, observed transitions are always a better model estimate, thus including a bias against such  $t_g$ . This results in a refined generalization confidence  $gc_T(t_g)$ :

$$nob(t_g) = (1 - \beta)^{occ(t_b)}, \beta \in [0, 1] \quad (10)$$

$$t_g(s_i, a_k, s'_j), n = occ(t(s_i, a_k, s'_j)), ceb(t_g) = \begin{cases} 0 & \text{if } n = 0 \\ \sqrt[n]{\gamma} & \text{else} \end{cases} \quad (11)$$

$$gc_T(t_g) = gc_{basic}(t_g) * nob(t_g) * ceb(t_g) \quad (12)$$

Parameters  $\alpha$ ,  $\beta$  and  $\gamma$  determine the share of each aspect and have to be determined empirically for a setting.

While  $gc_T(t_g)$  is also used directly in further processing steps (see sec. 2.4), its immediate purpose is to calculate transition probabilities  $p_T(t_g)$  for hypotheses based on the most similar observed transition and its confidence:

$$\bar{p}_T(t_g(s_i, a_k, s'_j)) = gc_T(t_g) * p_T(t_b), t_b : \text{baseline } t \quad (13)$$

$$p_T(t_g) = \frac{\bar{p}_T(t_g)}{\sum_{s' \in S} p(s_i, a_k, s')} \quad (14)$$

Generalization assumes that estimated model knowledge, which hypotheses are, is better than always maintaining the zero assumption arising potentially from missing demonstrations. This is a Bayesian way of handling lack of information, yet not without

pitfalls. Aggressively generalized transitions which do not reflect real world properties might in the worst case lead to an unusable policy. Therefore, a verification stage as described next is crucial.

## 2.4 Demonstration Request Generation for Verification

Request generation consists of two parts: relevance estimation of generalized transitions  $t_g$  and composition of demonstration requests. Relevance estimation ranks  $t_g$  so that requests are only generated to verify high impact  $t_g$ .

Relevance estimation is performed individually for  $t_g$ . Because evaluating all possible combinations of tuples  $(t_g^1, \dots, t_g^n)$  is infeasible, relevances arising only from multiple  $t_g$  added together, are not covered.

To evaluate a single  $t_g$ , the original model  $T_D$  generated from demonstrations without any generalization is taken and  $t_g$  added to form an evaluation model. For some further steps, an initial state  $s_{init}$  is needed, which can either be the most frequent initial state in demonstrations, be sampled from all initial states in demonstrations or defined by hand. Next, a policy is computed from the evaluation model and a number of simulation runs with that policy performed on the model. With these steps, all information has been gathered to calculate a set of relevance aspects  $v_i(t_g)$ . First, a subset, *state relevance* aspects  $v_i(s_i)$  are computed:

$$\text{Freq. in demo. paths : } v_{demo}(s_i) = |s_i \in demo| / |demo| \quad (15)$$

$$\text{Freq. in sim. paths : } v_{sim}(s_i) = |s_i \in sim| / |sim| \quad (16)$$

$$\text{In T : } v_{in}(s_i) = \frac{|\{(\bar{s}, a, s_i) | T(\bar{s}, a, s_i) > 0, \bar{s} \in S, a \in A\}|}{|S| * |A|} \quad (17)$$

$$\text{Prob. in transitions : } v_{out}(s_i) = \frac{\sum_{\bar{s} \in S, a \in A} T(\bar{s}, a, s_i)}{|S| * |A|} \quad (18)$$

$$\text{Rel. utility : } v_{util}(s_i) = \frac{u(b(p(s_i) = 1.0)) - \min(u(s \in S))}{\max(u(s \in S)) - \min(u(s \in S))} \quad (19)$$

Total state relevance is empirically weighted:

$$\bar{w}_i, \{demo, sim, in, out, util\} : v(s_i) = \frac{\sum_{j=1}^5 \bar{w}_j v_j(s_i)}{\sum_{j=1}^5 \bar{w}_j} \quad (20)$$

Relevance for  $t_g(s, a, s')$  can then be estimated:

$$\text{Origin state relevance : } v_s(s, a, s') = v(s) \quad (21)$$

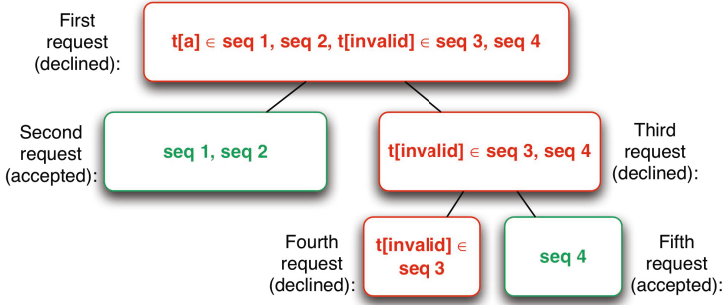
$$\text{Effect state relevance : } v_{s'}(s, a, s') = v(s') \quad (22)$$

$$\text{Freq. in sim. paths : } v_{sim}(s, a, s') = |(s, a, s') \in sim| / |sim| \quad (23)$$

$$\text{Relative reward : } v_r(s, a, s') = \frac{r(s, a) - \min(r(\bar{s}, \bar{a}))}{\max(r(\bar{s}, \bar{a})) - \min(r(\bar{s}, \bar{a}))} \quad (24)$$

$$\text{Probability : } v_p(s, a, s') = p(s, a, s') \quad (25)$$

$$\text{Generalisation confidence : } v_{gc}(s, a, s') = gc_T(s, a, s') \quad (26)$$



**Fig. 3.** Successive request generation for a path containing an invalid generalized transition which will never occur in a demonstrated mission

Total transition relevance is empirically weighted:

$$w_i, \{s, s', sim, r, p, gc\} : v(s, a, s') = \frac{\sum_{j=1}^6 w_j v_j(s, a, s')}{\sum_{j=1}^6 w_j} \quad (27)$$

Hypotheses  $t_g$  can then be ranked for inclusion in interactive demonstration requests which lead to verification of these  $t_g$ . Interactive requests have to be understood by human teachers, should both include as many relevant  $t_g$  as possible and be suitable for a coherent demonstration sequence. First, suitable potential state-action chains have to be generated, each containing a maximum number of relevant transitions. Then, this chain has to be transformed into a request description which allows enough flexibility for the human to include deviations is necessary.

A greedy algorithm is used which performs a path search from an initially chosen generalized transition  $t_g^a$  via observed transitions  $t_i$  to a closest generalized transition  $t_g^b$ . The path search prefers likely transitions  $p(t_i(s, \bar{a}, s')) > p(t_j(s, a, s')), p(t_i(s, \bar{a}, s')) > p(t_k(s, \bar{a}, s'))$  and short paths, using a breadth-first search. For  $t_g^a$ , a path from a start state  $s_{init}$  (as described above) to  $t_g^a$  is searched first. If that path cannot be found,  $t_g^a$  is temporarily discarded. Next, paths  $t_r^a \rightarrow t_r^b, t_r^b \rightarrow t_r^c, \dots$  to closest transitions are incrementally retrieved, added as long as one can be found.

A mechanism is necessary to detect  $t_g^{invalid}$  which do never occur in the mission. If the critical  $t_g^{invalid}$  in the path cannot be bypassed in the demonstration, the whole request is declined without the system getting to know where it failed. Hence, for each final state-action chain request, a binary tree is formed with each level splitting the set of remaining  $t_g^x$  in the path in two halves. In case a  $t_g^{invalid}$  is present in a segment, only the longest chains in the tree without  $t_g^{invalid}$  will be demonstrated. That way, the  $t_g^{invalid}$  can be identified after further demonstrations. Initially, the human teacher is only presented with the request representing the root path and then successively with sub-nodes if requests have to be declined because of  $t_g^{invalid}$  (see fig. 3).

Next, the internal representation has to be transformed into a natural request. Four techniques are suitable in principle:

1. State-action chains: unambiguous, but very complicated request descriptions and inflexible.
2. Action sequences: compact request descriptions, but effects cannot be controlled.
3. State sequences: complicated request descriptions, actions ambiguous, limited flexibility.
4. Sequences of modality changes: similar to state sequences, but compact request descriptions.

In practice, a combination of modality changes and action sequences balances advantages best. A sequence of modality changes is given with two special cases: a) in case a transition has  $s = s'$ , only the action is given, b) for  $t_g$  (but not other  $t_i$ ), the action is given additionally to the modality change.

Then, the sequence is transformed into a graph visualization or natural speech, e.g. "Begin in state top/Pos1. Change the modality Pos from Pos1 to Pos2. Change the modality Lane from top to bottom and Pos from Pos2 to Pos3 by performing action switch." .

Finally, the resulting demonstrations are incorporated into the model generation process.

### 3 Experimental Setup and Results

The process as described in sec. 2.2 has been implemented on a multi-modal service robot including skills such as autonomous navigation, localization of small and large (furniture) objects, natural speech processing, classification of human body movements on top of NITE body tracking as well as autonomous motion planning (see fig. 4 and 5). The same skill components are used during both observation of demonstrations and autonomous execution. A Kinect sensor was utilized for visual perception components.

#### 3.1 Demonstration Stage

During demonstrations, the robot observes one human, *robot actor* (roAc), representing the role of the robot and optionally a second human, representing an interacting human, *human actor* (huAc). Classification of human activities translates a body movement over time into symbolic activities *MovementType* as described in [5]. As a result, the following tuple of recording modalities  $D_1, \dots, D_n$  is used:

$(roAc.Pose, huAc.Pose, huAc.MovementType, huAc.Utterance, Obj.Poses)$

And action  $a$  modalities:

$(roAc.MovementType, roAc.Utterance)$

The recording system performs abstraction as mentioned in sec. 2.2 for each modality. These modalities cover common aspects of multi-modal service robots, therefore, the utilized system is a typical representative. The robot follows a *roAc* actively with its head while the wide angled Kinect sensor allows to keep a large area around in the field



**Fig. 4.** Snapshot of a recording. Robot observing on the left, human actor in the center, robot actor on the right.

of view to track objects and a *huAc* in the vicinity. In case of dialog, however, headsets are used for clear distinction between *roAc* and *huAc*. Several demonstrations of a pre-defined mission, representing differing courses of events and structural transition probability frequencies in that mission are recorded and segmented, leading to a set of sequences used for mapping and generalization.

### 3.2 Evaluation Missions

For evaluation of generalization and requests, a mission was chosen, serving a human a chair discretely and proactively without any spoken dialog. Basically, the goal was for the robot to learn which poses (distance and orientation) of the interacting human reflected a high likelihood of a desire to sit down and with which manipulation actions to pull a chair towards the human depending on the chair pose.

This mission contains several different types of uncertainties, present in the real world execution in a compact sized model, suitable for evaluation. Structural transitions representing human behavior and chair poses before pulling as well as the basic state and action sets can be generated from demonstrations.

### 3.3 Generation Stage

Two missions were demonstrated, A) with a focus on generalizing initial human pose (position and orientation) in respect to the likelihood of sitting down when presented with a chair (high reward). Mission B) had a focus on position and orientation of the chair and corresponding likelihoods of two different classes of motions (parameterizations of motion planning) being suitable for pulling the chair.



**Fig. 5.** Execution snapshot. Interacting human on the left, robot performing the robot role. Manipulation actions are executed using parameterized motion and grasp planning with online object localization.

In mission A) the following modalities  $D_i$  were used:  $D_1^A$ : RobotPosition,  $D_2^A$ : Chair-Position,  $D_3^A$ : ChairOrientation,  $D_4^A$ : HumanPose and 3 different demonstration sequences performed, each a different number of times, to achieve non-uniformly distributed probabilities. The mapping lead to  $|D_1^A| = 4, |D_2^A| = 4, |D_3^A| = 3, |D_4^A| = 9 \Rightarrow |S| = 432, |A| = 8$ . A total of 1157 generalizations of non-observed effects ( $t_g$ ) were made, with 18 being considered crucial by a human expert analysis to scale to non-demonstrated aspects of the mission. Most non-crucial  $t_g$  affected the *idle* action. A total of 20  $t_g$  were rated above the general threshold of which 7 had to be declined as invalid. The rest could be confirmed with demonstrations, again a different number of times to correct the estimated frequencies to the desired mission values. For mission B) the same modalities were used and 5 different demonstration sequences, again a different number of times performed. The mapping lead to  $|D_1^A| = 4, |D_2^A| = 5, |D_3^A| = 5, |D_4^A| = 5 \Rightarrow |S| = 400, |A| = 8$ . A total of 1273 generalizations of non-observed effects ( $t_g$ ) were made, with 14 being considered crucial by a human expert analysis. Most of the other  $t_g$  affected the *idle* action. A total of 22  $t_g$  were rated above the general threshold and could be confirmed. These examples show the necessity of relevance estimation and the first one also the need for the binary tree request generation to decline false hypotheses.

### 3.4 Execution Stage

During execution of the mission by the robot, the computed policy was used by the online decision making system to autonomously select the next symbolic action when the previous terminated. Robot motion planning actions are represented by *manipulation strategies* (see [6] for details of the representation). All perception, processing



and actuation was running solely on board (see fig. 5). In the real setting, deviations of perceived human and chair poses as well as in executed navigation and manipulations actions occur regularly as considered by observation model probabilities and error transition probabilities in the model. Deviations in real human poses and chair poses, as considered by structural transitions, were applied in the same manner as during demonstrations.

### 3.5 Evaluation Results

For evaluation, policies based on models generated from initial demonstrations and then further demonstrations after requests were pitted against a hand modeled finite state machine (FSM). To evaluate if the POMDP was able to consider the learned aspects sufficiently, the mission was performed (no matter if POMDP or FSM were controlling the robot) with human behavior and chair pose frequencies corresponding to the pre-defined mission concept (the "real" probabilities). Execution time (average duration of one main task performance within the mission) and failure frequency can be considered the primary performance criteria of the decision making system. Both missions were executed by the robot (and interacting human) and measured by a supervisor ten times for each controller. It included courses of events which were not present in the initial demonstrations, but only after generalization and request generation.

Mission	Avg. dur. POMDP	Fail. POMDP	Avg. dur. FSM	Fail. FSM
A	4:30 min	2/10	4:50 min	3/10
B	4:35 min	3/10	4:45 min	4/10

Failures included pulling the chair when the human had no interest and failing to pull the chair. Delays occurred when taking some time to interpret the human pose correctly, looking again for the chair or retrying to pull the chair.

The results show that the learned POMDP was able to match and even slightly surpass the performance of the hand-tailored FSM. The learned POMDP can achieve good performance from an efficient number of natural demonstrations of non-technical experts.

## 4 Related Work

There exists work related to several aspects presented here. Many approaches to learn symbolic task descriptions from human demonstrations are discussed in literature. Learning models for probabilistic decision making has been tackled mainly without using natural human demonstrations but still highlights important aspects for model generation. Generalizing task models for planning is also related. Finally, generation of interactive requests to improve models has been applied successfully to task model creation.

Popular low level imitation learning is quite distinct from the work presented here. On the symbolic level, several different categories can be distinguished, among them learning hierarchically organized sequences, learning finite state machines and learning planning models. A system to learn hierarchical representations of manipulation

tasks was investigated by [7], yet neither multi-modality, nor flexibility by planning are covered. Learning finite state machines completely from the scratch, including basic structure and transitions was presented in [8] and tested in the robot soccer domain. Learning planning models, although not probabilistic ones, is discussed in [9].

Apart from using model-free policy learning, generating (PO)MDP models by means of learning is the obvious approach to the challenge how to retrieve the model. A technique which uses active learning with meta-distributions has been developed in [10], refining several aspects, e.g. transitions probabilities. Yet, it needs an initial state and action space and also an initial model. A theoretically even more powerful framework, BA-POMDPs, is explained in [11]. Because of complexity aspects, however, its planning horizon is very small in real settings, limiting its practicality. An imitation-learning related approach, generating properties of an MDP model for specific robot tasks is presented in [12].

There is a multitude of work concerning generalization of task knowledge for planning. In the scope of this work, those techniques, considering observations of humans are most relevant. A way to modify symbolic plans by analysis of observation of humans performing tasks was investigated in [13]. In that case, the application domain on a multi-modal service robot is also very similar to the PMPM-PbD application. The hierarchical PbD system in [7] was also extended by means of generalization over different, similar demonstrations, however, without further verification.

Interactive requests to a human by a robot to improve task models were used in a work [14] also using a kind of relevance measure to guide efficient selection of requests.

## 5 Conclusions and Future Work

We present a process to extend a POMDP model of a multi-modal robot mission when generated from a limited set of human demonstrations. It performs generalization of observed causal effects with subsequent verification by rating and then generation of requests for efficient further demonstrations. A strong emphasis was made on implementing the concept on a real, complex, multi-modal service robot, including both autonomous recording of demonstrations and execution to evaluate its suitability in realistic applications.

Future work has to include multi-mission spanning background knowledge for more fine grained generalization masks  $\kappa_{scope}$ , leading to even better selected hypotheses sets. Concerning request generation, empirical weighting parameters  $w_i$  could be learned over many missions.

**Acknowledgement.** This work has been conducted within the german SFB 588 “Humanoid Robots” granted by DFG.

## References

1. Schmidt-Rohr, S.R., Lösch, M., Jäkel, R., Dillmann, R.: Programming by demonstration of probabilistic decision making on a multi-modal service robot. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, Taipei, Taiwan (2010)

2. Cassandra, A.R., Kaelbling, L.P., Littman, M.L.: Acting optimally in partially observable stochastic domains. In: Proceedings of the Twelfth National Conference on Artificial Intelligence (1994)
3. Kurniawati, H., Hsu, D., Lee, W.: SARSOP: Efficient point-based POMDP planning by approximating optimally reachable belief spaces. In: Proc. Robotics: Science and Systems (2008)
4. Schmidt-Rohr, S.R., Knoop, S., Lösch, M., Dillmann, R.: Bridging the gap of abstraction for probabilistic decision making on a multi-modal service robot. In: RSS, Zürich (2008)
5. Lösch, M., Schmidt-Rohr, S., Knoop, S., Vacek, S., Dillmann, R.: Feature set selection and optimal classifier for human activity recognition. In: RO-MAN (2007)
6. Jaekel, R., Schmidt-Rohr, S.R., Loesch, M., Dillmann, R.: Representation and constrained planning of manipulation strategies in the context of programming by demonstration. In: IEEE International Conference on Robotics and Automation, ICRA 2010 (2010)
7. Pardowitz, M., Knoop, S., Dillmann, R., Zollner, R.: Incremental learning of tasks from user demonstrations, past experiences, and vocal comments. IEEE Trans. on Systems, Man, and Cybernetics (2007)
8. Grollman, D., Jenkins, O.C.: Incremental learning of subtasks from unsegmented demonstration. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (2010)
9. Veeraraghavan, H., Veloso, M.: Learning task specific plans through sound and visually interpretable demonstrations. In: IROS (2008)
10. Ross, S., Chaib-draa, B., Pineau, J.: Bayes-adaptive pomdps. In: NIPS. MIT Press (2007)
11. Jaulmes, R., Pineau, J., Precup, D.: A formal framework for robot learning and control under model uncertainty. In: 2007 IEEE International Conference on Robotics and Automation (April 2007)
12. Shon, A.P., Storz, J.J., Rao, R.P.N.: Towards a real-time bayesian imitation system for a humanoid robot. In: 2007 IEEE International Conference on Robotics and Automation, pp. 2847–2852 (2007)
13. Tenorth, M., Beetz, M.: Priming Transformational Planning with Observations of Human Activities. In: IEEE International Conference on Robotics and Automation, ICRA (2010)
14. Chernova, S., Veloso, M.: Interactive policy learning through confidence-based autonomy. Journal of Artificial Intelligence Research 34 (2009)

# A Case Study of Safety in the Design of Surgical Robots: The ARAKNES Platform\*

L. Alonso Sanchez<sup>1</sup>, M.Q. Le<sup>1</sup>, K. Rabenoroso<sup>1</sup>, C. Liu<sup>1</sup>, N. Zemiti<sup>1</sup>, P. Pognet<sup>1</sup>,  
E. Dombre<sup>1</sup>, A. Menciassi<sup>2</sup>, and P. Dario<sup>2</sup>

<sup>1</sup> Dept. of Robotics, LIRMM

French National Center of Scientific Research (CNRS), Montpellier, France  
{sanchezsec, le, rabenoroso, liu, zemiti, pognet, dombre}@lirmm.fr

<sup>2</sup> BioRobotics Institute, Scuola Superiore Sant'Anna

Pisa, Italy

{arianna.menciassi, paolo.dario}@sssup.it

**Abstract.** This work presents a case study of safety in the design of the ARAKNES surgical robotic platform dedicated to single port laparoscopic surgery. The framework for the design of medical robots is shortly described and applied, focusing on safety. Moreover, it is explained how the design process can be placed in the context of the European community directives for medical devices.

**Keywords:** surgical robots, safety standards, design framework.

## 1 Introduction

Originally, medical robotics designs were inspired by the success of industrial robotics. However, the specificities of medical applications led to research on original kinematics, actuation mechanisms, compatibility with medical imaging devices, biocompatibility and more, and also to modify the regulations for production of medical devices, comprising surgical robots. After nearly three decades of research in the field, many prototypes have been built and validated technically, some clinically, but just few managed to find their way into operating rooms or medical offices [1].

In general, a medical robot is a complex system that consists of 1) an articulated and motorized mechanical structure, 2) electronic components, 3) a software controller, and 4) a human machine interface (HMI). These components are used to perform interventions in a constrained and not fully structured environment, inside and/or outside of the patient's body, in collaboration with the medical staff. Thus, it is easy to understand that a system failure or dysfunction can be extremely critical. For that reason, safety guidelines should be followed in order to ensure that each component is designed, implemented and integrated to achieve the intended medical tasks in optimal conditions and in compliance with the existing device regulations.

---

\* This work was supported by the European Commission in the framework of ARAKNES FP7 European Project no. 224565.

Hardware safety is the basic requirement for a robot. Prior enhancement methods included sensor redundancy, mechanical limits and default detection [2]. Ng. et al. also presented mechanical safety enhancement methods for a surgical robot [3]. Ergonomics analysis methods have also been used for HMI design [4]. Lewis and Maciejewski put forwards method to analyze joint failures [5] whereas Ikuta et al. used impact force and stress as the safety value for human-care robots [6].

Software safety received more attention with the intensive use of computers, integrated circuits and advanced functionality in robots. Safety approaches which could be used include fault tree analysis [7], [8], even tree analysis [7], fault tolerance algorithm [9] and dependability principles [10]. Recently, Haddadin et al. proposed reactive control strategies that can significantly improve the human safety during physical interaction with environment [11]. The control approach consists of a collision detection method, allowing to reduce contact forces far below any level which is dangerous to humans.

Dombre et al. presented in [1] a generic framework for the design of medical robots taking into account the above mentioned considerations. In this paper, the design process of the ARAKNES platform used for single port laparoscopic surgery is presented [12]. Particular interest is dedicated to system safety. Moreover, it is also described how the design framework can comply with European directives for bringing a surgical robot prototype into the operating room.

This paper is organized as follows. Section 2 introduces the ARAKNES platform and the involved design framework, focusing on the implemented safety features. Section 3 places the design framework in the context of existing European directives for medical devices. Finally, conclusions and perspectives are drawn in section 4.

## 2 The ARAKNES Platform: From Surgical Gesture Analysis to Robot Design

The ARAKNES platform (Figure 1) is an innovative concept for endoluminal surgery in which the main objective is to locate most of the Degrees of Freedom (DoF) available to the surgeon inside the patient, leading to less invasiveness. Preliminary results concerning the first developed prototypes were presented in [12].



**Fig. 1.** ARAKNES platform concept for Single Port Laparoscopy [13]

According to [1], the design process can be described by four tasks, which deal with the system specifications, the design methodology, the technological choices, and the system safety. The first three tasks are indirectly related to system safety, while the last one tackles directly the issue. Each task and its corresponding subtasks are shortly described in this section.

## **2.1 Definition of the Surgical System Specifications**

### **2.1.1 Analysis of the Surgical Gesture**

To define the system requirements, the surgical gesture is analyzed by the robotic expert that studies how the medical staff performs a similar procedure. Apart from looking at the state-of-the-art of the concerned intervention, data is measured and recorded (e.g. tool positions, manipulation forces, configuration of the elements inside the operating room). Then, an analysis of all the information will lead to the system specification (i.e. required number of DoF, required speeds, forces, and more).

In the ARAKNES platform for instance, instruments are inserted through a cylindrical access port in the navel, through an incision of about 30–35 mm [14]. The navel is used to gain access to the abdomen in a practically scar-less way. However, it requires a remote-centre-of-motion (RCM) at the level of the umbilicus. Additional information concerning the required forces and speeds can be found in [13]. The ARAKNES system should also offer the surgeon a dexterity and freedom of movement equivalent to the ones of his/her hands.

### **2.1.2 Kinematic and Dynamic Specifications**

The specifications are defined from the data analyzed in the preceding step. For instance, a bimanual robot with six active DoFs plus the gripper is required in order to reproduce the movements of the surgeon's hands. An external manipulator for positioning the miniature arms inside the abdomen should respect the RCM constraint. Refer to [12], [13], [15] for more details on the ARAKNES prototype.

### **2.1.3 Kinematic Topologies**

An appropriate type of kinematics to satisfy the specifications is then selected by the designer [1]. Conventional serial kinematics, for instance, allows achieving a large workspace, high dexterity and good obstacle avoidance capabilities (e.g. “snake” like robots), necessary for completing surgical tasks, while keeping the two miniature robotic manipulators attached to the umbilical access port. Parallel kinematics could offer, on the other hand, a stiff support that complies with the RCM constraint required by the external positioning mechanism holding both the access port and the bi-manual robot [15].

## **2.2 Design Methodology**

Prototypes are the result of a set of design choices, and methodological approaches exist to lead to an implementation of a concept. These approaches depend on whether the starting point is a robot concept (topological synthesis) or parameters optimization

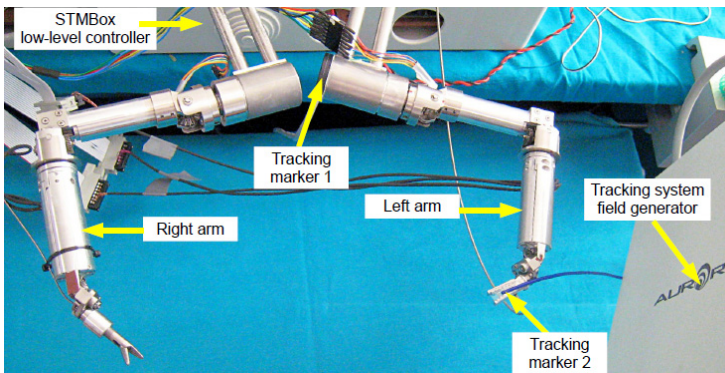
of a previously designed concept is carried out (dimensional analysis). In the case of ARAKNES, the starting point is a new concept constrained by the use of the smallest available components, such as electrical actuators and embedded electronic components. The reader can refer to [1] and [3] for more detailed examples of design methodologies.

### 2.3 Technological Choices

Next, the designer has to make a series of decisions concerning the actuators, sensors and materials which will be used to manufacture the prototypes. For instance, if the system will be used under a MRI or a CT scanner, the compatibility of its components has to be verified. Additionally, certain components must be sterilizable.

### 2.4 System Safety

Figure 2 shows the bi-manual robot prototype of the ARAKNES platform [12]. The safety features that have been implemented on this prototype are the result of the application of a set of recommendations for risk reduction in medical robots as described in [1]. The recommendations are based on 1) the use of intrinsically safe components, 2) the concept of redundancy, and 3) other additional design constraints, such as joint position and velocity limits and software thresholds on critical signals like contact force. All guidelines in [1] can be applied at the hardware (electromechanical and electrical components), software and operational levels.



**Fig. 2.** The bimanual robot prototype for single port laparoscopic surgery

#### 2.4.1 Hardware Safety

The hardware safety is tackled at two levels: the mechanical level and the electrical level.

Concerning the mechanical level, the mechanical design of the bi-manual arms excludes most singularities of the robot's workspace, with the only exception of the wrist singularity which is handled by software [12]. A high reduction ratio limits the

speed of the actuators. The robotic arms are designed in such a way that in case of a power failure, the robot is locked due to the gear reduction ratio.

Concerning the electrical level, ST Microelectronics developed custom electronic boards, namely STMBox [12]. These boards could be sterilizable, and include power limitations of the actuator's current in order to avoid overheating or damaging the miniature electrical motors. Moreover, a deadlock watchdog and a communications timeout were integrated in the low-level actuator controller. A watchdog timer is an electronic hardware device that can be used to automatically detect software anomalies and reset the processor, if any occurs. In this case, the system will be restarted or safely stopped as if a human operator had switched off the power.

The global safety sequence performed in case of error is:

1. Engage brakes if available. For instance, the external positioning platform DIONIS [15] has breaks, but the internal bi-manual platform SPRINT does not [13]. On the other hand, it has a reduction ratio that keeps the system in position.
2. Stop motion using the emergency stop ramp time and deceleration.
3. Wait for the axes to be stopped and the brakes to be engaged, as well as limit the time for robot power-off after an error.
4. Disable the power amplifiers.
5. Disable the actuators power, but leaving sensors and encoders powered-on. Thus, joint positions are known at any time, even when motors are uncontrolled.

## 2.4.2 Software Safety

The software components of the ARAKNES platform involve 1) the surgeon's console interface, 2) the low-level actuator controller and 3) the high-level controller software. In each case, an incremental design through functional blocks of the components has been carried out, allowing to reduce the certification requirements of some components (i.e. the console is considered a Class I medical device while the other two are Class IIb devices as it will be explained in section III).

### 2.4.2.2 Software Architecture

As shown in Figure 3, the software architecture was implemented using concurrent processes running in parallel and in separate processing units [12], [16] dedicated to specific tasks: security (Main task), Cartesian control (Left and Right arm tasks), Joint control (STMBox task), Communication with peripheral units and sensors (force sensor and electromagnetic tracker tasks), HMI communication (teleoperation task), and so on.

Another software safety which was implemented is the supervision task (Figure 4), which uses a software watchdog to verify the correct execution of the other processes at each sampling period in a round-robin manner. It is also possible to check the status of the operating system thanks to the "proc" interface provided in Linux RTAI [12]. If an error is detected an emergency shutdown routine is executed to stop the system in a predictable manner.



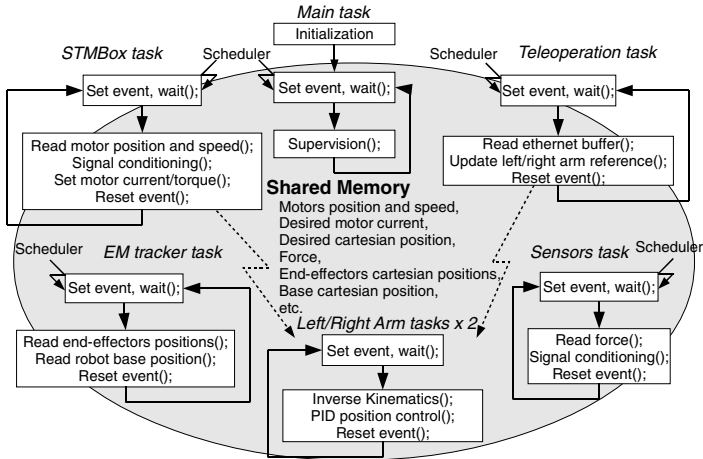


Fig. 3. Modular high-level controller software architecture [12], [16]

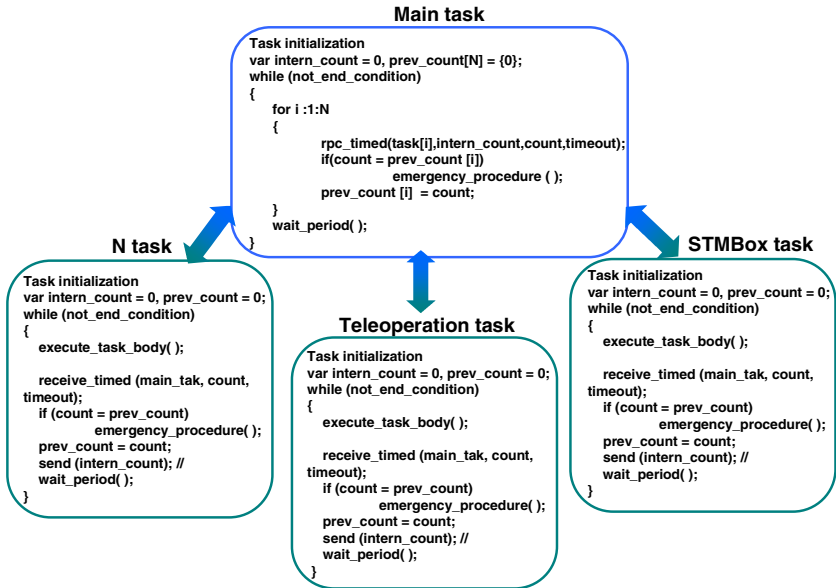


Fig. 4. Dedicated watchdog/supervision process to check task execution in round-robin fashion [12]

Variable time consuming procedures should be avoided to decrease the execution-time of the processes (thus, to increase the sampling frequency for reacting faster in case of emergency). If an overrun is detected, it is taken into account in the control algorithms by using a real elapsed time value.

#### 2.4.2.2 *Kinematic Singularities*

The kinematics resolution method takes into account the possibility that the robot is lead through or nearby a singularity configuration by using a damping factor in order to ensure system stability [17]. Most singularities are out of the workspace thanks to the designed mechanical structure, but special care is taken concerning the wrist singularity.

#### 2.4.2.3 *Virtual Fixtures*

Virtual fixture joint limits have been implemented. To be effective, these limits must be lower than the mechanical ones. This fictive redundancy allows to increase the lifetime of mechanical components and to guarantee a limit even if the mechanical joint limit is broken. This feature may also be interesting to reduce the workspace according to the operating mode and create forbidden region virtual fixtures.

Concerning the low-level controller software, the custom hardware and software developed by ST Microelectronics allows the use of threshold limits in order to restrict the payload, forces, torques, velocities, and acceleration, which might useful for handling collisions.

### 2.4.3 **Operational Safety**

At the operational level, user manuals must be furnished for components such as the high-level and low-level controllers. Software version control and documentation systems were integrated during the implementation phase, as required by most device certification standards. Additionally, other partners of the ARAKNES project [18] are developing advanced functions for the surgeon's console, such as pre-operative planning and training of the surgeon and his team with a simulator.

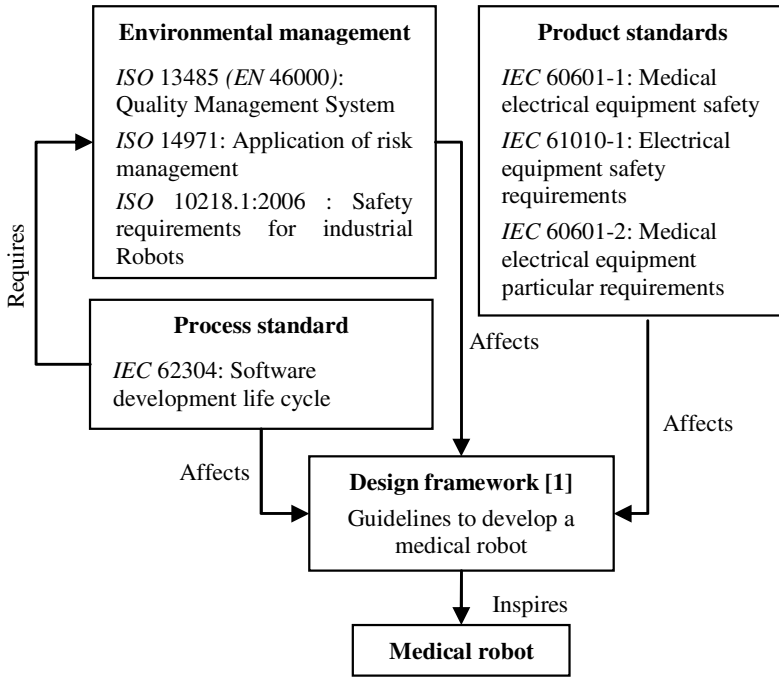
## 3 **European Community Directives**

The European directives 93/42/CEE and 2007/47/CE classify all medical devices into 4 classes:

- Class I: low level of risk
- Class IIa: level of average risk
- Class IIb: high level of risk
- Class III: critical level of risk

The classification depends on criteria such as the duration of use, the level of invasiveness, the active nature – or not – of the device, the concerned organ, etc. According to the previous criteria, each individual component or unit of a surgical robot can take part for any class (e.g. the surgeon's console is Class I as mentioned in Section II). However, if the system is considered globally, a surgical robot could be part of any class between IIa and III. The ARAKNES platform is a Class IIb active surgical device.

Figure 5 shows a diagram of the relationship between the applied medical robot design framework [1] and the European standards for medical devices. The latter define a set of documents and tests required to control the product development,



**Fig. 5.** Relationship between the presented guidelines for medical robot design and the European directives for medical devices

validation and manufacturing processes before the final product release. Readers should refer to these standards to determine the exact requirements for each hardware or software component.

The environmental management is of particular importance in order to obtain the European marking required for taking a robot into the operating room.

Recently, JWG9 (Joint Working Group) has been formed by IEC SC 62A with ISO TC 184/SC2, and it is expected that additional medical robots standards will be introduced [19].

In any case, in the design methodology each subsystem should be identified by the following properties: level of risk, medical device class, and type of component (i.e. electromechanical, electrical, etc.) in order to apply the risk reduction recommendations of [1].

## 4 Conclusion

In this paper, a case study in the design of surgical robot was provided. A generic framework including system specifications, design methodology, technological choices, etc., has been used for a safety assessment dedicated to the single port laparoscopic surgery. During system integration and evaluation, other remaining

issues will be taken into account for providing clear error messages or warning signals (such as lights or sounds) to the surgeon in order to indicate the system's status, errors or unauthorized behaviors. Besides the design criteria for mechanical systems, the one for vision systems will be also considered as one of important research paths to be investigated, as they have indirect impact on patient safety. Finally, another aspect of this study is to improve working conditions within the operation room environment by ensuring not only safety but also comfort and ergonomics of the surgeon during laparoscopic surgery.

**Acknowledgements.** This work was supported by the European Commission in the framework of ARAKNES FP7 European Project no. 224565.

## References

1. Dombre, E., Poignet, P., Pierrot, F.: Design of Medical Robots. In: Troccaz, J. (ed.) *Medical Robotics*, 1st edn., pp. 141–176. Wiley (2011)
2. Troccaz, J., Lavallee, S., Hellion, E.: A passive arm with dynamic constraints: a solution to safety problems in medical robotics. In: *International Conference on Systems, Man and Cybern. Systems Engineering in the Service of Humans*, pp. 166–171 (1993)
3. Ng, W.S., Tan, C.K.: On safety enhancements for medical robots. *Reliability Engineering & System Safety* 54(1), 35–45 (1996)
4. Rau, G., Radermacher, K., Thull, B., Von Pichler, C.: Aspects of Ergonomic System Design Applied to Medical Work Systems. In: *Computer-integrated Surgery: Technology and Clinical Applications*, pp. 203–221 (1996)
5. Lewis, C.L., Maciejewski, A.A.: Dexterity optimization of kinematically redundant manipulators in the presence of joint failures. *Computers & Electrical Engineering* 20(3), 273–288 (1994)
6. Ikuta, K., Nokata, M.: General evaluation method of safety for human-care robots. In: *International Conference on Robotics and Automation*, vol. 3, pp. 2065–2072 (1999)
7. Khodabandehloo, K.: Analyses of robot systems using fault and event trees: case studies. *Reliability Engineering & System Safety* 53(3), 247–264 (1996)
8. Connolly, B.: Software safety goal verification using fault tree techniques: a critically ill patient monitor example. In: *Conference on Computer Assurance, Systems Integrity, Software Safety and Process Security*, pp. 18–21 (1989)
9. Hamilton, D., Visinsky, M., Bennett, J., Cavallaro, J., Walker, I.: Fault tolerant algorithms and architectures for robotics. In: *Electrotechnical Conference*, pp. 1034–1036 (1994)
10. Dowler, N.J.: Applying software dependability principles to medical robotics. *Computing & Control Engineering Journal* 6(5), 222–225 (1995)
11. Haddadin, S., Albu-Schaffer, A., De Luca, A., Hirzinger, G.: Collision detection and reaction: A contribution to safe physical human-robot interaction. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3356–3363 (2008)
12. Sanchez, A., Petroni, G., Piccigallo, M., Scarfogliero, U., Niccolini, M., Liu, C., Stefanini, C., Zemiti, N., Menciasci, A., Poignet, P., Dario, P.: Real-Time Control and Evaluation of a Teleoperated Miniature Arm for Single Port Laparoscopy. In: *International Conference in Medicine and Biology Society*, pp. 7049–7053 (2011)
13. Piccigallo, M., Scarfogliero, U., Quaglia, C., Petroni, G., Valdastrì, P., Menciasci, A., Dario, P.: Design of a novel bimanual robotic system for single-port laparoscopy. *IEEE/ASME Transactions on Mechatronics* 15(6), 871–878 (2010)

14. Galvao Neto, M., Ramos, A., Campos, J.: Single port laparoscopic access surgery. *Techniques in Gastrointestinal Endoscopy* 11(2), 84–93 (2009)
15. Beira, R., Santos-Carreras, L., Rognini, G., Bleuler, H., Clavel, R.: A novel remote-center-of-motion parallel manipulator for minimally invasive surgery. In: *Applied Bionics and Biomechanics, Special issue in Surgical Robotics* (2010)
16. Poignet, P., Dombre, E., Merigeaux, O., Pierrot, F., Duchemin, G.: Design and control issues for intrinsically safe medical robots. *International Journal on Industrial Robot* 30(1), 83–88 (2003)
17. Chiaverini, S., Siciliano, B., Egeland, O.: Review of the damped least-squares inverse kinematics with experiments on an industrial robot manipulator. *IEEE Transactions on Control Systems Technology* 2(2), 123–134 (1994)
18. ARAKNES Project Website, Internet, <http://www.araknes.org> (accessed on January 31, 2012)
19. Doctor Robot, I presume? International Electrotechnical Commission, [http://www.iec.ch/etech/2011/etech\\_0711/ind-2.html](http://www.iec.ch/etech/2011/etech_0711/ind-2.html) (accessed on April 24, 2012)

# Design of Master Console Haptic Handle for Robot Assisted Laparoscopy\*

Chi Yen Kim<sup>1</sup>, Byeong Ho Kang<sup>2</sup>, Min Cheol Lee<sup>3</sup>, and Sung Min Yoon<sup>2</sup>

<sup>1</sup> Division of Mechanical Engineering Technology  
Yeungnam College of Science & Technology  
Daegu, South Korea  
chiykim@ync.ac.kr

<sup>2</sup> School of Mechanical Engineering  
Pusan National University  
Busan, South Korea  
kbh8548@nate.com  
Tactics1019@gmail.com

<sup>3</sup> Department of Mechanical Engineering  
Pusan National University  
Busan, South Korea  
mclee@pusan.ac.kr

**Abstract.** Now robot assisted laparoscopy has some problem to depend only on image of the surgical field. However, there is a research result which shows that it is possible to measure the operation force on commercialized instrument inside patient without sensors [1]. By using measured information, haptic technique can be implemented into surgical robot system. Therefore, this paper proposes the design of master console handle to put haptic function into laparoscopy surgical robot system. The design is centered on the implementation particularly. Therefore this paper suggests the design of master console handle whose structure can match with the joint and DOF of instrument to and the adoption of cable-conduit mechanism to make light structure to perform a delicate haptic function. Cable-conduit mechanism removes the weight and inertia of link caused by haptic actuator and encoder which is separated from handle link.

**Keywords:** Haptic, Surgical robot, Robot assisted Laparoscopy.

## 1 Introduction

Robot assisted laparoscopy is better than standard laparoscopy as easier instrument manipulation, limited mobility of straight laparoscopic instruments and ergonomic position for the surgeon [2]. In robot assisted laparoscopy, while the surgeon's hands

---

\* This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (2011-0014915).

and fingers direct the surgery using master console handle, the movements are translated by the computer to precise movement of the microsurgical instruments on the robotic arms inside the patient's body. As the surgeon is separated from patient, he cannot feel the touch of organ and so has to depend only on image of the surgical field, which occur some problems [3].

To improve laparoscopy surgical robot system to supplying to surgeon with the surgical field sensing information, several researches have attempted to put haptic function into surgical robot system[4][5][6]. However, these studies has mainly focused on haptic mechanism and stayed in research step before becoming commercialized. And commercialized laparoscopic surgical robot, Da Vinci, adopts better redundant DOF master handle then DOF of slave robot to perform comport driving to reduce fatigue. To realize haptic function on existed surgical robot system, each link of master handle has to contain encoder to measure the movement of surgeon's motion and haptic actuator to generate haptic force. Therefore, increasing weight and inertia makes difficult to perform a delicate haptic function as surgical sense.

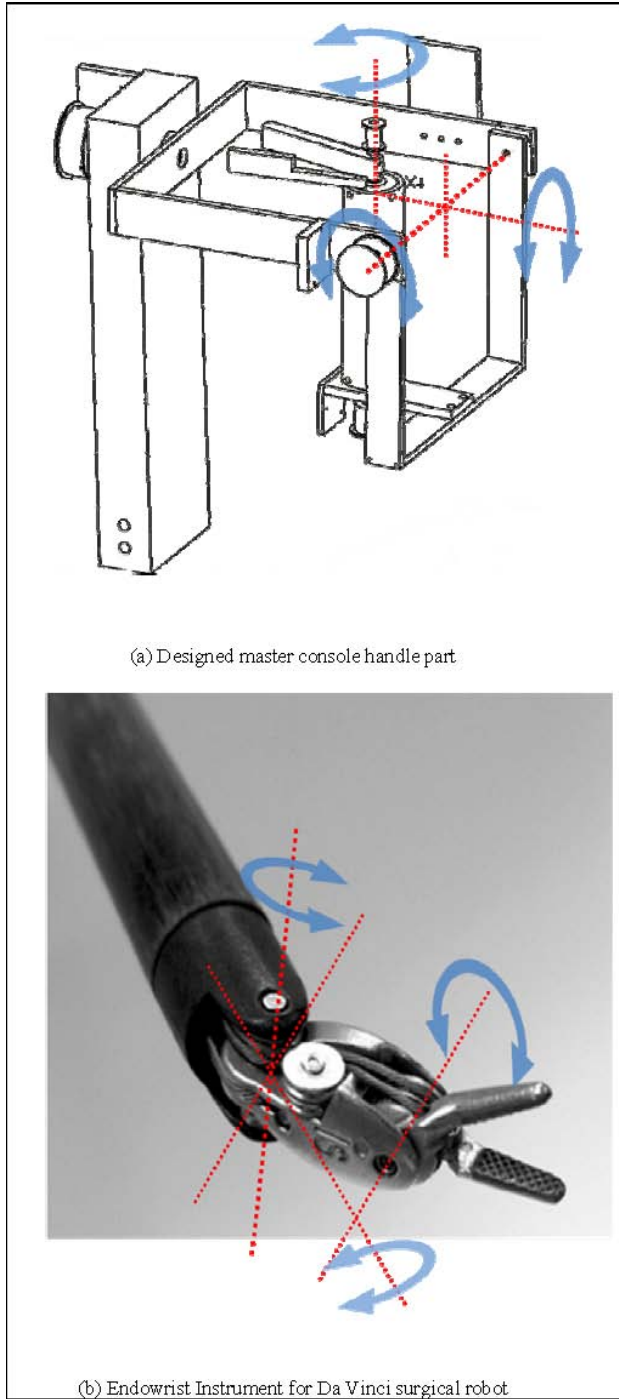
This study proposes light master handle design which can be used in already commercialized surgical robot system to apply haptic function. To facilitate application of haptic system into commercialized system application, this paper suggests very similar structure with the end-effect part of instrument used in slave robot arm. To prune the weight and inertia of handle so that it can generate minute force which can be felt by human, cable-conduit structure is adopted.

## **2 Design of Master Handle**

### **2.1 Design of Master Console Handle**

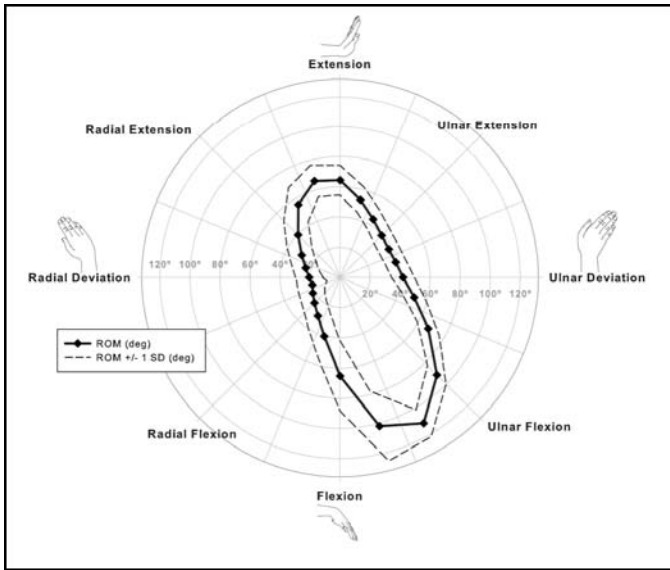
As explained at introduction, commercialized laparoscopy surgical robot systems use a redundant DOF master handle to perform comport driving for surgeon. However, the redundant DOF master console handle will meet the difficulties when haptic function is applied into the system in that added links increase the weight and inertia as well as complex haptic force calculation is required to generate the force. Actually, the reason why commercialized surgical robot didn't consider the haptic system is the difficulties to measure the operating force on instrument. However, in previous research shows the possibility of sensing the force without sensors [1]. Thus, this paper designs the handle with considering haptic system.

Generally human has habitude to orient himself to implement easily in spite of constrained shape. Particularly, if shape of handle is same as ocular one of instrument, adoptive habitude is much better in visual servoing system like as laparoscopy. Therefore, this paper proposes the design of master console handle whose structure can match with the joint and DOF of instrument as possible as it can.



**Fig. 1.** Designed scheme compared with Endowrist instrument tip





**Fig. 2.** Mean range of motion [7]

Fig. 1 shows the handle design scheme proposed by this paper. To make similar shape and structure with the tips of instrument, gripping axis is deviated from the union axis origin. The roll and pitch axis of handle are concentrated on one point same as the instrument. And the number of DOF, joints of handle is equal to one of instrument. During designing the shape, the size and workable space is decided after considering the size of hands and range of motion value as shown in Fig 2 stated in Joseph's paper [7].

## 2.2 Adopt of Cable Conduit Structure

The major contribution of this paper is the weight reduction design to embody the haptic function in robot assisted laparoscopy. To make light structure, this paper suggests the adoption of cable-conduit power transmission mechanism to measure the motion of surgeon's handling and generate the haptic force in master handle. Cable-conduit mechanism can be simply explained as cable displacement transmission caused by cable sliding on conduit and fixation during the rigid state of conduit. However, as a conduit shows flexible feature and motion, it looks like transferring the motion and force freely in the space. Therefore, inner motion of free space object like as the handle in the master console of the robot assisted laparoscope seems to be transferable after using cable conduit method. Again, the force seems to be able to be transmitted as the force of cable system is transferred by the tension of cable displacement.

The dynamic of cable-conduit mechanism is analyzed by Agrawal and et al [8]. As the paper said in details, the force and motion is transmitted by relative movement between cable and conduit. For the reason, the friction between them affects much on the force and movement as following government equation.

$$u''(x,t) - \frac{\mu}{R(x)} u'(x,t) \text{sign}(\dot{u}(x,t)) = 0 \tag{1}$$

Where,  $u(x,t)$  denote its axial displacement in  $x$  at time  $t$ , the partial derivative of a function  $u(x,t)$  to the spatial variable  $x$  is denoted by  $u'(x,t)$  and the partial derivative with respect to the time variable  $t$  by  $\dot{u}(x,t)$ ,  $\mu$  is the coefficient of friction,  $R(x)$  is the radius at  $x$  as shown in fig .5.

In fact, the cable-conduit system has a merit which transmits the force and displacement freely in space. However the force is affected by shape of conduit, as  $R(x)$  in eq. 1. The shape seems to be friction in force transmission. The shape friction brings on slacking phenomenon which occasion time delay caused by cable propagation and remaining backlash during reverse direction changing as analyzed at[8].

From the viewpoint of applying cable conduit into master consol haptic handle, slacking effect will occur time delayed measurement at the first movement of surgeon's motion and wrong measurement at the moment of changing direction. Therefore, to prevent the problem, low friction cable-conduit material has to be chosen. And curvature length of cable-conduit could be linearised straightly as possible as it can in the implementation of constructing master console handle for haptic system.

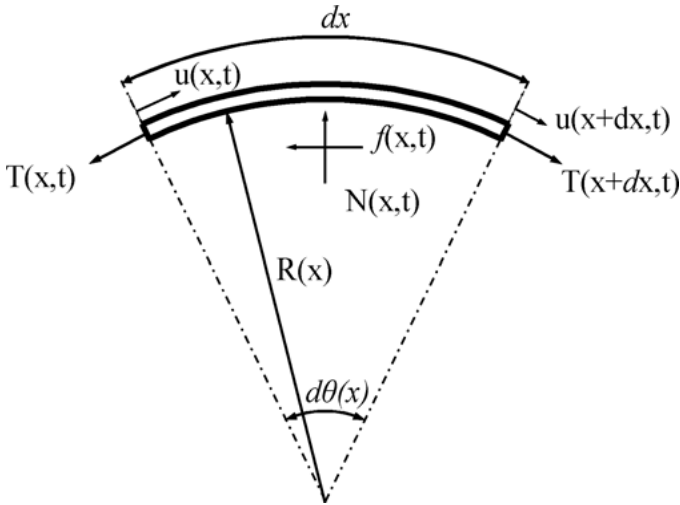


Fig. 3. Forces balance diagram of cable element [8]

### 3 Realization and Experimental Test

The proposed design of this paper in Fig. 1 is exemplified by producing model made up by aluminum and PVC pipe as shown in fig 4. Whole weight of exemplified model is less than 0.8 Kg except standing object which is installed in right side in fig 4. Therefore the proposed design and realization satisfies the light 4 DOF structure enable to make up precise haptic system available to master console handle for robot assisted laparoscope system. Still under constructing, the feeling of driving is experimented and felt as natural and non-constraint movement within wrist moving range as in Fig. 2.



**Fig. 4.** Exemplified handle model

To make up complete handle, pulley for each joint should be attached on. But the whole weight of 4 DOF will be not over 1.2 kg. Comparing with 5 DOF existing handle which contains encoder and sometimes motor to compensate weight, proposed system is much lighter structure below the weight of motor and encoder and so could be less burden to surgeon.

The final aim of this study is constructing complete 4 DOF haptic handle using cable-conduit which is able to make user fill the force generated for haptic signal as well as measure the motion of user movement. However it is in process study, this paper tries to cover up to making structure and showing the possibility of adopting cable-conduit. To prove the possibility of adopting cable-conduit mechanism in handle for surgical robot master consol, this paper examined the cable conduit system for 1 DOF as shown in Fig. 5. The experimental test system is constructed to generate movement and measure it in other side. Also the pretension can be adjustable by using sliding plate and screw.

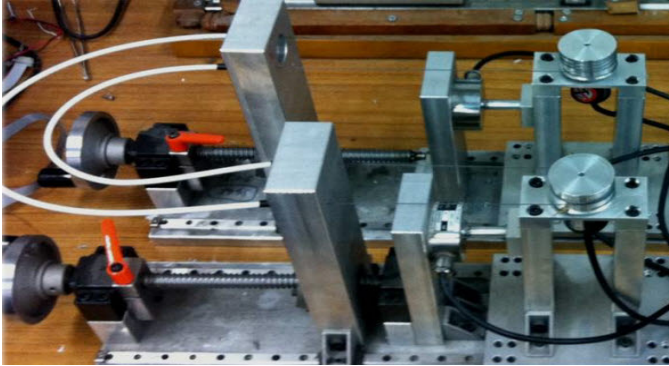


Fig. 5. Experimental system of cable conduit system

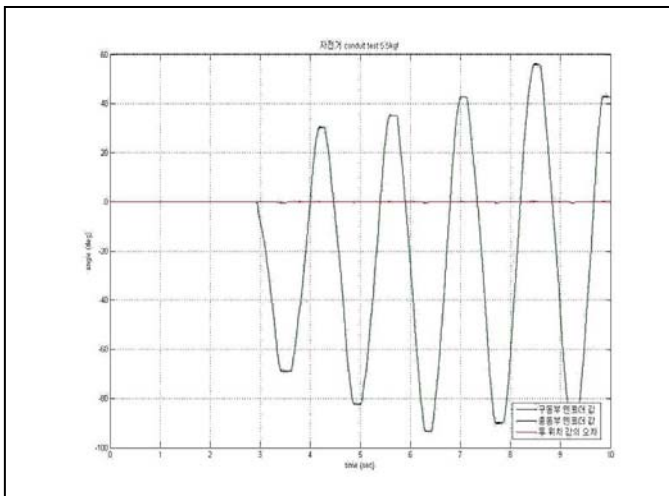


Fig. 6. Experimental test result of cable conduit system

The experimental test was performed by generating periodic and repetitive trajectory on driving pulley and measuring the following pulley connected cable-pulley system as shown in Fig. 5. In the experiment, this paper used conduit for bicycle break system which is a little high stiffness and lubricated. From the experimental test result shown in Fig. 6 it is found that there is not much effect of slacking which considered in chapter 2. Therefore, it is possible to adopt cable-conduit system in master console haptic handle for robot assisted laparoscope.

## 4 Conclusion

This paper studied to make light master handle design which can be used in already commercialized surgical robot system to apply haptic function. To facilitate

application of haptic system into commercialized system application, this paper suggests very similar structure with the end-effect part of instrument used in slave robot arm. Also, to prune the weight and inertia of handle so that it can generate minute force which can be felt by human, cable-conduit structure is adopted. Cable-conduit system can bring slacking phenomenon which can occur time delay and mis-measurement in master console handle system, this paper examined the performance using experimental test system. From the test result, it is proven that lubricated and a little stiffness conduit is adoptable to make up haptic handle. Although this study is in process, this paper proved the possibility of proposed method.

## References

- [1] Lee, M.C., Kim, C.Y., Yao, B., Penie, W.J., Song, Y.E.: Reaction Force Estimation of Surgical Robot Instrument Using Perturbation Observer with SMCSPO Algorithm. In: Proceedings of AIM 2010, Montreal, Canada, pp. 181–186 (July 2010)
- [2] Corcione, F., Esposito, C., Cuccurullo, D., Settembre, A., Miranda, N., Amato, F., Pirozzi, F., Caizzo, P.: Advantages and limits of robot-assisted laparoscopic surgery. *Surgical Endoscopy* 19(1), 117–119 (2002)
- [3] Giulianotti, P., Coratti, A., Angelini, M., Sbrana, F., Cecconi, S., Balestracci, T., Caravaglios, G.: Robotics in General Surgery- Personal Experience in a Large Community Hospital. *Archives of Surgery* (238), 777–784 (2003)
- [4] Vlachos, K., Papadopoulos, E., Mitropoulos, D.N.: Design and Implementation of a Haptic Device for Training in Urological Operations. *IEEE Transactions on Robotics and Automation* 19, 801–809 (2003)
- [5] Payandeh, S., Li, T.: Towards new designs of haptic devices for minimally invasive surgery. *International Congress Series*, pp. 775–781 (2003)
- [6] Hayward, V., Gregorio, P., Astley, O., Greenish, S., Doyon, M.: Freedom-7: A High Fidelity Seven Axis Haptic Deviec With Application to Surgical Training. In: *Experimental Robotics VI*, pp. 445–456 (1998)
- [7] Crisco, J.J., Heard, W.M.R., Rich, R.R., Paller, D.J., Wolfe, S.W.: The mechanical Axes of the Wrist Are Oriented Obliquely to the Anatomical Axes. *The Journal of Bone & Joint Surgery* 93, 169–177 (2011)
- [8] Agrawal, V., Penine, W.J., Yao, B.: Modeling of Transmission Characteristics Across a Cable-Conduit System. *IEEE Trans. on Robotics* 36(5), 914–924 (2010)
- [9] Jacobs, I.S., Bean, C.P.: Fine particles, thin films and exchange anisotropy. In: Rado, G.T., Suhl, H. (eds.) *Magnetism*, vol. III, pp. 271–350. Academic, New York (1963)
- [10] Elissa, K.: Title of paper if known (unpublished)
- [11] Nicole, R.: Title of paper with only first word capitalized. *J. Name Stand. Abbrev.* (in press)
- [12] Yorozu, Y., Hirano, M., Oka, K., Tagawa, Y.: Electron spectroscopy studies on magneto-optical media and plastic substrate interface. *IEEE Transl. J. Magn.* 2, 740–741 (1987); *Digests 9th Annual Conf. Magnetics Japan*, p. 301 (1982)
- [13] Young, M.: *The Technical Writer's Handbook*. University Science, Mill Valley (1989)

# Fusion of Inertial Measurements and Vision Feedback for Microsurgery

Yan Naing Aye, Su Zhao, Cheng Yap Shee, and Wei Tech Ang

School of Mechanical and Aerospace Engineering  
Nanyang Technological University  
Singapore  
yanna@ntu.edu.sg

**Abstract.** A microsurgery system that achieves real-time enhanced micrometer scale positioning accuracy by fusing visual information from a high speed monovision camera mounted on an optical surgical microscope and acceleration measurements from an intelligent handheld instrument, *ITrem2*, is presented. The high speed camera captures images of the tool tip of *ITrem2* to track its position in real-time. The focus value of the tool tip in the acquired image is used to locate the tool tip along the principal axis of the objective lens of the microscope and edge based geometric template matching gives the position in pixel coordinates. *ITrem2* utilizes four dual-axis miniature digital MEMS accelerometers to sense and update the motion information. The system has a first in, first out (FIFO) queue to track the recent history of the slow non-drifting position estimation from the vision system and acceleration readings from the inertial sensors together with their respective time stamps. In the proposed method, real-time visual servoing of micrometer scale motion is achieved by taking into account the dynamic behavior of the vision feedback and incorporating synchronized fusion of these complementary sensors.

**Keywords:** sensor fusion, visual servoing, microsurgery.

## 1 Introduction

Fusion of vision and inertial sensors has been used in many systems, especially in augmented reality applications. An important aspect of the fusion is to relate acquired information from multiple sensors which have different sampling rates. Redbinder [1] presented a method to fuse low bandwidth visual observations and high bandwidth rate gyro measurements to estimate the camera orientation. The system introduced a fixed time delay to the inertial measurements before performing sensor integration. With assumed periodic measurements, jitter was not considered. Armesto [2] presented a generic multi-rate sampling data system using size-varying input and output equations. Measurements need not be synchronous and periodic. The real-time issues were not taken into account. Jeroen *et al.* [3] described a tightly coupled real-time camera pose estimation supporting multi-rate signals synchronized by using one

central clock. While this method provides real-time performance, it is not suitable for loosely coupled systems where the camera cannot be mounted on the moving object.

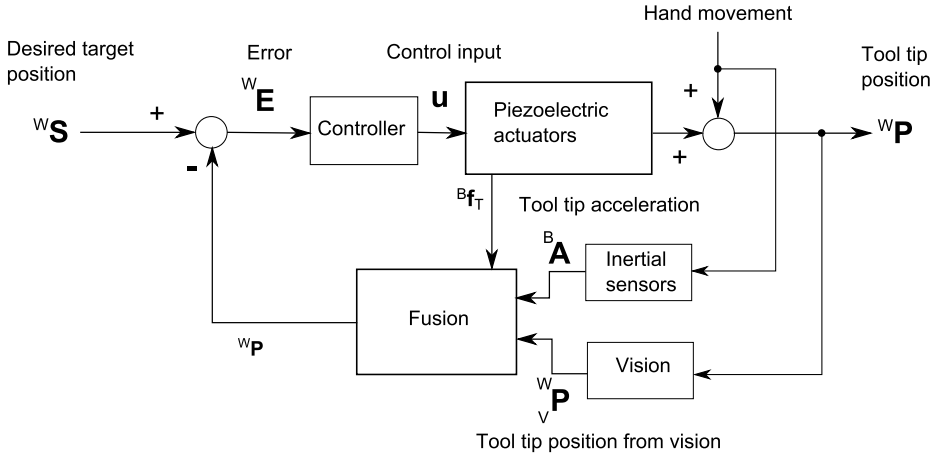
To reduce the amount of information that needs to be extracted from the vision system, Huster [4] described a method that generated useful vehicle trajectories to achieve more robust sensing. As a further step, to limit the error for a short period of missing vision measurements, Parnian [5] presented tool motion modeling. It tracked an industrial tool using vision and inertial sensors fusion. The tool motion modeling was proposed to bind the error in the acceptable range for a short period of missing data. These methods are appropriate for the applications where the nature of the movement of the object being tracked is deterministic. In the context of micromanipulation in microsurgery, physiological tremor is a well-studied hand movement. Several techniques for online modeling and estimation of tremor have been implemented [6,7,8].

The use of visual servo control is common in sensor fusion applications. Karras [9] demonstrated the fusion of a laser vision system and inertial measurement units. A visual servoing method was developed to control an underwater robotic vehicle. Sensor fusion for micromanipulation tasks was presented by Zhou [10]. In that paper, force and vision feedback were integrated for sensor-based microassembly. They discussed the implementation of an optical beam deflection sensor in a force controlled micropositioning systems. In contrast to other systems, it used force sensor as an alternative complementary sensor.

The approach described in this paper deals with the real-timeliness of a high performance visual servo control system that is to be used in the computer vision aided intelligent handheld instrument, *ITrem2*, to enhance human manual positioning accuracy at the micrometer scale in microsurgery. Vision measurements from a high speed monovision camera mounted on an optical surgical microscope and inertial information from *ITrem2* is combined and exploited to compensate the error caused by physiological hand tremor of a surgeon while nullifying common types of erroneous motion such as drift. *ITrem2* not only requires micrometer scale accuracy but also hard real-time prerequisite. The proposed method emphasizes on improving the feedback sensing by utilizing accurate time stamps of the vision and inertial measurements that are provided by the real-time computer. Time aware fusion of the inertial measurements provides micrometer accuracy even with significant delay and jitter in the vision measurements.

## 2 Methodology

The structure of the *ITrem2* system is shown schematically in Fig. 1.



**Fig. 1.** Position-based visual servo with fusion of inertial sensing

## 2.1 Vision System

The vision system of *ITrem2* constantly extracts the tool tip feature in each acquired image using edge based geometric template matching. The geometric matching gives the coordinate positions of the tool tip in sub-pixel accuracy.

There are several focusing algorithms to estimate the distance of the tool tip from the center of the objective lens based on the blurriness of an image [11]. Among them, Normalized Variance method is mentioned to provide the best overall performance. After calibration, the value of  ${}^wZ$  in the world coordinate frame can be solved on-line from focus value calculation of the tool tip using parabola-fitting approximation [12].

## 2.2 Inertial Measurement Unit

The rotation matrix,  ${}^wR_B$ , is used to transform the tool tip acceleration from the body coordinate frame to the world coordinate frame as shown in (1). It is updated at each sample point using *Euler Angle Sequence*.

$${}^wA = {}^wR_B {}^BA \quad (1)$$

$${}^wA = {}^wf_B({}^BA) \quad (2)$$

where  ${}^wf_B$  represents the function that performs coordinate transformation. Tilt sensing method using nonlinear regression model of low-g MEMS accelerometer described by Ang [13] is used to calculate the tilt and roll angles of *ITrem2*. Pan angle rotation is obtained from orientation of the tool tip image in the pixel frame.



### 2.3 Visual Servo Control

*ITrem2* uses position based visual servo control integrated with inertial sensing to fulfill the need for hard real-timeliness in microsurgery. The micromanipulator in *ITrem2* is a piezo actuated parallel flexural mechanism that generates the 3D motion so that erroneous hand motion can be compensated. For a point-to-point positioning task in which the tool tip at  ${}^W\mathbf{P}$  is to be brought to a desired target location,  ${}^W\mathbf{S}$ , in the world coordinate frame, the error function may be defined as

$${}^T\mathbf{E}({}^T\mathbf{f}_B) = {}^T\mathbf{f}_B({}^B\mathbf{f}_W({}^W\mathbf{P} - {}^W\mathbf{S})) \quad (3)$$

where the coordinate transformation from the tool tip frame to the body frame,  ${}^T\mathbf{f}_B$ , is the value to be controlled. The control input to be computed is the desired micromanipulator translational movement,  $\mathbf{u}$ , as shown in (4)

$$\mathbf{u} = -K {}^T\mathbf{E}({}^T\mathbf{f}_B), \quad (4)$$

where  $K$  is a proportional feedback gain. The proportional control law will drive the tool tip so that the value of the error function is zero.

### 2.4 Fusion Algorithm

Each time an image is acquired, the vision system stores the time stamp of the image with an accuracy of one microsecond. When the image processing and pose estimation are completed, the estimated world coordinate of the tool tip from vision,  ${}^W_v\mathbf{P}$ , the controlled value,  ${}^T\mathbf{f}_B$ , and the associated time stamp are sent to the fusion unit. The fusion unit has a first in, first out (FIFO) queue that stores the latest measurements from the accelerometers and associated estimated tool tip position,  ${}^W\mathbf{P}$ . Since the sampling rate of the vision system is much slower than that of the inertial measurement, without assuming periodicity, the fusion unit checks the availability of the vision information at each inertial measurement cycle. If new image information is available, the fusion unit performs the following steps to estimate the current position.

- 1) The position estimation of the tool tip from vision,  ${}^W_v\mathbf{P}$ , is associated with the corresponding acceleration measurement in the queue by calculating the delay time of the image and using the inertial sampling period  ${}_AT$ . Due to the considerable jitter in the sampling period of the vision system, the delay time is not constant and the exact value is calculated from the time stamp associated with the image.
- 2) If the piezoelectric actuators are turned off, the tool tip of the *ITrem2* will be at a neutral position. Using  ${}^W_v\mathbf{P}$  and  ${}^T\mathbf{f}_B$ , the neutral position of the tool tip of *ITrem2* in the world coordinate frame,  ${}^W_v\mathbf{P}'$ , is calculated.
- 3) The tool tip velocity of *ITrem2* is calculated using the last two positions obtained from vision measurements, which are illustrated in the Fig. 2.

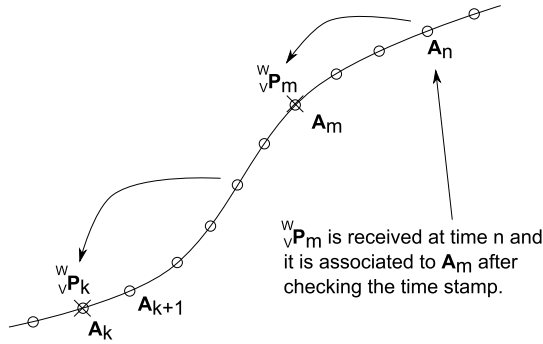


Fig. 2. Illustration of acceleration measurements and vision information

Initially,  $\Delta \mathbf{P}_k$  and  $\Delta \mathbf{V}_k$  are set to zero. Then,  $\Delta \mathbf{P}_m$  and  $\Delta \mathbf{V}_m$  are calculated iteratively from  $k+1$  to  $m$  using

$$\Delta \mathbf{P}_i = \Delta \mathbf{P}_{i-1} + {}_A T {}^w \mathbf{V}_{i-1} + \frac{1}{2} {}_A T^2 {}^w \mathbf{A}_{i-1}, \quad (5)$$

$$\Delta \mathbf{V}_i = \Delta \mathbf{V}_{i-1} + {}_A T {}^w \mathbf{A}_{i-1}, \quad i = k+1, \dots, m. \quad (6)$$

Thereafter, the tool tip velocity at time  $m$  is obtained from the following equation.

$$V_m = \frac{1}{(m-k)_A T} ({}^w_v \mathbf{P}'_m - {}^w_v \mathbf{P}'_k - \Delta \mathbf{P}_m) + \Delta \mathbf{V}_m \quad (7)$$

4) The measured tool tip position,  ${}^w_v \mathbf{P}'_m$ , and the predicted tool tip position in the queue,  ${}^w \mathbf{P}_m$ , are fused with the Kalman filter.

5) The tool tip position at time  $n$  is estimated iteratively from the new  ${}^w \mathbf{P}_m$ ,  $\mathbf{V}_m$ , and  $\mathbf{A}_m$  as follows.

$${}^w \mathbf{P}_i = {}^w \mathbf{P}_{i-1} + {}_A T {}^w \mathbf{V}_{i-1} + \frac{1}{2} {}_A T^2 {}^w \mathbf{A}_{i-1}, \quad (8)$$

$${}^w \mathbf{V}_i = {}^w \mathbf{V}_{i-1} + {}_A T {}^w \mathbf{A}_{i-1}, \quad i = m+1, \dots, n. \quad (9)$$

If there is no new vision information, the fusion unit simply updates the pose estimation for the current acceleration sample value using (8) and (9).

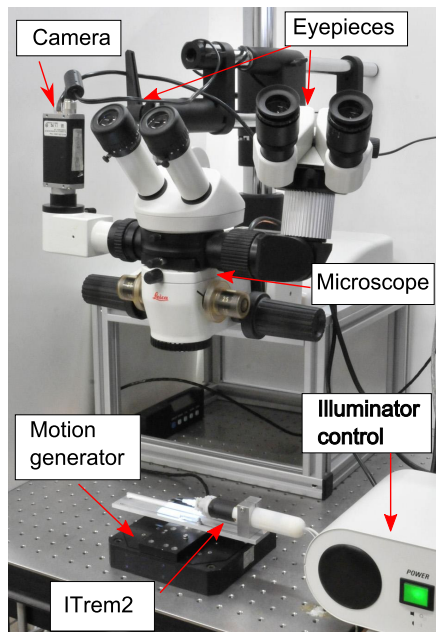
### 3 Experiment

#### 3.1 Setup

The experimental setup of *ITrem2* is shown in Fig. 3. The experiment is conducted to track the tremor like movement produced by a piezoelectric motion generator (Physik

Instrumente GmbH & Co.KG, Germany) comprising an E-712-3CDA digital controller and a P-561.3CD piezo-nanopositioning system. It can generate three DOF (degree-of-freedom) linear motion of up to 150 $\mu$ m peak to peak in each axis. It also provides analog output lines to monitor the position of each axis and these are used as the ground truth for the experiment. *ITrem2* is mounted on the motion generator with its inertial sensing axes aligned with the corresponding axes of the motion generator.

The vision system consists of an optical surgical microscope (M651 MSD, Mikrosysteme Vertrieb GmbH, Germany) with a built-in coaxial illuminator and retrofitted with a monovision camera (piA640 -210gm/gc camera from Basler AG, Germany). The microscope is equipped with a beamsplitter and a stereo attachment for a second observer. Therefore the workspace can be viewed simultaneously by the camera, a surgeon, and an assistant. The monovision camera provides the workspace view of 5mm x 3.7 mm and its sampling rate is 50Hz. Image processing is performed in real-time using the NI PXIe-8130 real-time embedded computer running on LabVIEW real-time operating system connected to the camera via a Gigabit ethernet interface. IMAQ Vision for LabVIEW is used to implement the edge based geometric template matching.



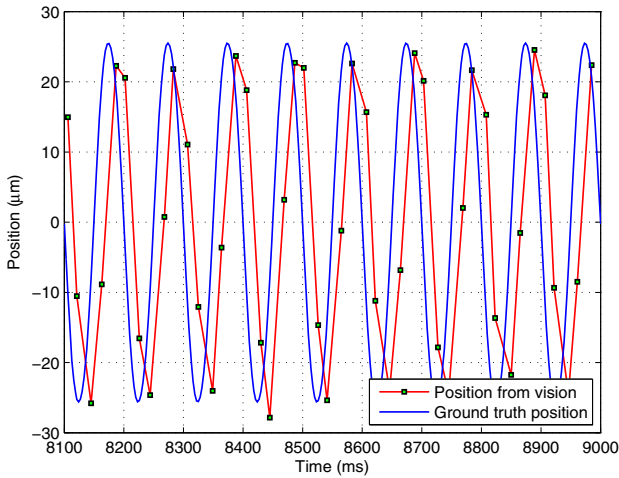
**Fig. 3.** Experiment setup of microscope, camera and motion generator

There are four dual-axis miniature digital MEMS accelerometers (ADIS16003, Analog Devices, USA) inside *ITrem2*. They sense three DOF motion of the instrument tip using the method similar to its predecessor [14]. After performing a moving average filtering, the embedded microcontroller (AT89C51CC03, Atmel, USA) inside *ITrem2* sends the acceleration data to the real-time computer at 333

samples per second. The CAN (Controller-area network) interface with a bandwidth of 500kbps is used to achieve robust and real-time communication between the embedded microcontroller and the real-time computer. Three DOF motion of the micromanipulator in *ITrem2* is generated by the piezo actuated parallel flexural mechanism. Maximum movement of the micromanipulator along X-axis and Y-axis is  $150\mu\text{m}$  and that of Z-axis is  $28\mu\text{m}$ .

### 3.2 Results

To assess the performance of the fusion algorithm, the experiment was carried out using constant  ${}^T\mathbf{f}_B$  i.e.  $K=0$ . Therefore, the error in (3) is uncorrected and the output of the vision feedback can be compared with the ground truth position of the motion generator. The sensing experiment is conducted using a sinusoidal movement of  $51\mu\text{m}$  peak-to-peak at 10Hz along the X-axis of the world coordinate frame.



**Fig. 4.** Position information using vision only

There is an obvious delay in position information from the vision system compared to the ground truth positions from motion generator as shown in Fig. 4. The estimated positions from the fusion unit versus the ground truth positions are shown in Fig. 5.

To measure the accuracy of the fusion algorithm under different frequencies, the experiment was also performed with various sinusoidal movements from 2Hz to 8Hz at intervals of 2Hz. The RMSE of the vision measurements is smaller at the lower frequencies. Probable causes are larger phase lag and greater motion blur of the tool tip image at the higher frequencies. On the other hand, the signal-to-noise ratio of the acceleration measurements is better at the higher frequencies. The results shown in Table I confirm that the fusion of these complementary sensors provides lower RMSE at various frequencies from 2Hz to 10Hz.

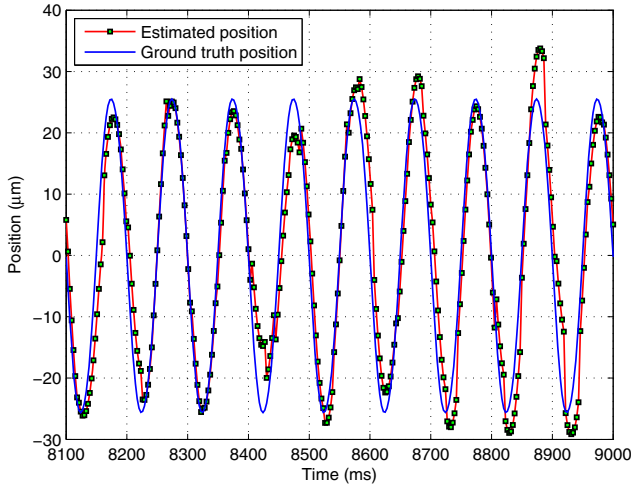


Fig. 5. Estimated positions using the fusion algorithm

Table 1. Accuracy at different frequencies

Frequency (Hz)	Peak-to-peak displacement ( $\mu\text{m}$ )	RMSE using vision only ( $\mu\text{m}$ )	RMSE using Fusion ( $\mu\text{m}$ )	Error reduction (%)
10	51	19.12	6.09	68.12
8	56	17.19	5.90	65.68
6	60	13.98	5.30	62.08
4	63	9.97	5.23	47.54
2	64	5.25	4.95	5.71

## 4 Conclusion

This paper presents an algorithm for real-time fusion of visual and inertial sensors. With the modified feedback in the visual servo control to take into account delay and jitter of the vision system, the system substantially improves sensing speed and accuracy for microscopic applications. It is intended to cancel rapid erroneous hand movement caused by tremor and, at the same time, to provide absolute position for automatic micromanipulation tasks.

**Acknowledgements.** Vision-Aided Active Handheld Instrument for Microsurgery project is funded by Agency for Science, Technology & Research (A\*STAR) and the College of Engineering, Nanyang Technological University. The authors thank them for the financial support of this work.

## References

1. Rehbinder, H., Ghosh, B.K.: Multi-rate fusion of visual and inertial data. In: International Conference on Multisensor Fusion and Integration for Intelligent Systems, MFI 2001, pp. 97–102 (2001)
2. Armesto, L., Chroust, S., Vincze, M., Tornero, J.: Multi-rate fusion with vision and inertial sensors. In: Proceedings of Robotics and Automation, ICRA 2004, April 26 -May 1, vol. 1, pp. 193–199 (2004)
3. Hol, J.D., et al.: Robust real-time tracking by fusing measurements from inertial and vision sensors. *Journal of Real-Time Image Processing* 2, 149–160 (2007)
4. Huster, A., Frew, E.W., Rock, S.M.: Relative position estimation for AUVs by fusing bearing and inertial rate sensor measurements. In: MTS/IEEE OCEANS 2002, October 29–31, vol. 3, pp. 1863–1870 (2002)
5. Parnian, N., Golnaraghi, M.F.: Integration of vision and inertial sensors for industrial tools tracking. *Sensor Review* 27(2), 132–141 (2007)
6. Tan, U.-X., Veluvolu, K., Latt, W.T., Shee, C.Y., Riviere, C., Ang, W.T.: Estimating displacement of periodic motion with inertial sensors. *IEEE Sensors Journal* 8, 1385–1388 (2008)
7. Veluvolu, K.C., Ang, W.T.: Estimation of physiological tremor from accelerometers for real-time applications. *Sensors* 11(3), 3020–3036 (2011)
8. Latt, W.T., Veluvolu, K.C., Ang, W.T.: Drift-free position estimation of periodic or quasi-periodic motion using inertial sensors. *Sensors* 11(6), 5931–5951 (2011)
9. Karras, G.C., Loizou, S.G., Kyriakopoulos, K.J.: A visual-servoing scheme for semi-autonomous operation of an underwater robotic vehicle using an IMU and a Laser Vision System. In: Robotics and Automation (ICRA), May 3–7, pp. 5262–5267 (2010)
10. Zhou, Y., Nelson, B.J., Vikramaditya, B.: Fusing force and vision feedback for micromanipulation. In: Proceedings of 1998 IEEE International Conference on Robotics and Automation, May 16–20, vol. 2, pp. 1220–1225 (1998)
11. Sun, Y., Duthaler, S., Nelson, B.J.: Autofocusing Algorithm Selection in Computer Microscopy. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, August 2–6, pp. 70–76 (2005)
12. Wu, Q., Merchant, F.A., Castleman, K.R.: *Microscope Image Processing*. Academic Press (2008)
13. Ang, W.T., Riviere, C.N., Khosla, P.K.: Nonlinear regression model of a low-g MEMS accelerometer. *IEEE Sensors J.* 7(1&2), 81–88 (2007)
14. Latt, W.T., Tan, U.-X., Shee, C.Y., Riviere, C.N., Ang, W.T.: Compact Sensing Design of a Hand-held Active Tremor Compensation Instrument. *IEEE Sensors Journal* 9, 1864–1871 (2009)

# A New Concept for a “Vaginal Hysterectomy” Robot

Kovit Khampitak<sup>1</sup>, Wathanyu Neadsanga<sup>2</sup>, Sirivit Taechajedcadarung-sri<sup>1</sup>,  
and Thantakorn Pongpimon<sup>1</sup>

<sup>1</sup> Khon Kaen University

<sup>2</sup> Rajamangala University of Technology Isan,

Khon Kaen 40002 Thailand

kovit@kku.ac.th

**Abstract.** The design concept of a novel vaginal hysterectomy robot which is composed of three compound robots is discussed. The results of *in vitro* mechanical evaluation of the first two prototypes are reported and ideas for future development discussed.

**Keywords:** Surgical robot, Medical robot, Hysterectomy robot.

## 1 Introduction

Nowadays, the Da Vinci surgical system, approved by the FDA in September 2001, is the state-of-the-art surgical technology [1]. Its disadvantages are its higher costs and longer operating room times [2]. In addition, none of the existing robotic systems are equipped to provide high quality feedback; this is of major concern as errors can have potentially devastating consequences [3].

Many different types of small task specific medical robots are currently being developed in many institutes. For example, a robot that provides assisted motion for knee and shoulder joints is gaining wide acceptance [4]. A robot-controlled fluid system was introduced to detect *ex vivo* breast cancer chemotherapy sensitivity [5].

In surgery, the SpineAssist robot has shown a high level of accuracy *in vivo* for the implantation of lumbar pedicle screws [6]. A dexterous miniature robot for natural orifice transluminal endoscopic surgery has been tested in multiple animal model studies [7] and also been performed successfully in humans, including appendectomy and cholecystectomy [8-10].

Hysterectomy is the most common non-pregnancy-related surgery, with approximately 600,000 procedures performed annually in the USA [11]. Vaginal hysterectomy, a natural orifice procedure, has been reported as a standard hysterectomy which the advantage of less pain and rapid recovery [12]. However, before performing this procedure a new surgeon needs a period of learning from an expert. In order to simplify this procedure we have designed a prototype of a compound multiple simultaneous vaginal hysterectomy robot and subjected it to mechanical tests. Once perfected, we hope this robot could help the young

gynecologists to perform vaginal hysterectomies with the same speed and success rate achieved by an expert.

## 2 Design Concept and Procedure

The compound vaginal hysterectomy robot is composed of three small robots working simultaneously; ligaments and vessels cutting, uterine bisecting and vaginal cuff sealing robots. The mechanical joints were located in a common body with motions controlled by external moving slings.

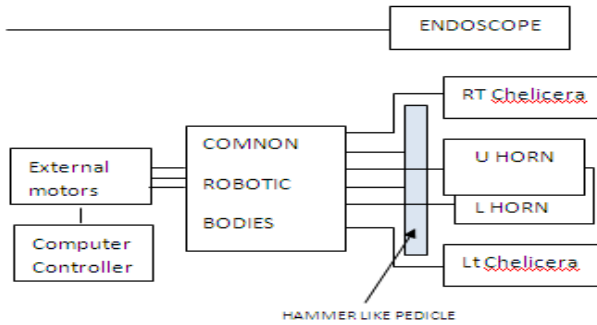


Fig. 1. Diagram showed a compound vaginal hysterectomy robot

### 2.1 Ligaments and Vessels Cutting Robot

In the first scorpion-like robot, the right and left chelicerae could move with two DOFs diving by two sling sheaths. The robot was able to grip and manipulate the uterine suspensory ligaments and vessels at the parametrium to free the uterus from human cavity.

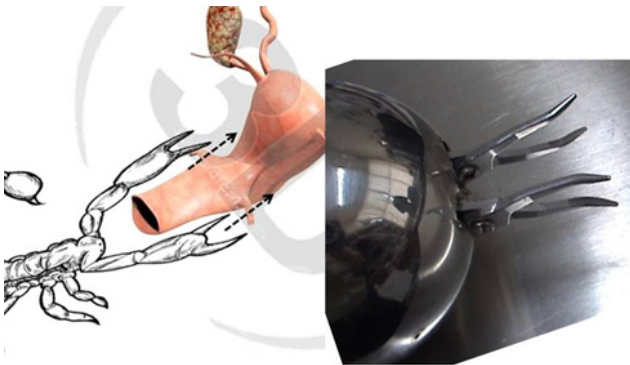


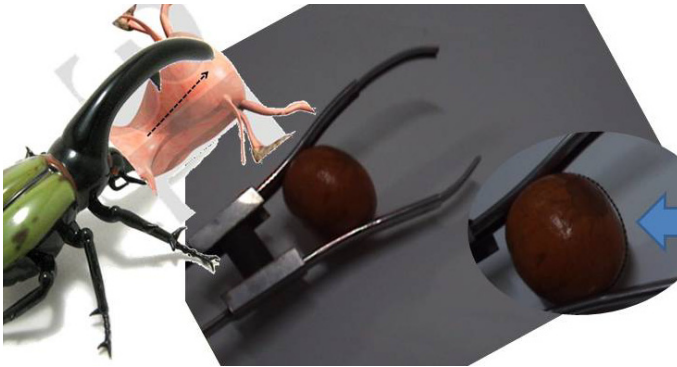
Fig. 2. The right and left chelicerae of the scorpion-like robot were able to grip and manipulate the ligaments and vessels at parametrium. (black arrow)



## 2.2 Uterine Bisecting Robot

A Dynastes Hercules-like robot was designed to manipulate the uterus, working simultaneously with the first robot. It could move its upper and lower horns (two DOFs ) by opening and closing their joint.

During the large uterine operation this robot could bisect the large uterus with a cutting saw. Each horn was composed of an inner longitudinal groove containing a movable saw. The saw is moved by a power sling and sheath from an external motor. It saws the uterus into two smaller pieces which can then be easily removed through the vaginal canal.



**Fig. 3.** A Dynastes Hercules-like robot was designed to manipulate and cut the uterus simultaneously with the work of the first robot. During the large uterine operation this robot can bisect the uterus using a cutting saw inside the sulcus. (arrow)

## 2.3 Vaginal Cuff Sealing Robot

This is a robot with two hammer-like pedicles, containing suture material. It could work automatically after the start of the process. We plan to produce a first prototype in the near future.



**Fig. 4.** Vaginal cuff sealing robot

### 3 Experiments

The tests occurred in October 2011 in Khon Kaen University, Thailand. The purpose was to test the mechanical movement of the first prototypes of the ligaments and vessels cutting and uterine bisecting robots.

#### 3.1 The Mechanical Test of the Ligaments and Vessels Cutting Robot

Ten pieces of fresh raw meat size 5x5x0.5 cm<sup>3</sup> were used in this experiment. A 3 minute gripping test was performed to compare between the robotic chelicerae and traditional Heaney clamps. The depths of tissue collapse were recorded by visual ruler and the results subject to statistical analysis. Table I shows that there were no statistically significant differences between the results achieved with the two devices.

**Table 1.** The gripping test compared between the robotic chelicerae and traditional Heaney clamps

Experiment	Gripping test of tissue collapse(mm)		
	<i>The robotic chelicerae</i>	<i>Traditional Heaney clamps</i>	<i>p.</i>
		3	
1	2	3	> .05
2	3	3	
3	3	3	
4	3	2	
5	2	2	
6	3	3	
7	3	3	
8	3	3	
9	2	3	
10	2	3	
		2	

No statistical difference by Wilcoxon Signed Rank test\*

#### 3.2 The Mechanical Test of the Uterine Bisecting Robot

The mechanical bisecting experiment was performed using apple fruit as the specimens. In three experiments, one achieved complete bisection of the fruit while the other two trials resulted only in partial bisections of the apples. The problems were caused by instability of the sling motion and the lack of stability of the specimens during the cutting process.

### 4 Discussion

Nowadays, the new technology using robotic surgery and other automated devices, is gradually penetrating our practice. The steps of development of the surgical robots could be divided into three essential stages.

**The first stage is “Robot assisted human surgeries”.** The DaVinci Model is already on the market and shows many benefits in assisting surgeries. The next models, which are now being developed in many laboratories around the world will be quite small in size and have a quick set-up time. However, these robots still need human to operate in the external console, their operators require considerable training to use them, and they are still expensive.

**The second stage is “Human assisted robotic surgery”.** The surgical procedures for the robots need to be simplified so that the robot could easily operate under human guidance. The vaginal hysterectomy robot being developed in our laboratory is small sized and designed specifically for this type of operation. The surgeon should be able to control the robot simply by giving permission for each of operating steps. After it is completely developed, the robot could help young gynecologists to operate as if they are experienced experts. In addition, they might do the vaginal hysterectomy in less than 30 minutes.

**The last stage will be “Real robotic surgery”.** The robots will be able to perform the operation with limited or even no human control. In this stage the major development trends will be; 1) miniaturization and augmented reality [13, 14], 2) automation [15] and 3) no theatre operation or remote controlled telesurgery.

## 5 Future Work

In further work, the prototype of the third robot will be produced. Then, the first and second robots will be combined with it to investigate the compound device’s functioning and control system. Finally, we hope to develop the vaginal hysterectomy robot so as to miniaturize it. We also hope to give it augmented reality and create a semiautomatic robot.

## 6 Conclusion

In this paper, we reported our tests of two prototypes of a vaginal hysterectomy robot which are designed to perform the procedures of the traditional vaginal hysterectomy. The experiments showed no different in gripping power between traditional method and robotic power. However, the bisection processes need to be revised and re-evaluated.

**Acknowledgements.** This research was performed at Khon Kaen University (KKU) with financial support from the KKU research fund. The authors wish to thank the Head of the Department of Obstetrics and Gynecology, Faculty of Medicine for giving permission to conduct this study and Prof. Dr.Terry Rambo for English correction.

## References

1. Ayav, A., Bresler, L., Brunaud, L., et al.: Early results of one-year robotic surgery using the Da Vinci system to perform advanced laparoscopic procedures. *J. Gastrointest. Surg.* 8(6), 720–726 (2004)
2. Turchetti, G., Palla, I., Pierotti, F., et al.: Economic evaluation of da Vinci-assisted robotic surgery: a systematic review. *Surg. Endosc.* 26(3), 598–606 (2012)
3. Tavakoli, M., Aziminejad, A., Patel, R.V., et al.: Methods and mechanisms for contact feedback in a robot-assisted minimally invasive environment. *Surg. Endosc.* 20(10), 1570–1579 (2006)
4. Wilkening, A., Baiden, D., Ivlev, O.: Assistive control of motion therapy devices based on pneumatic soft-actuators with rotary elastic chambers. *IEEE Int. Conf. Rehabil. Robot* 2011, 1–6 (2012)
5. Kleinhans, R., Brischwein, M., Wang, P., et al.: Sensor-based cell and tissue screening for personalized cancer chemotherapy. *Med. Biol. Eng. Comput.* 50, 117–126 (2012)
6. Sukovich, W., Brink-Danan, S., Hardenbrook, M.: Miniature robotic guidance for pedicle screw placement in posterior spinal fusion: early clinical experience with the SpineAssist. *Int. J. Med. Robot* 2(2), 114–122 (2006)
7. Lehman, A.C., Wood, N.A., Farritor, S., et al.: Dexterous miniature robot for advanced minimally invasive surgery. *Surg. Endosc.* 25(1), 119–123 (2012)
8. Kalloo, A.N., Singh, V.K., Jagannath, S.B., et al.: Flexible transgastric peritoneoscopy: a novel approach to diagnostic and therapeutic interventions in the peritoneal cavity. *Gastrointest. Endosc.* 60(1), 114–117 (2004)
9. Lima, E., Henriques-Coelho, T., Rolanda, C., et al.: Transvesical thoracoscopy: a natural orifice transluminal endoscopic approach for thoracic surgery. *Surg. Endosc.* 21(6), 854–858 (2007)
10. Raman, J.D., Bergs, R.A., Fernandez, R., et al.: Complete transvaginal NOTES nephrectomy using magnetically anchored instrumentation. *J. Endourol.* 23(3), 367–371 (2009)
11. Wu, J.M., Wechter, M.E., Geller, E.J., et al.: Hysterectomy rates in the United States. *Obstet. Gynecol.* 110(5), 1091–1095 (2007)
12. Jahan, S., Das, T.R., Mahmud, N., et al.: A comparative study among laparoscopically assisted vaginal hysterectomy, vaginal hysterectomy and abdominal hysterectomy: experience in a tertiary care hospital in Bangladesh. *J. Obstet. Gynaecol.* 31(3), 254–257 (2012)
13. Marescaux, J., Solerc, L.: Image-guided robotic surgery. *Semin. Laparosc. Surg.* 11(2), 113–122 (2004)
14. Nishihara, S., Sugano, N., Nishii, T., et al.: Clinical accuracy evaluation of femoral canal preparation using the ROBODOC system. *J. Orthop. Sci.* 9(5), 452–461 (2004)
15. Menciassi, A., Quirini, M., Dario, P.: Microrobotics for future gastrointestinal endoscopy. *Minim. Invasive Ther. Allied Technol.* 16(2), 91–100 (2007)

# Classification of Modeling for Versatile Simulation Goals in Robotic Surgery

Stefan Jörg, Rainer Konietzschke, and Julian Klodmann

Robotics and Mechatronics Center, German Aerospace Center (DLR),  
Wessling, Germany  
stefan.joerg@dlr.de

**Abstract.** Simulation is common practice for surgeon training in particular for robotic surgery. This paper introduces further relevant applications of simulation that improve patient safety. Therefore, the design of a modular simulator for minimally invasive robotic surgery is presented. The authors introduce a classification of hierarchical levels of modeling details for the three aspects Application, System, and Patient. Furthermore, the principal use case classes Training, Workflow Validation, Workflow Design, Monitoring, and Robot Design of simulation for robotic surgery are introduced. For each class standard simulator setups are presented. The application of the classification is exemplified for the use cases Training and Robot Design.

**Keywords:** surgical robotics, simulation, patient safety.

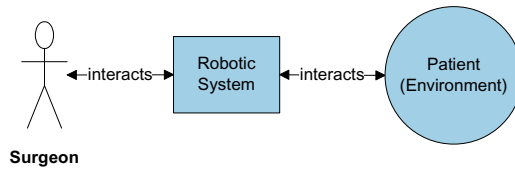
## 1 Introduction

It is common practice, to use the method of simulation to train surgeons with the aim to improve patient safety [3] [6] [13]. Training has been recognized as being even more important when surgical robotic systems are involved as their use requires new skills [8]. Other than training, there are more useful applications of surgical simulation that increase patient safety in the context of robotic surgery, e.g. preoperative planning or workflow optimization.

The intention of this paper is to introduce the relevant applications of simulation for robotic surgery and to describe their relation to patient safety. In particular, the relevant properties of simulation in robotic surgery are identified and then used to classify the various simulation types. The effort of modeling and implementation for the various simulation types is tackled by the introduction of levels of modeling details and a modular simulator architecture.

As robotic surgery involves complex technical systems the authors regard the simulation thereof from a systems engineering point of view. The System Modeling Language SysML is an industry standard for the specification of such systems [12]. Following SysML, we define a *use case* as the description of the intended *goal* of a simulation.

To establish a common ground, the definition of a Simulator for Robotic Surgery is given in the following section. Based on this, the main components



**Fig. 1.** The Robotic System Simplifies the Interface Between Surgeon and Environment. The Surgeon Only Interacts with the Robotic System.

of a simulator for robotic surgery are introduced. Sec. 3 introduces the concept for approximation of the real world with levels of details with the aim to optimize the modeling and simulation effort for a given *use case*. Sec. 4 introduces the principal use case classes of simulation. Sec. 5 exemplifies the setup of the simulator.

## 2 A Simulator for Robotic Surgery

In this section the definition of a simulator for robotic surgery is presented. This definition will be later used for the discussion of the relevant use cases and the simulator setups related to that.

There are various definitions of simulation in the literature that reflect the broad application domains of this method [5] [14]. We use the more formal definition of *simulation* as observation or execution of a model over time. A *model* is an approximate representation of a real-world system. A *simulator* implements and executes a model to perform a simulation. *Analysis* is the process to verify and validate the simulation result. Thus, the method of simulation involves the tasks to model, simulate, and analyze [14].

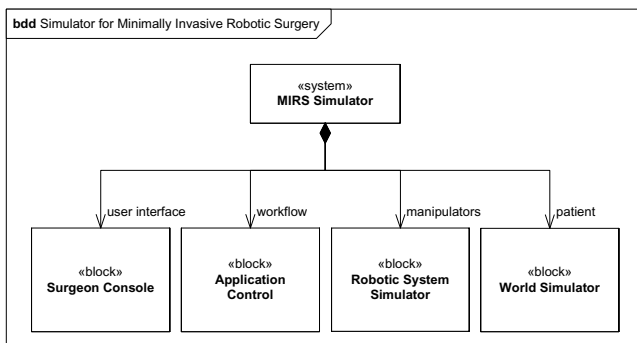
A simulator interacts with an *entity-under-test*. If the entity is human, the simulation is called *interactive* (human-in-the-loop). Surgeon training is a well-known example of interactive simulation.

Simulation is *real-time* when the simulation time advances with the physical time of the real-world. If the simulator interacts with a real-world system the simulation has to be real-time. Hence, interactive simulation is always real-time.

A model is *discrete-event* or *continuous-time*. A discrete-event model updates its state upon the occurrence of an event. In general, events occur aperiodically. In contrast to that, a continuous model defines its state as a continuous function of time, e.g. as differential equations. Advanced simulations of real-world systems use a combination of both [5].

Practically, the *implementation* of a simulation of a *continuous-time* model approximates the continuous states at periodic steps in simulation time, i.e. utilizes a *discrete-time* model. Similar to that robotic systems are implemented as discrete signal systems [7]. This similarity can be leveraged to use the same implementation of control laws for the simulation and the real system.

In robotic surgery, the robotic system is usually the main way a surgeon interacts with the patient. Therefore, the focus of this paper is on models where



**Fig. 2.** A Modular Simulator for Minimally Invasive Robotic Surgery (MIRS)

only the robotic system is the intermediate between surgeon and environment (see Fig. 1). This simplifies the user interface. Instead of various surgeon tools only the user console of the robotic system interfaces directly with the surgeon. Especially, for the tele-robotic setups of minimally invasive surgery it is natural to use the same interface for simulation and real-world scenarios 1.

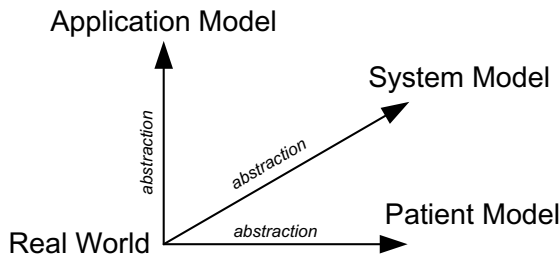
Fig. 2 depicts the main components of a simulator for minimally invasive robotic surgery. The four components Surgeon Console, Application Control, Robotic System Simulator and World Simulator partition the model of the surgical robotic system into corresponding classes: User interface, workflow, manipulators, and patient.

This modularity of the simulator can be used to adapt the simulator to various use cases. The simulator’s use case determines the details of each model (its relevant aspects) and the interface between each component (exchange of data between models). Level of detail introduces a classification of aspects of a model’s approximation of the real world. The following section discusses the main aspects and introduces corresponding hierarchies of abstraction levels.

### 3 Level of Modeling Details

A model is an approximation of the real world. Levels of modeling details introduce a classification of this approximation. The main purpose of the introduction of levels of modeling details is the scaling of modeling and computation effort according to the simulation’s use case. Hence, a proper classification has to grasp the relevant issues of the simulator’s application domain. We refer to these relevant issues as the aspects of the model. For robotic surgery, we identified three relevant aspects of the model:

- The Application Model captures the surgical procedure (*Application-Centric*)
- The System Model captures the implementation of the robotic system (*System-Centric*)



**Fig. 3.** Three Distinct Aspects of Simulation for Robotic Surgery are Application, System, and Patient. Together, they span the modeling space for robotic surgery.

- The Patient Model captures the environment with the focus on the patient (*Patient-Centric*)

Depending on the simulation’s use case, each of these aspects is of more or less interest, i.e. the approximation needs to be more or less realistic for a certain simulation goal. To adapt the simulation model to a certain use case the details of the surgical procedure model, of the robotic system model, and of the patient model can be scaled independently (see Fig. 3).

The classification of each aspect as a hierarchy of abstraction levels facilitates the adaption of the simulation model to a certain use case. Starting at the most abstract level each successive level adds a new detail to the model. Finally, Level 0 represents the real-world. The presented hierarchies divide each aspect into four levels of modeling detail. The criteria for the discrimination between levels are: First, the importance of a distinction in real-world terms (e.g. anatomy-true or case-specific data). Second, the existence of a distinct step of complexity in terms of modeling (e.g. rigid body or deformable objects). As shown in Fig. 6, this classification can be improved by a further refinement within each level. In the following, the hierarchies for the three aspects are introduced.

### 3.1 Application-Centric Modeling Hierarchy

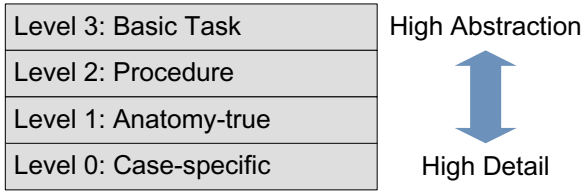
The aspect of application considers how detailed the surgical procedure is modeled. Typically, a surgical procedure is a workflow of tasks or goals. Consequently, the most abstract view of a procedure considers only tasks and abstracts the existence of a workflow. At the lower levels the procedure is determined by anatomy or even patient-specific parameters such as geometric constraints (workspace, path of instruments, etc.).

*Level 3 Basic Task:* On this level, the application model captures only basic tasks, such as suturing or cutting. There is no workflow and no surgical goal.

*Level 2 Procedure:* The procedure level adds the workflow to the basic tasks. This includes a surgical goal.

*Level 1 Anatomy-true:* This level adds the details of human anatomy, such as organs and vessels.



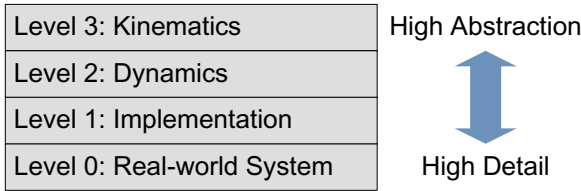


**Fig. 4.** Application-Centric Levels of Modeling Details

*Level 0 Case-specific:* This level adds the specific data and anatomy parameters of a certain case (patient).

### 3.2 System-Centric Modeling Hierarchy

The system model has the focus on the modeling details of the implementation of the surgical robotic system. This refers to the main components of a surgical simulator as depicted in Fig. 2. Therefore, for each level the impact on each component is discussed, except the Application Control component which is mainly affected by the Application-Centric aspect.



**Fig. 5.** System-Centric Levels of Modeling Details

*Level 3 Kinematics:* Only the kinematical aspects are modeled on this level, dynamic effects are not considered (e.g. masses, inertia, and forces). **Surgeon Console:** User input is simulated and limited to motion, e.g. playback of pre-recorded user input or dedicated test patterns. The **robotic system** is modeled as an ideal positioning device: It follows its desired trajectory instantly. **World:** Manipulation of objects in the environment is limited to motion.

*Level 2 Dynamics:* This level adds dynamics, i.e. the effects of masses, forces, and torques. Relevant dynamic effects of implementation properties are also modeled on this level, e.g. torque ripple, latency, friction. **Surgeon Console:** Simulated user input is augmented by the dynamic properties of input devices. Thus, the effect of input devices on the system, e.g. its control laws, etc., can be investigated. **Robotic System:** The movement of the robot is constrained by its dynamics (e.g. inertia, maximum motor torque, friction, etc.) and by external forces of the environment. An accurate dynamics model of the robotic system

is necessary for this [10]. **World:** Interaction of objects (including the robot) in the environment is modeled with forces and torques.

*Level 1 Implementation:* This level adds the cycle-true computation of discrete signal systems. The cyclic execution at discrete points in time is an approximation of the continuous physical world, i.e. of *continuous-time* models. Thus, the effects on the system’s performance due to this discrete approximation can be incorporated to the simulation. Level 1 is useful to simulate the behavior of implementations of complex systems. Details of the mechatronic implementation are added to the simulation. These include communication errors and delays, computation time, quantization of sensor values. **Surgeon Console:** Mechatronic implementation of input and feedback devices. **Robotic System:** Mechatronic implementation of the robotic system. **World:** There are no implementation details of the environment.

*Level 0 Real System:* This level represents the real-world system, e.g. the robot and its tools, input devices, GUIs, 3D-Displays, the patient, animal models or phantoms.

The three levels of kinematics, dynamics and implementation cover the most relevant details of simulation in descending order. It is reasonable to refine this main classification within each level. Especially Level 2, the dynamics, has varying aspects of interest depending on the simulation’s use case. Fig. 6 depicts a possible hierarchy within Level 2.

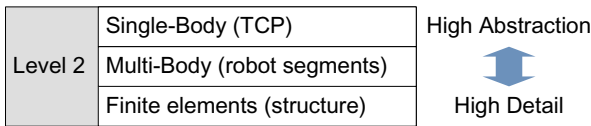


Fig. 6. Reasonable Levels of Modeling Details within Level 2 (Dynamics)

### 3.3 Patient-Centric Modeling Hierarchy

The patient model regards the details of the environment with the focus on the patient. Basdogan et al. [6] state that versatility in patient modeling, finding the right realism and integrating material properties of organs are all important issues for surgical simulation. The following hierarchy defines three major abstraction steps of the environment model.

*Level 3 Rigid Body:* On this level, the environment is modeled as rigid body geometry with stiff object contacts.

*Level 2 Soft Tissue:* This level adds the detail of material properties to the modeled objects (e.g. soft tissue for organs).

*Level 1 Functional:* This level adds functions to the tissue, i.e. blood flow, objects are cuttable, tear-able.

*Level 0 Real-World:* The patient, a phantom or an animal.



Fig. 7. Patient-Centric Levels of Modeling Details

### 4 A Classification of Principal Use Cases

Starting the classification from a systems engineering point of view the principal use case classes of simulation for robotic surgery are discussed in this section. As robotic surgery involves sophisticated mechatronic systems it is reasonable to derive the categories of application from that domain. Important system engineering processes are design/implementation, validation and verification [2]. Including training from the domain of surgery, the four main categories of goals for simulation in the context of robotic surgery are:

*Training* - goal of training is to use the method of simulation to improve the skills of involved humans.

*Design* - goal is to use simulation for the design process of a system. This includes the robotic system, the design of a surgical procedure, etc.

*Validation* - goal is to identify the proper system or task specifications.

*Verification* - goal is to use simulation to determine whether a system or procedure works as specified.

In the following, the principal use case classes for simulation of robotic surgery are discussed and related to the four categories. Figures 8-13 show the configuration of the modular simulator, introduced in Section II, for the principal use case classes.

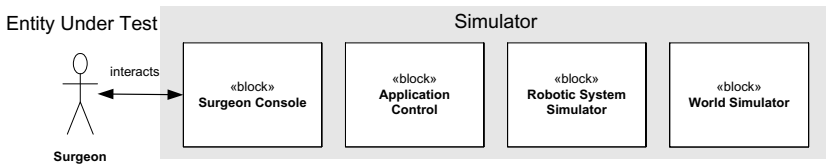


Fig. 8. Simulator Setup for the Class Surgeon Skill Training

*Use Case Class Surgeon Skill Training* (Category: Training) It is common practice to use simulation to train surgeon skills, either on a certain surgical procedure or to operate the robotic system. **Goal:** Improve skills of surgeons. **Typical Setup:** Requires all four components of the simulator (see Fig. 8). The surgeon is the entity-under-test. Hence, the simulation is always interactive

and accordingly real-time. **Patient safety** is immediately affected by surgeon skills. However, a training curriculum is required that proves to enhance surgeon skills [4].

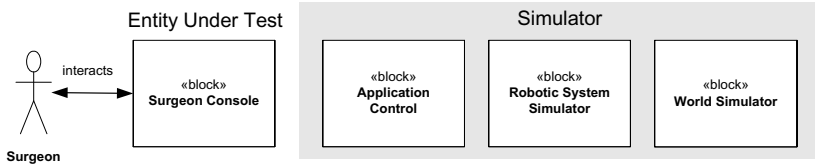


Fig. 9. Simulator Setup for the Class User Interface Designs

*Use Case Class User Interface Design* (Category: Design, Validation) **Goal:** Improve and validate the design of the surgeon console. **Typical Setup:** The simulator (see Fig. 8) consists of Application Control, Robotic System and World simulators. The Surgeon Console is the entity-under-test. **Patient Safety:** A validated user interface reduces the risk of mal-operation. An improved user interface puts the surgeon’s focus on the surgical procedure. It provides the surgeon with relevant information only and alerts him of impending risks.

*Use Case Class Workflow Validation* (Category: Validation). Workflow falls into two categories: Surgical Procedure and Technical Workflow. The former has the focus on the surgical goal, the latter on the operation of the robotic system. **Goal:** Validate a given workflow with human-centered (usability) methods. **Typical Setup:** The simulator setup (see Fig. 10) is similar to the class User Interface Design, except that Application Control is the entity-under-test. **Patient safety:** A workflow design that has been validated with the surgeon-in-the-loop reduces the risks for the patient because the workflow has been proven to be more likely performed well by surgeons.

*Use Case Class Workflow Design* (Category: Design). **Goal:** Improve the design of a surgical or technical workflow. Pre-operative planning falls into this class. **Typical Setup:** Requires both the robotic system and the world simulators. Application Control is the entity-under-test (see Fig. 11). **Patient safety:** Improved workflow for a certain procedure (e.g. shorter operation time, simplified tasks) reduces the risks for the patient.

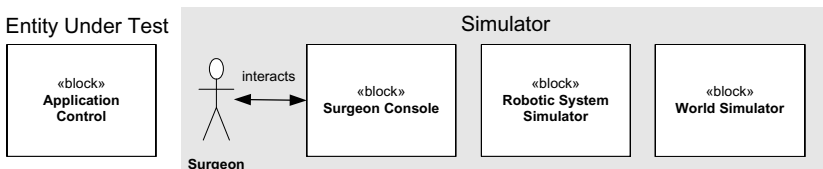


Fig. 10. Simulator Setup for the Class Workflow Validation

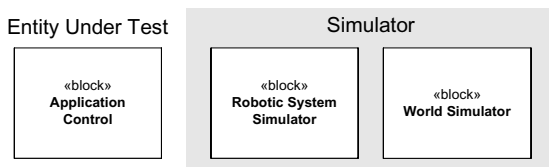


Fig. 11. Simulator Setup for the Class Workflow Design

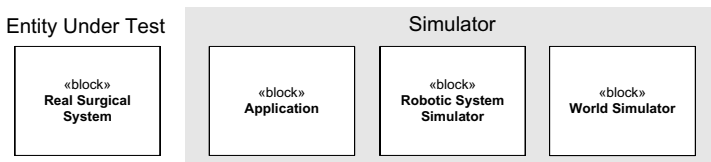


Fig. 12. Simulator Setup for the Class Monitoring

*Use Case Class Monitoring* (Category: Verification). During surgery, the simulator runs in parallel to the real system. A monitor compares the simulation state with the real state. It estimates whether the real system works as intended. A variant of monitoring is predictive simulation, where the future state of the real system is estimated from the current simulation state [9]. **Goal:** Continuously verify real system during operation. **Typical Setup:** The simulator comprises of Application Control, Robotic System Simulator, and World Simulator (see Fig. 12). The real system is the entity-under-test. **Patient safety:** Monitoring increases patient safety by reducing the risk to deviate from a planned procedure. The comparison of simulated and real state enables to detect system failures, deviations from the planned workflow, and planning errors.

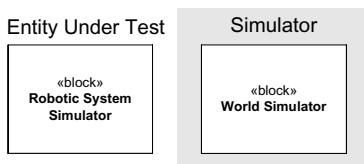


Fig. 13. Simulator Setup for the Class Robot Design

*Use Case Class Robot Design* (Category: Design, Validation, Verification) **Goal:** Improve the design of the robotic system, e.g. optimize the kinematics for a certain workspace. **Typical Setup:** Requires only the World Simulator (see Fig. 13). The Robotics System Simulator is the entity-under-test. **Patient safety:** A better robotic system that is more suitable for the intended procedure reduces the overall risks for the patient.

## 5 Simulator Configurations for Use Cases

This Section exemplifies how to setup the simulator regarding its modeling aspects and details for the two principal use case classes Surgeon Skill Training and Robot Design.

### 5.1 Surgeon Skill Training

The *Surgeon Skill Training* class can be divided into several training goals: (1) **Technical:** training the use of the technical system, i.e. the surgical robotics system, (2) **Procedure:** training a certain surgical procedure and (3) **Patient-specific:** training a patient-specific surgical intervention.

For the **Technical** goal, the mere *basic task* may be sufficient in terms of the *Application*. In case the methods to step through the workspace are part of the training, also the *procedure* needs to be modeled. As concerns the *System*, a simple *kinematics* modeling can usually show most of the relevant system behavior. However, if e.g. the robotic system offers force feedback, also some *dynamics* should be modeled to allow the user to understand this functionality. In the dimension of *Patient*, a *rigid-body* environment is often used; see first training steps in [11]. The force feedback functionality however could be better introduced by using *soft tissue* modeling. If the robotic system uses special instruments (e.g. cautery devices), also a *functional* modeling may be required.

The second goal, **Procedure**, usually involves an *anatomy-true* modeling in *Application*. To allow for interaction between surgical tools and patient, *dynamics* need to be modeled in *System*, and the *Patient* should be modeled either as *soft tissue* or even *functional*.

For the third goal, **Patient-specific**, the *Application* needs to be modeled *case specific*, otherwise it is similar to the second goal Procedure.

### 5.2 Robot Design

Simulation methods are widely used to optimize a *Robot Design* also in non-medical areas. Usual goals in this class are to find (1) **Kinematics:** the kinematics structure of the robot (e.g. number and sequence of joints, link lengths), (2) **Dynamics:** the dynamics parameters (e.g. masses of the segments, stiffnesses, maximum accelerations, applicable forces and torques), or (3) **Implementation:** the appropriate implementation (e.g. sensor resolution, motor characteristics).

In terms of *Application*, the task needs to be *anatomy-true* in most cases to allow for good comparability with reality. The classification w.r.t. the *System* is straight-forward regarding the goal names. As concerns the *Patient*, a *rigid body* or *soft tissue* modeling can be chosen.

## 6 Conclusions

The hierarchical levels of modeling details of the three aspects Application, System, and Patient enable the setup of a simulator for versatile goals in robotic

surgery. The presented use case classes are only the basis from that the actual use cases are derived further. How this works, is illustrated in the previous section. Moreover, metrics of the impact on patient safety should be developed for each use case. This is ongoing work within the project SAFROS (Patient Safety in Robotic Surgery). Moreover, the presented classification is currently used for the implementation of a Robotic System Simulator within the SAFROS project.

The developed concepts are also applicable to tele-robotics in general, e.g. the operation of robots in hazardous environments such as space or disaster zones, etc. Herein the aspect of Environment of Fig. 3 moves from Patient to the new domain. Furthermore the described methods enable to combine different simulations (e.g. workflow, monitoring and simulations to design robots), often used separately in various fields of robotics, within one modular simulator.

**Acknowledgements.** The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 248960. <http://www.safros.eu>

## References

1. da Vinci skills simulator, [http://www.intuitivesurgical.com/products/skills\\_simulator/](http://www.intuitivesurgical.com/products/skills_simulator/) (last visited January 30, 2012)
2. Systems Engineering Handbook, Version 3.1. International Council on Systems Engineering (INCOSE) (2007)
3. Agency for Healthcare Research and Quality: Improving patient safety through simulation research (2008), <http://www.ahrq.gov/qual/simulproj.html>
4. Alimisis, D., Vicentini, M., Fiorini, P.: Towards a problem-based training curriculum for surgical robotics: the SAFROS project. In: Bastiaens, T., Ebner, M. (eds.) Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications, Chesapeake, VA, pp. 297–302 (2011)
5. Banks, J. (ed.): Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice. Wiley & Sons, New York (1998)
6. Basdogan, C., Sedef, M., Harders, M., Wesarg, S.: VR-based simulators for training in minimally invasive surgery. *Computer Graphics and Applications* 27(2), 54–66 (2007), doi:10.1109/MCG.2007.51
7. Franklin, G., Powell, J.D., Workman, M.L.: Digital Control of Dynamic Systems, 3rd edn. Addison Wesley (1998)
8. Herron, D., Marohn, M.: A consensus document on robotic surgery. *Surgical Endoscopy* 22, 313–325 (2008), doi:10.1007/s00464-007-9727-5
9. Hirzinger, G., Brunner, B., Dietrich, J., Heindl, J.: Sensor-based space robotics-ROTEX and its telerobotic features. *IEEE Transactions on Robotics and Automation* 9(5) (1993)
10. Klodmann, J., Konietschke, R., Albu-Schäffer, A., Hirzinger, G.: Static calibration of the DLR medical robot MIRO, a flexible lightweight robot with integrated torque sensors. In: 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3708–3715. IEEE, San Francisco (2011), doi:10.1109/IROS.2011.6095097

11. Lerner, M., Ayalew, M., Peine, W., Sundaram, C.: Does training on a virtual reality robotic simulator improve performance on the da vinci surgical system? *Journal of Endourology* 24(3) (2010)
12. Object Management Group, OMG Systems Modeling Language (OMG SysML), Version 1.2. Document formal/2010-06-01 (2006), <http://www.omg.org/spec/SysML/1.2>
13. Satava, R.: Historical review of surgical simulation - a personal perspective. *World Journal of Surgery* 32(2), 141–148 (2008), doi:10.1007/s00268-007-9374-y
14. Sokolowski, J., Banks, C. (eds.): *Principles of Modeling and Simulation: A Multi-disciplinary Approach*. Wiley & Sons, New York (2009)



# Role of Holographic Displays and Stereovision Displays in Patient Safety and Robotic Surgery

Ali Sengül<sup>1</sup>, Attila Barsi<sup>2</sup>, David Ribeiro<sup>1</sup>, and Hannes Bleuler<sup>1</sup>

<sup>1</sup> Robotic Systems Laboratory, EPFL  
Ecole Polytechnique Fédérale de Lausanne, Switzerland

<sup>2</sup> Holografika Ltd. Budapest, Hungary  
ali.sengul@epfl.ch

**Abstract.** In this study, the role of different 3D vision systems on the patient safety in the context of robotic surgery was studied. Clearly safety is the foremost importance in all surgical procedures. It is well studied in the clinical surgical procedures but the role of different 3D vision system in the context of patient safety is hardly ever mentioned. The assessment of the quality of the 3D images and role of force feedback was studied with two distinct methods (spatial estimation and depth perception) in two different vision systems (holographic and stereovision). The main idea in this study is to investigate quantitatively the role of the vision system in patient safety.

**Keywords:** Safety in surgical robotics, 3D Display, Depth Perception, Stereovision, Holographic Displays, Force Feedback.

## 1 Introduction

Current surgical robots are teleoperator devices and in teleoperation systems, operator interface enables the control of the remote surgical tools hence the operator interface must give the surgeon a complete and realistic perception of the surgical site. The operator interface must provide essential visual and haptic information during the surgical procedure.

To increase patient safety during the procedure, 3D visualization can be used to increase the realism of the visual perception. For the construction of 3D images a well-known technique stereoscopy is usually used. The main idea is to create depth illusion to provide the eyes of the viewer with two different images, representing two perspectives of the same object, with a minor deviation similar to the perspectives that both eyes naturally receive in binocular vision. Stereo Vision was introduced in minimal invasive surgery in early 1990s [1]. Stereoscopic vision (SV) is used in surgical robots, as the da Vinci robot (3D for the surgeon, 2D for the assistants) [2]. This system provides the surgeon full SV by using an endoscopic camera with two lenses. The surgeon controls the remote surgical tools under pure 3D vision without any force feedback. The da Vinci robot is capable of generating large forces and has a big danger of not providing force feedback. Patient safety can be improved with the operator interface integrated with force feedback and high quality 3D images.

The advantage for the use of SV is gain of realism. Numerous research point its advantage in term of speed performance [3, 4] and precision of manipulation [5] but some other research claim that there is no significant benefit [4, 7], they point however that it could be due to the fact that the technology is not advanced enough yet. Some of them show that SV benefits for novice but not for experienced surgeons [8,9]. However, stereovision has also some limitations that should be carefully studied. If we compare a complex real scene viewed binocularly and a computer display of the same scene with all depth cues carefully constructed, the geometric patterns of stimulation striking the two eyes are the same in the two cases. Nevertheless, psychophysical research and experience with virtual reality displays [10] show that the depth in the computer display will appear flattened relative to the real scene from which it is derived. A plausible cause for depth flattening is the fact that computer displays present images on one surface. This means that depth information from focus cues such as; accommodation and the retinal blur gradient are inconsistent with the depicted scene.

Patient safety in a teleoperation system depends on the quality of the presented information in the operator interface. In other words, faithful reproduction of the surgical area in the operator interface is crucial for the patient safety. High quality and realistic rendering of the images and force feedback would increase the patient safety. Studying the interplay of 3D vision and haptics could give some new insights about the future operator interfaces. Therefore, an objective assessment of the quality of the 3D perception with operator interfaces integrated with 3D vision and with haptics is crucial. The operator interfaces can be equipped with haptic devices and advanced 3D visualization displays to increase the perception the surgical area hence the patient safety. New operator interfaces that include holographic displays and force feedback could increase the quality of the representation of the operation room. 3D holographic images can be seen without glasses, without the need for head positioning or eyetracking, since the screen gives a natural "window-like" 3D vision for multiple viewers.

This research is a part of a larger European project Patient Safety in Robotic Surgery (SAFROS) that is about to increase the safety in surgical robotic. For surgical robotic systems, it is crucial to perceive surgical area accurately and precisely. The visual display must provide an exact replica of surgical area. With the current technology to have an exact replica of the surgical site is not reachable but by knowing the limitation of different displays would help to prevent inappropriate uses of these displays hence it would increase the patient safety [11].

In this study we want to investigate quantitatively (error rates and task completion times) under new operator interfaces with integrated 3D vision and haptics. The assessment of the quality of the 3D will be measured objectively and subjectively. The objective measure is based on distance estimation measures (accuracy and precision) and task completion time; the subjective measures are based on the questionnaire ratings. Furthermore, the assessment will be done with two different vision systems (holographic and stereovision).

Patient safety would be increased by using new surgeon console that includes force feedback and better 3D display with more realistic 3D representation. We assume that

using haptic feedback and holographic display would result in a better task performance and presence hence it would increase the patient safety.

This paper is organized as follows. In Section 1 we describe experimental protocols and details of the used hardware and software. In section 2 we present obtained result and in section 3 we discuss our findings concerning depth estimation and spatial assessment.

## 2 Methods

### 2.1 Participants

Nine male participants with an average age of 32.5 (ranging from 23 to 59) for the experiment 1 and nine male participants with an average age of 30.6 (ranging from 25 to 42) for the experiment 2 were recruited. They were right-handed participants and had no disorder of vision or touch and had no history of neurological or psychiatric conditions. Each experiment took around 40 minutes per participant. Participants were informed about the general purpose of the research, gave their informed consent and were compensated for participating to the experiment.

### 2.2 Experimental Protocol and Set Up

In this study two different 3D displays (holographic and stereovision) were used. For the stereovision display we will use the display of the MIMIC simulator that is the simulator of the da Vinci Robot [12], see figure 1. For the holographic 3D display, we will use the 3D displays of Holografika, whose HoloVizio technology generates a whole "3D light-field", not only limited number of views, with 3D images that can be seen with unassisted naked eye by multiple viewers simultaneously, see figure 2. With this technology, the 3D display of the surgical area will be visible by all personnel of the operating room, and not only by one single surgeon, as it is customary with the "da Vinci" system [13].

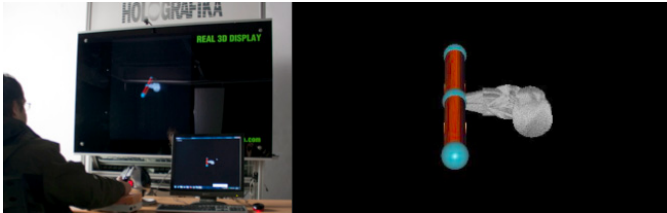
A haptic device (Novint Falcon) was used to track the hand movement and to provide force feedback. CHAI 3D open source platform and set of C++ libraries was used for modeling and simulating the haptics, and for visualization of the virtual world.



**Fig. 1.** 3D Stereoscopic display of Mimic Simulator with a haptic device

### 2.2.1 Spatial Estimation Experiment

We conducted a study to evaluate virtual spatial estimation with two different 3D displays. It is assumed that higher level of immersion would increase sense of presence and more realistic display hence it would result a higher patient safety [14]. Immersion is related with the task performance in VR and presence is related with the user's subjective rating.



**Fig. 2.** Holographic 3D display showing spatial estimation experiment VR

The mismatch between accommodation and convergence in stereoscopic displays were shown to cause visual fatigue and discomfort [15]. This would decrease patient safety and using other displays that does not cause visual fatigue and discomfort would be better for patient safety.

For the spatial assessment, we used a technique that is equivalent to the line bisection task. This technique was chosen since it is a common technique to evaluate for spatial processing and it was validated in real world [15]. In this task participants were required to indicate the center of an object (presented in horizontal or vertical) by bisecting the midline [15].

VRs are mainly related with visual channel. Addition of other channels aimed at improving the realism and increased immersion and sense of presence. A significant amount of literature was published to address the visual realism however the contribution of visual and haptic channel in immersion feeling was not well quantified and studied [11]. Therefore in this study we evaluated the immersion and presence of two different displays by using line bisection task and the contribution of haptic feedback.

A virtual environment including a surgical robot and a cylinder was modeled. To indicate the end and the middle of the cylinder, blue spheres were used. A cylindrical tube was shown either horizontally or vertically. Participants were asked to judge the midline of the cylinder moving a virtual model of a robotic tool. They started the movement at the center of the tube and were required the reach rear and front end of the tube and then to judge the center of the tube by pressing a button on the haptic device. During this task their task completion time and their distance to the center was recorded. In order to evaluate the effect of force feedback, for some cases a virtual wall was modeled at the rear and front end of the tube.

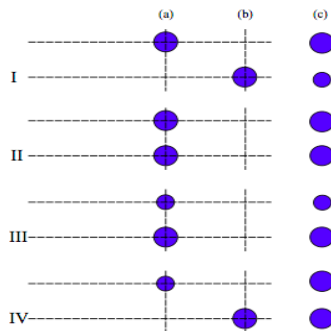
The experiment was designed in a 2x2x2 factorial manner. The 3 factors were “direction” of cylindrical tube (horizontal vs. vertical), “force” (with force vs without force), and the type of “display” (stereovision vs holographic). There was a block of practice trial, which was not analyzed. Practice block was followed by 16 experimental blocks of 3 trials each.

**2.2.2 Depth Perception Experiment**

Depth perception is one of the key issues in 3D displays. In this experiment we evaluated depth perception by using a very simple task. Subjects had to compare the relative depths and relative size of two virtual objects (spheres). Different visual cues are processed by human vision system and combined in human brain to create a 3D perceptual world [16]. Therefore, depth perception is considered as an invisible cognitive state and inaccessible state. To overcome these limitations, researches used allocentric and egocentric distance comparison for the depth perception assessment [11,17,18,19]. In this task observer compares the relative distance between object and a reference point. With this method visual cues involved in depth perception can be accessed directly.

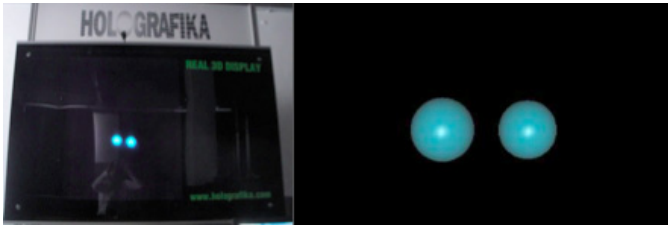
In this study we used the method that was used for depth perception assessment in real word by the works described [11,20,21]. In this experiment, subject answered force choice questions about the size and distance two objects. There were four comparison questions that were composed of two factors (size and/or position), figure 3. These questions are:

1. Both objects are the same size, and at the same depth (apparent sizes are equal),
2. Both objects are the same size, but at different depths (apparent sizes are different),
3. Both objects are different sizes and at the same depth (apparent sizes are different),
4. Both objects are different sizes, but at different depths (such that apparent sizes are equal),



**Fig. 3.** Four cases of two object (a) and (b) real size and location, (c) retinal size [11]

One might think that using only question 1 and 2 would be sufficient for depth perception evaluation. In this case participants could rely on apparent size rather than the depth that could make it difficult to interpret the result. In order to be sure about the result that is based on perceived and not on the apparent size, two other comparison questions 3 and 4 were added. Furthermore we used spheres as virtual objects to force subjects to rely on stereopsis and convergence, see figure 4 [11].



**Fig. 4.** Holographic display showing VR of depth perception experiment

The experiment was designed in a 2x2x2 factorial manner. The 3 factors were “size” of the sphere (same vs. different), “depth” (same vs different), and the type of “display” (stereovision vs holographic). A practice trial was performed before the experiment and it was not analyzed. Practice block was followed by 16 experimental blocks of 8 trials each.

### 2.2.3 Questionnaire

Participants filled out a questionnaire to rate their subjective experiences at the end of the experiment. The questionnaire was adapted from the [22]. The questionnaire was in written form divide into different questions about the presence/immersion, external awareness, quality, enjoyment and ergonomics. Participants were instructed to assign a value ranging from 1 to 7 (1 = strongly disagree to 7 = strongly agree) to 10 statements (see table 1).

### 2.2.4 Data Analysis

Statistical analysis was made based on the mean of the task completion times and position errors in the spatial estimation experiment and based on the mean of the reaction times and error rates for the depth perception experiment. Data from the all trials were analyzed by repeated-measures ANOVAs on the mean values of task completion times and position errors for spatial estimation experiment and mean values of reaction times and error rates for the depth perception experiment. Individual ANOVA variables were compared with post-hoc tests. Paired t-test was used for the comparison of the questionnaire ratings. All numerical values were presented the mean and between-subjects standard error. A significant effect was reported for  $p \leq 0.05$ .

**Table 1.** Questionnaire about the virtual reality experience

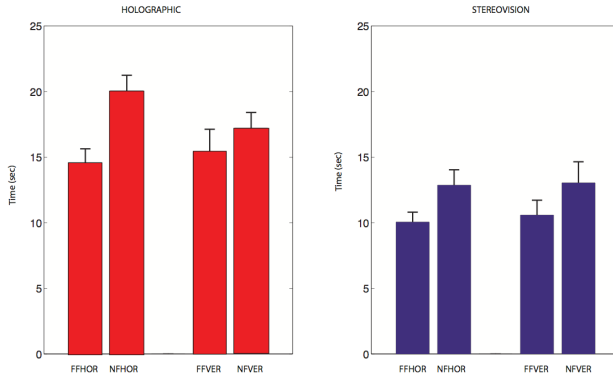
Dimension	Questions
Presence/ Immersion	- The depth impression was realistic - I had a 3D impression of the displayed environment and objects.
External awareness	-I was not aware of the real world—the laboratory
Quality	-I could estimate the distance of the objects well. -I had the feeling that I could reach into the virtual world and touch the objects. -I could see the virtual world clearly
Enjoyment	-The quality of the graphical presentation was fascinating.
Ergonomics	-After the experiment I had muscle pain in the neck. -The display was comfortable and not tiring for the eyes -After the experiment I had muscle pain in the arm.

### 3 Results

#### 3.1 Spatial Estimation Experiment

To compare the spatial assessment performance in both holographic and stereovision display, distance to the center point and task completion time means with standard deviations were shown in fig 5 and fig 6.

ANOVA tests on task completion time rates revealed a main effect of display ( $F_{(1, 8)} = 13.9, p = 0.005$ ). Other significant factors are Force ( $F_{(1, 8)} = 59.7, p < 0.001$ ) confirming that task completion time was significantly less when force feedback was applied and a significant interaction between Direction and Force. ( $F_{(1, 8)} = 16.7, p = 0.003$ ). Post-hoc comparison reveals a significant difference between conditions 1 No Force Horizontal (NFHor) and 2 Force Horizontal (FFHor) ( $p = 0.014$ ), 1 and 4 Force Vertical (FFVer) ( $p = 0.03$ ), 2 No Force Vertical (NFVer) and 3 ( $p = 0.01$ ), 3 and 4 ( $p = 0.026$ ) 3 and 4 ( $p = 0.02$ ) for stereoscopic display and significant difference between conditions 1 (NFHor) and 2 (FFHor) ( $p < 0.001$ ), 1 and 3 (NFVer) ( $p = 0.002$ ), 2 (NFVer) and 3 ( $p = 0.003$ ), 3 and 4 ( $p = 0.02$ ) for holographic display.

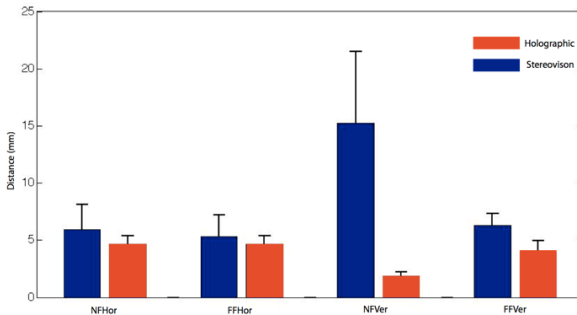


**Fig. 5.** Task completion time and role of force feedback

ANOVA tests on distance to the center revealed an interaction between Display and Dir ( $F_{(1, 8)} = 5.9, p = 0.04$ ). Post-hoc comparison did not reveal any significant difference between conditions for stereoscopic display but it revealed significant difference between conditions 1 (NFHor) and 3 (NFVer) ( $p = 0.005$ ), 2 (FFHor) and 3 (NFVer) ( $p = 0.005$ ), 3 and 4 (FFVer) ( $p = 0.02$ ) for holographic display.

### 3.2 Depth Perception Experiment

To compare the depth perception performance in both holographic and stereovision display, error rate and response time means with standard deviations were shown in fig 7 and fig 8. For holographic display subjects showed better accuracy but slower response times.

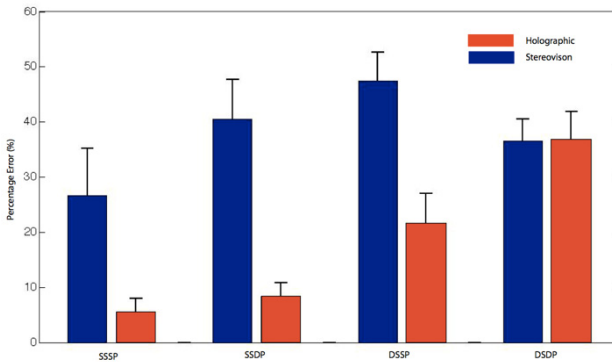


**Fig. 6.** Distance to center of cylindrical tube

ANOVA tests on error rates revealed a main effect of display ( $F_{(1, 8)} = 28.9, p < 0.001$ ), conforming that holographic display has lower error rate compared to stereoscopic display. Other significant factors are Size ( $F_{(1, 8)} = 11.7, p = 0.009$ ) and a significant interaction between Display, Size and Depth. ( $F_{(1, 8)} = 6.57, p = 0.03$ ). Post-hoc comparison reveals that for the SV system, different conditions reveals no



significant difference between four comparisons but it reveals a significant difference between conditions 1 Same Size Same Position (SSSP) and 3 Different Size Same Position (DSSP) ( $p = 0.016$ ), 1 and 4 Different Size Different Position (DSDP) ( $p < 0.001$ ), 2 Same Size Different Position (SSDP) and 3 ( $p = 0.037$ ), 2 and 4 ( $p < 0.001$ ) 3 and 4 ( $p = 0.02$ ) for the holographic display.



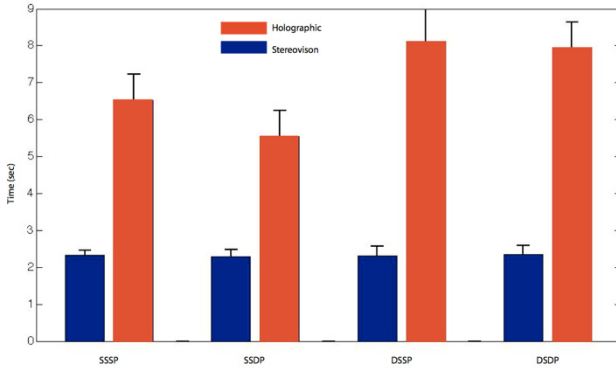
**Fig. 7.** Percentage error: Holographic display has lower % error

ANOVA tests on reaction time revealed a main effect of display  $F_{(1,8)} = 40.5$ ,  $p < 0.001$ , conforming that holographic display has larger response time compared stereoscopic display. Other significant factors are size ( $F_{(1,8)} = 22.2$ ,  $p = 0.002$ ) and a significant interaction between Display and Size. ( $F_{(1,8)} = 12.83$ ,  $p = 0.007$ ). Post-hoc comparison reveals that for the SV system, different conditions reveals no significant difference between four comparisons but it reveals a significant difference between conditions 2 and 3 ( $p = 0.007$ ), 2 and 4 ( $p = 0.01$ ) for the holographic display.

### 3.3 Questionnaire

Concerning questionnaire scores, we found that ratings in question Q4 about the effect of force feedback (“I had the feeling that I could reach into the virtual world and touch the objects.”) was significantly larger in the force feedback condition (Q4:  $p < 0.0413$  for SV and  $p = 0.0402$  for holographic vision). No significant difference between force feedback and no feedback condition for the other questions.

Then we performed 2-tailed paired t-test for the average question rating for stereoscopic display and holographic display. In this case we found Q8 (After the experiment I had muscle pain in the neck) significantly higher ratings for the stereoscopic display ( $p = 0.003$ ) and Q9 (The display was comfortable and not tiring for the eyes) significantly higher rating for holographic display ( $p < 0.001$ ).



**Fig. 8.** Reaction time: Holographic display has higher reaction time

## 4 Discussion

VEs can be viewed using different 3D vision system that can influence the perception of the environment. Role of different 3D virtual reality platform and vision system on the patient safety in the context of robotic surgery was studied. The assessment of the quality of the 3D images and role of force feedback was studied with two distinct methods in two different vision systems (holographic and stereovision). The idea behind this approach was to investigate quantitatively the role of the vision system in patient safety.

We found some difference in task completion time in the spatial estimation experiment. The result of the spatial estimation experiment showed the importance of force feedback. As it is seen in figure 5, participants were able to complete the task faster in the case of force feedback. Providing force feedback at the end front and rear end of the tube enabled them complete the task faster. This could be explained that participants used the visual cues and haptic cue to access the boundaries of the tube. The integration of visual and haptic cues could have increased their performance. Furthermore, movements were less precise for the stereovision display and midline assessment accuracy was lower for the stereovision display. As it is seen in figure 6, the distance to the midline of the tube was lower for the holographic display.

One of the main goals of this study was to determine whether participants' performance in depth perception task changes when different 3D display was used. The ANOVA result on the percentage error of the depth perception experiment revealed a difference. This effect was achieved for most the conditions for the holographic display and this was not the case for the stereovision display. In other words, participants showed a better performance in holographic display compared to the stereovision display. As it is seen in figure 7, participants had significantly lower error rate for the holographic display, except in the ambiguous comparison condition (condition 4: DSDP). Furthermore, ANOVA result on the response time of the depth perception experiment also revealed a difference. The response time for the holographic display was significantly higher then the stereovision display for all the

conditions, see fig 8. One might interpret the competing result on accuracy and response time as speed-accuracy problem. However this should not be the case for this experiment because participants were asked to respond when they are sure about the answer. Hence the difference in the error rates could not be due to the speed accuracy tradeoff. In the holographic display participants took more time by looking at different angles, therefore their response times are longer and they have higher accuracy rate.

Subjective ratings on presence/immersion, external awareness, quality and enjoyment were satisfying for both of the 3D displays. However, subjective rating on the ergonomics was not satisfying for the stereovision display. Participants report muscle pain in the neck and discomfort due to the tiring for the eyes for the stereovision display. On the other hand, subjective rating on the ergonomics for the holographic display was satisfying.

## 5 Conclusion

This study focused on the assessment of surgeon's accuracy and action/perception limit under different surgical operator interface with different 3D vision systems and how different vision system and haptic could increase patient safety. A statistical comparison was done on the quantitative measures of the two distinct displays. A comparison of 3D SV system of the da Vinci simulator and holographic visual system was performed. As expected our result showed that using haptic feedback and holographic display resulted in better task performance and presence hence it would increase the patient safety.

**Acknowledgements.** This research project is supported by European Project FP-7 SAFROS [www.safros.eu](http://www.safros.eu).

## References

1. Cuschieri, A.: Minimal access surgery and the future of interventional laparoscopy. *Am. J. Surg.* 161, 404–407 (1991)
2. Da Vinci Surgery, Intuitive Surgical (2012), <http://www.intuitivesurgical.com/index.aspx>
3. Vasilyev, V., Novotny, M., Martinez, F., Loyola, H., Salgo, I., Howe, R., Nido, J.: Stereoscopic vision display technology in real-time three-dimensional echocardiography-guided intracardiac beating-heart surgery. *J. Thorac. Cardiovasc. Surg.* 135, 1334–1341 (2008)
4. Taffinder, N., Smith, S., Huber, J., Russell, R., Darzi, A.: The effect of a second-generation 3D endoscope on the laparoscopic precision of novices and experienced surgeons. *Surg. Endosc.* 13, 1087–1092 (1999)
5. Mueller-Richter, U., Limberger, A., Weber, P., Ruprecht, K., Spitzer, W., Schilling, M.: Possibilities and limitations of current stereo-endoscopy. *Surg. Endosc.* 18, 942–947 (2004)

6. Hofmeister, J., Frank, T., Cuschieri, A., Wade, N.: Perceptual aspects of two-dimensional and stereoscopic display techniques in endoscopic surgery: review and current problems. *Semin. Laparosc. Surg.* 8, 12–24 (2001)
7. Peitgen, K., Walz, M., Holtmann, G., Eigler, F.: A prospective randomized experimental evaluation of three-dimensional imaging in laparoscopy. *Gastrointest. Endosc.* 44, 262–267 (1996)
8. Bergen, P., Kunert, W., Bessell, J., Buess, G.: Comparative study of two-dimensional and three-dimensional vision systems for minimally invasive surgery. *Surg. Endosc.* 12, 948–954 (1998)
9. Suematsu, Y., Martinez, J., Wolf, B., Marx, G., Stoll, J., DuPont, P.: Three-dimensional echo-guided beating heart surgery without cardiopulmonary bypass: atrial septal defect closure in a swine model. *J. Thorac. Cardiovasc. Surg.* 130, 1348–1357 (2005)
10. Watt, S., Akeley, K., Ernst, M., Banks, S.: Focus cues affect perceived depth. *Journal of Vision*, 834–862 (2005)
11. Naceri, A., Chellali, R., Dionne, F., Toma, S.: Depth Perception Within Virtual Environments: Comparison Between two Display Technologies. *International Journ. on Advances in Intelligent Systems* 3(1&2) (2010)
12. Dv Trainer. Mimics (2012), <http://www.mimic.ws/>
13. Holografika, Holografika (2012), <http://www.holografika.com/>
14. Beck, L., Wolter, M., Mungard, N., Vohn, R., Staedtgen, M., Kuhlen, T., Sturm, W.: Evaluation of Spatial Processing in Virtual Reality Using Functional Magnetic Resonance Imaging (fMRI). *Cyberpsychology Behavior and Social Networking* 13 (2010)
15. Hoffman, D., Girshick, A., Akeley, K.: Vergence-accommodation conflicts hinder visual performance and cause visual fatigue. *Journal of Vision* 8, 1–30 (2008)
16. Swan, J., Jones, A., Kolstad, E., Livingston, M., Smallman, H.: Egocentric depth judgments in optical, see through augmented reality. *IEEE Trans. Vis. Comput. Graph.* 13(3), 429–442 (2007)
17. Loomis, J.M., Da Silva, J.A., Philbeck, J.W., Fukusima, S.S.: Visual perception of location and distance. *Current Directions in Psychological Science* 5, 72–77 (1996)
18. Loomis, J.M., Knapp, J.M.: Visual perception of egocentric distance in real and virtual environments. In: Hettinger, L.J., Haas, M.W. (eds.) *Virtual and Adaptive Environments*, pp. 21–46 (2003)
19. Aznar-Casanova, J.A., Matsushima, E.H., Silva, J.A.D., Ribeiro-Filho, N.P.: Can exocentric direction be dissociated from its exocentric distance in virtual environments? *Percept. Psychophys.* 70(3), 541–550 (2008)
20. Gruber, H.E.: The relation of perceived size to perceived distance. *The American Journal of Psychology* 67, 411–426 (1954)
21. Berryhill, M.E., Fendrich, R., Olson, I.R.: Impaired distance perception and size constancy following bilateral occipitoparietal damage. *Exp. Brain Res.* 194(3), 381–393 (2009)
22. Armbruster, C., Wolter, M., Kuhlen, T., Spijkers, W., Fimm, B.: Depth Perception in Virtual Reality: Distance Estimantions in Peri- and Extrapersonal Space. *Cyber Psychology & Behavior* 11 (2008)

# A Methodological Framework for the Definition of Patient Safety Measures in Robotic Surgery: The Experience of SAFROS Project

Angelica Morandi<sup>1</sup>, Monica Verga<sup>1</sup>, Elettra Oleari<sup>1</sup>,  
Lorenza Gasperotti<sup>2</sup>, and Paolo Fiorini<sup>2</sup>

<sup>1</sup> e-Services for Life & Health, Fondazione Centro San Raffaele del Monte Tabor Milan, Italy  
{morandi.angelica,oleari.elettra,verga.monica}@hsr.it

<sup>2</sup> Altair Robotics Laboratory, Department of Computer Science, University of Verona  
lorenza.gasperotti@metropolis.sci.univr.it, paolo.fiorini@univr.it

**Abstract.** This paper describes an innovative methodological approach developed within the SAFROS European project (Patient Safety in Robotic Surgery, FP7) that allows to identify a set of metrics for the evaluation of patient safety during the pre- and intra-operative phase of robotic interventions. This methodology enlarges the focus of the evaluation: safety is assessed not only through the analysis of features and limitations of the technological solutions involved in the execution of target surgical interventions (product safety) but also in terms of the effects of their interaction within the surgical scenario (process safety). Moreover potentialities and obstacles emerging from the introduction of such technologies into the operating room and the hospital as systems (organizational safety) are evaluated. The proposed method has been applied to the analysis of one gold standard and two benchmark procedures. The output of the analysis is a complex set of key metrics addressing safety at the above described levels. This paper reports the most relevant and interesting subset of the obtained results. The extendibility of the method to the analysis of other interventions will be verified with suitable experiments in the next future.

**Keywords:** patient safety, robotic surgery, safety metrics, surgical workflow, SAFROS project.

## 1 Introduction

Patient safety is defined by the World Health Organization (WHO) as the absence of preventable harm to a patient during the process of health care. The discipline of patient safety is the coordinated effort to prevent harm to patients, caused by the process of health care itself [1]. During the last 10 years the relevance attributed to the prevention and management of errors raised considerably within both the American and European medical community. The publication of [2] [3] by the Institute Of Medicine (IOM) and of [4] by the Quality Interagency Coordination Task Force (QuIC) demonstrated the willingness to face the considerable number of medical errors that negatively affect patient outcomes and the performance of the medical staff in general. The introduction of robotic surgery marked a turning point for the surgical practice in the context of

patient safety [5]. It enhanced the benefits introduced by minimally invasive technologies (reduction of intra- and post-operative blood loss, shortening of patients recovery time etc) and solved some of their limitations providing the surgeon with high quality stereo vision and 7 degrees of freedom tools, thus enabling them to perform extremely precise and dexterous movements [6]. Despite the ability of surgeons in using surgical robots and the precision of such machinery, no study has looked at all aspects of safety in robotic surgery. As of today, the analysis of the effects due to the introduction of robotic technologies has been conducted by evaluating the requirements for a proper preoperative diagnostic, for the performance of the intervention and for the training of surgeons separately (e.g. in [7]). In addition, the evaluation of the performance quality of these phases does not involve the use of metrics that specifically address patient safety. The SAFROS project (Patient Safety in Robotic Surgery, EU 7th Framework Programme) aims at developing technologies and methods for the enhancements of patient safety in surgical procedures extending the analysis to the whole surgical workflow. The SAFROS project addresses the complete planning-execution loop in surgical robotics, to ensure a seamless data flow and consistent accuracy in all phases of robotic-assisted surgery. The project aims at demonstrating that the introduction of assistive technologies improves patient safety during the performance of surgical intervention with respect to their current way of execution. To this extent the identification of a set of metrics for the objective assessment of patient safety is required.

## 2 Objective

The aim of this paper is to describe the methodological approach introduced in the context of the SAFROS project to define the metrics that will enable the assessment of patient safety in robotic surgery. The analysis has been extended to the whole surgical workflow, starting from the pre-operative phase to the execution of the operation. Moreover it took into account all the aspects concurring to the complexity of the system and to patient safety, including technological, procedural, environmental and patient related factors. This methodology has been developed to identify a list of metrics related to both technical and medical aspects as indicators for a safe surgical performance. The described method has been applied to the analysis of a standardized robot-assisted procedure such as the Robot-Assisted Radical Prostatectomy (RALP) [10] and of two case studies, namely pancreatic tumor enucleation and the Abdominal Aortic Aneurysm (AAA) repair [8] [9]. Positive results obtained from the application of the method in the selected case will be the base for its wider usage.

The first part of the paper reports the methodological framework as it has been developed within the SAFROS project. The description follows the subdivision of the method into the three levels of the analysis: namely product, process and organizational safety. Then the paper reports the results obtained for the specified surgical procedures. Finally, conclusions are drawn and possible extensions proposed.

## 3 Methods

The methodological approach applied within the SAFROS project includes three increasingly complex levels of analysis: product safety analysis, process safety analysis,

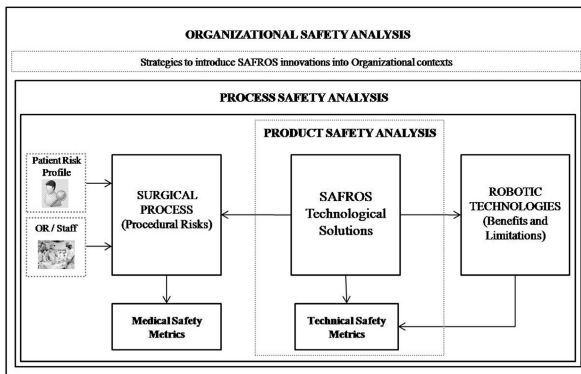


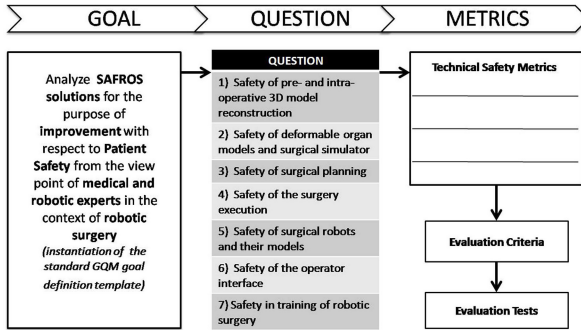
Fig. 1. Schema of the whole methodological approach

and organizational safety analysis. This allows to assess the safety of the applied technologies considering them first singularly, then analyzing their interaction into a surgical process and finally integrating them into a wider organizational context (Fig. 1). Such a schema makes possible to reengineer the surgical process through the introduction of technological innovations and to extract patient safety indicators to assess the impact of these solutions.

Literature researches [7] [11]–[15] and interviews to expert surgical teams and expert robotic surgeons allowed the identification of the most relevant limits affecting commercial surgical robots specifically related to their use in the considered surgical procedures [13] [14]. It has been pointed out that four are the main categories in which robotic features could be grouped (mechanics, imaging, surgical workflow and operating room (OR) environment and patient), each of them bringing benefits and limitations when introduced into a surgical environment. As example, referring to mechanics, the possibility to have accurate tools with scaled movements and extra degrees of freedom represents a great advantage but on the other side robots are cumbersome, with time consuming to set-up.

### 3.1 Product Safety Analysis

The product safety level aims at evaluating the features of SAFROS technological solutions, some of which are already existing whilst other are being developed within the project. SAFROS products have a potential to increase patient safety, possibly overcoming the current limitations of robotic surgery. The analysis focuses on virtual and synthetic organ models, virtual simulator for training and planning, pre-operative planning technologies, operating room monitoring system and robotic simulator. The specifications of SAFROS technological products were outlined with a set of technical safety metrics to understand their impact on patient safety during surgical interventions and to assess their potentiality. The product safety analysis has been supported by the Goal-Question-Metric (GQM) model [16] [17], a goal-driven measurement system for software development, where each goal triggers the identification of appropriate metrics.



**Fig. 2.** The GQM approach applied in SAFROS

We adapted this approach in the context of SAFROS project (see Fig. 2) defining its specific Goal: “analyze SAFROS products for the purpose to improve the Patient Safety from the view point of medical and robotic experts in the context of robotic surgery”.

In order to satisfy the SAFROS objectives, we outlined SAFROS main Questions and the corresponding technical safety metrics. For each solution we pointed out what evaluation criteria have to be investigated and which laboratory evaluation tests we have to perform. As a result, a list of 55 Technical Safety Metrics and related evaluation methods has been obtained. An excerpt of them is provided in the Table 1.

### 3.2 Process Safety Analysis

The process safety level expands the focus of previous analysis through the application of a systems approach, that evaluates the effects of the interaction of the above mentioned products and their integration into the different phases of surgical procedures. The scope of this research is to identify and select the criticalities in the surgical process in order to design the ways in which SAFROS technologies could reduce them and improve benefits for the patient. At this level the analysis comprises the robotic and the surgical procedures, also including the factors influencing patient safety such as the operating room environment and patient related information. The conclusions of this level of the study conducted to the identification of a list of medical safety metrics related to each element, starting from the study of the risks embedded in the whole surgical scenario. For this purpose we exploited a simplified version of the Failure Mode and Effects Analysis (FMEA) [18] as risk assessment tool to study the gold standard robot-assisted radical prostatectomy (RALP), and the two SAFROS benchmark procedures (vascular and abdominal). Preliminarily we identified the procedural steps of the pre- and intra-operative phases for each procedure. FMEA helped in identifying all the procedure-related risks, considering also the respective causes and effects. In close collaboration with OR personnel (e.g. qualified surgeons, anesthetist, nurses, ...) through site visits and interviews, the risks have been ranked according to a Criticality Index (CI). This parameter is obtained by multiplying the estimated frequency of occurrence



**Table 1.** Example of Technical Safety Metrics resulting from the product safety analysis

Technical Safety Metrics	
Products	Criteria
<i>Virtual and synthetic organ models</i>	
Realism of physical model dynamic behaviour	Force deviation from actual in dynamic experiments (% , mN)
Mechanical properties	Measured Youngs modulus [Pa], deviation from target [Pa], stress-strain curve acceptability: yes/no
<i>Surgical Planner</i>	
Error between real and reconstructed environment	Linear errors (m), yes/no (task dependant)
Reduction in cognitive load/ time when facing adverse events in simulation	Ratio between trained completion time and untrained completion time %, ratio <1 significantly
<i>Pre-operative planning technologies</i>	
Error between real/registered environment	Linear error (mm), if $\leq 1\text{cm}$ = yes
Stability of the estimated registration transforms	Variance of the transform parameters, if $<5\%$ = yes
Quality of obstacle detection and human tracking	mm, <10cm
Time delay between real world motions and update of the virtual scene	<500ms
<i>Robotic system simulator</i>	
Cartesian positioning accuracy	mm, rad
Accurate torque measurement	N, Nm

(O, from 1 to 4) of the risk by the expected severity (S, from 1 to 5) of the damage caused to the patient in the event the risk converts into reality Fig. 3.

As example, in Table 2 is reported some of the highly ranked risks selection for the most representative procedural steps of the robotic surgical workflow, referring to RALP. From the risk analysis it was possible to deduce a set of medical safety metrics addressing the criticalities of the procedures taken into account. This will enable the qualitative and quantitative assessment of the improvement in patient safety provided by the introduction of SAFROS solutions.

The other determinant factors influencing the safety in surgical theatre are related to patient and the OR environment. Criticalities due to patient-specific clinical and anatomical characteristics, e.g. pathology and comorbidities, outlining the individual risk profile have also been collected by reviewing clinical documentations and informed consents forms and by collecting surgeons opinions. We also encompassed the characteristics of the operating scenario in terms of ergonomics and the role of human factors related to the internal dynamics of the surgical team [19]. The results of the Process safety analysis collected in Fig. 4 are the Medical Safety Metrics obtained from the analysis of the different factors influencing patient safety previously discussed (procedure-related risks, patient-specific data and operative environment).

Severity Occurrence	No damage (1)	Minor damage (2)	Medium damage (3)	Serious damage (4)	Really serious damage (5)
Remote (1)	1	2	3	4	5
Occasional (2)	2	4	6	8	10
Probable (3)	3	6	9	12	15
Frequent (4)	4	8	12	16	20

CI from 1 to 2 = Acceptable Risks
  CI from 7 to 12 = Medium Risks  
 CI from 3 to 6 = Low Risks
  CI from 13 to 20 = High Risks

**Fig. 3.** Risk Matrix used to rank the procedural risks

**Table 2.** Analysis of the risks related to the robot-assisted procedure

Process Step	Related Risks	Causes	Effects
Pre-operative planning	Not adequate evaluation of the surgical strategy	Incomplete/inaccurate evaluation of the pre-operative tests	Prolonged surgery Deviation from planned operation Not prompt reaction of the surgeon to complications
Surgical steps execution: First Surgeon	Partial or missed visualization of the robotic instruments as they are positioned in the pelvis	Technical malfunctions in the optical tool/camera Improper check of the tools (e.g. camera out of focus) Surgeon's inattention	Prolonged surgery Damage to anatomical structure Collisions
Surgical steps execution: First Assistant and the rest of the surgical team	Improper changing of the instruments	First assistants skill (e.g. insufficient confidence with robotic devices) Missed communication between OR staff Lack of knowledge of inserting the robotic instruments	Prolonged surgery Damage of the anatomical structures

Procedure-related risks	Medical Safety Metrics
<p><i>Robotic-surgical procedure</i></p> <ul style="list-style-type: none"> <li>- Not adequate evaluation of the surgical strategy (CI=6,93)</li> <li>- Partial or missed visualization of the robotic instruments as they are positioned in the pelvis (CI=2,83)</li> <li>- Improper changing of the instruments (CI=2,83)</li> </ul>	<ul style="list-style-type: none"> <li>- Fidelity of anatomical structures</li> <li>- Fidelity in feature and spatial relationship of the tumour</li> <li>- Accuracy in task execution</li> <li>- Damage to organs/vessels</li> <li>- Rapid identification of critical situations</li> <li>- Time to perform specific tasks, e.g. clamping, anastomosis (min)</li> <li>- Operative time (min)</li> <li>- Technical set-up of the robotic system</li> <li>- Adequacy of data for decision making</li> <li>- Adequacy of patient positioning on surgical table</li> <li>- Correctness of sterility and shaving procedure</li> <li>- Standardization of the port placement</li> <li>- Satisfactory ergonomics of the prime surgeon console</li> <li>- Adequacy of the internal workspace for the surgeon</li> <li>- Tool functioning</li> <li>- Visualization of the working field</li> <li>- Coordination skills</li> <li>- Leadership and management skills</li> </ul>
<p><i>AAA repair related-risks</i></p> <ul style="list-style-type: none"> <li>- Serious lumbar bleeding when opening the aneurysm (CI=10,4)</li> <li>- Bleeding of the suture line (CI=9,4)</li> <li>- Improper positioning the aortic clamp (CI=8,73)</li> <li>- Damages to the surrounding structures during the aorta exposition (CI=7,87)</li> <li>- Needle/suture breaking (CI=6,64)</li> <li>- Not gradual declamping, respecting the pressure values (CI=6,45)</li> </ul>	
<p><i>Pancreatic enucleation related-risks</i></p> <ul style="list-style-type: none"> <li>- Damage to pancreatic duct during enucleation (CI=7,55)</li> <li>- Damages to pancreatic parenchyma and surrounding anatomic structures during pancreas exposure (CI=7)</li> <li>- Damage to the spleen and spleen's vessels during pancreas tail resection (CI=6,5)</li> <li>- Not complete resection of the tumoral mass (CI=4,8)</li> <li>- Unexpected factors not evaluated during the planning (CI=2,76)</li> </ul>	
<p><i>Patient-specific data</i></p> <ul style="list-style-type: none"> <li>- Principal compliant</li> <li>- Comorbidities</li> <li>- Body mass index</li> <li>- Age, gender</li> <li>- Personal anatomical variability</li> <li>- Previous interventions</li> </ul>	
<p><i>Operative environment</i></p> <ul style="list-style-type: none"> <li>- Noise and light conditions</li> <li>- Availability of proper instruments</li> <li>- Distractions/interruptions</li> <li>- Human factors (teamwork, leadership, pressure...)</li> </ul>	

Fig. 4. Risk Matrix used to rank the procedural risks

### 3.3 Organizational Safety Analysis

The organizational safety level addresses the potential impact of the application of new technologies on a higher level environment: the operating room system and the hospital as an institution. It's relevant to evaluate which are the determinants for a successful possible introduction of SAFROS system in this complex organizational environment. In fact, high reliability in healthcare is founded on the context in which care is delivered, called organizational culture, and that has important influences on patient safety [20]. At the day of the publication of this paper the work concerning the Organizational safety analysis is still in progress. It requires considerable efforts and time as it involves different categories of professionals, including lawyers, sociologists, surgeons and members of hospitals management.

## 4 Results

SAFROS project aims at developing technologies and methods in order to enhance patient safety and proving their effectiveness when introduced in the robotic surgical environment. The analysis conducted to the definition of a set of technical and medical safety metrics for the evaluation of SAFROS solutions applied to pre or intra-operative phases of the robotic-assisted surgical procedure. In Fig. 5 is presented an extract of the result of the entire analysis. SAFROS solutions are reported together with their Technical Safety Metrics. The focus is on the benefits that their introduction imply in the context of patient safety. In the last column are finally referred the Medical Safety Metrics that each solution is addressing to and that will help to evaluate how effectively the patient safety is improved by their introduction.

SAFROS solution	Technical safety metric	Benefits for patient safety	Medical safety metric addressed
Pre-operative patient reconstruction as a reference for intra-operative registration	<ul style="list-style-type: none"> <li>- Error between real and reconstructed environment</li> <li>- Stability of the estimated registration transforms.</li> </ul>	Precise localization of organs, forbidden regions, and target structures	<ul style="list-style-type: none"> <li>- Fidelity of anatomical structures</li> <li>- Adequacy of data for decision making</li> <li>- Rapid identification of critical situations</li> </ul>
Robotic system simulator for pre-operative planning	<ul style="list-style-type: none"> <li>- Time delay between real world motions and update of virtual scene;</li> <li>- Update rate</li> <li>- Cartesian positioning accuracy</li> </ul>	Allow a proper planning of the OR to reduce the risks of wrong localizations. Reduce or avoid "Out of range Motion" due to kinematic limitations	<ul style="list-style-type: none"> <li>- OR set up time</li> <li>- Rapid identification of critical situations</li> <li>- Instrument collisions</li> </ul>
Intra-operative registration	<ul style="list-style-type: none"> <li>- Error between real and registered environment</li> <li>- Stability of the estimated registration transforms</li> <li>- Target registration error</li> </ul>	Matching the real-time US with the pre-operative reconstruction allows better visualization and localization of target structures inside the patient	<ul style="list-style-type: none"> <li>- Accuracy in task execution</li> <li>- Avoid structures</li> <li>- Overall operative time</li> <li>- Visualization of the working field</li> </ul>
Intra-operative supervision system in the shared workspace surgeon/robot	<ul style="list-style-type: none"> <li>- Quality of obstacle detection and human tracking</li> <li>- Time delay between real world motions and update of virtual scene</li> <li>- Update rate</li> </ul>	The supervision system provides an additional safety net for the robot reducing problems at the patient side, through an optical monitoring of workspace of the robot and the surgeon + staff.	<ul style="list-style-type: none"> <li>- Rapid identification of critical situations:</li> <li>- Instrument collision</li> <li>- Robot/OR staff collisions</li> <li>- Operative time</li> </ul>

Fig. 5. Extract of the SAFROS patient safety analysis results

## 5 Discussion and Future Perspectives

The application of the proposed method to the considered surgical procedures led to the identification of several metrics for the assessment of patient safety. The complexity of the obtained metrics and their wide focus, embracing not only single components but whole processes and systems, suggests the necessity of a holistic approach for patient safety analysis. Another important outcome of the performed work is the definition of a set of key aspect that should be taken into account for the development of new technological solution for robotic surgery. The application of the proposed protocol to the targeted pancreatic and vascular procedures may be considered as the main limitation of the current work. In fact as for today these interventions are not routinely assisted by robots, since their high complexity of execution has hampered an extensive use of technology. Despite this, their adoption as test-beds for the SAFROS method was grounded on the belief that patient safety, as a systemic notion, could be better addressed in unexplored fields where no consolidated practices exist. The extendibility of the proposed methodological approach to other medical or surgical specialties will be soon analyzed. In fact safety must drive the development of technological solutions in the early design phase and not introduced when already established design constraints can impede fur-

ther improvements. The correctness of the defined metrics for the assessment of patient safety levels will be verified. Tests on benchmark procedures and selected surgical tasks will allow to verify the reliability and consistency of the identified Safety Metrics. Testing activities will be devoted firstly to evaluate if the proposed methodological approach is useful to represent the different aspects of patient safety in robotic surgery. Secondly the experimental phase will validate whether the proposed solutions bring an improvement in patient safety compared to traditional procedures. The experimental protocol will be composed of three different levels of analysis, linked to those safety demonstration purposes characterizing the contents of the SAFROS methodology. To achieve these requirements for patient safety, tests will be grouped into three parts: 1) Technical Test, to assess the contribution in safety due to the introduction of SAFROS innovative technologies; 2) Functional Tests, to assess the safety when SAFROS solutions are introduced into a surgical scenario; 3) Integration tests, to verify the interaction between different SAFROS solutions. The definition of the testing protocols and the set-up of the testing activities requires time and resources and will be finalized in the coming period within SAFROS framework. The organizational safety analysis will be completed in the next future, exploring the field of human factors toward the environment of safety culture. In fact, a substantial improvement in safety could be brought not only through merely technical factors but also with non-technical ones, analyzing those human factors relevant for robotic surgery. This type of competence is important for the future development of a comprehensive training curriculum, where technical skills and human factors knowledge are integrated. The non-technical contents could be extended to executive leaders, in order to improve the effectiveness of the organizational safety. Confirming through the experimental protocol described, the existence of correlations between the Technical Safety Metrics and the Medical Safety Metrics, it will be possible to demonstrate that the proposed method is suitable for the objective and coherent with the underlying assumptions.

**Acknowledgements.** The research leading to these results has received funding from the European Union Seventh Framework Programme FP7/2007-2013 under grant agreement n. 248960 (SAFROS). We wish to acknowledge the Vascular Surgery Operative Unit and the Urology Department of Hospital San Raffaele, Milan (Italy), and the General Surgery Department of University Policlinico Borgo Roma, Verona (Italy), for their cooperation within the SAFROS project.

## References

1. [http://www.who.int/features/factfiles/patient\\_safety/en/index.html](http://www.who.int/features/factfiles/patient_safety/en/index.html)
2. Kohn, L.T., Corrigan, J.M., Donaldson, M.S.: To err is human: building a safer health system, Institute of Medicine Report. National Academy Press, Washington DC (2000)
3. Edington, M. (ed.): Crossing the Quality Chasm: A New Health System for the 21st Century, Institute of Medicine Report. National Academy Press, Washington DC (2001)
4. Quality Interagency Coordination Task Force, Doing what counts for patient safety: Federal actions to reduce medical errors and their impact. Agency for Healthcare Research and Quality, Washington DC (2000)

5. Makary, M.A., Sexton, J.B., Freischlag, J.A., Millman, E.A., Pryor, D., Holzmüller, C., Pronovost, P.J.: Patient safety in surgery. *Annals of Surgery* 243(5), 628–635 (2006), doi:10.1097/01.sla.0000216410.74062.0f
6. Gutt, C.N., Oniu, T., Mehrabi, A., Kashfi, A., Schemmer, P., Büchler, M.W.: Robot-assisted abdominal surgery. *British Journal of Surgery* 91(11), 1390–1397 (2004), doi:10.1002/bjs.4700
7. Herron, D.M., Marohn, M., The SAGES-MIRA Robotic Surgery Consensus Group: A consensus document on robotic surgery. *Surgical Endoscopy* 22(2), 313–325 (2008), doi:10.1007/s00464-007-9727-5
8. Nio, D., Diks, J., Bemelman, W.A., Wisselink, W., Legemate, D.A.: Vascular Surgery: A Systematic Review. *Eur. J. Vasc. Endovasc. Surg.* 33, 263–271 (2007), doi:10.1016/j.ejvs.2006.10.004
9. Martinie, J.B., Smeaton, S.M.: Laparoscopic Approaches to Pancreatic Endocrine Tumors. In: Greene, F.L., Heniford, B.T. (eds.) *Minimally Invasive Cancer Management*, Part 3, pp. 145–157 (2010)
10. Yuh, B.E., Hussain, A., Chandrasekhar, R., Bienko, M., Piacente, P., Wilding, G., Menon, M., Peabody, J., Guru, K.A.: Comparative analysis of global practice patterns in urologic robot assisted surgery. *J. Endourol.* 24(10), 1637–1644 (2010), doi:10.1089/end.2010.0024
11. Borden Jr., L.S., Kozlowski, P.M., Porter, C.R., Corman, J.M.: Mechanical failure rate of da Vinci robotic system. *Canadian Journal of Urology* 14, 3499–3501 (2007)
12. Murphy, D.G., Bjartell, A., Ficarra, V., Graefen, M., Haese, A., Montironi, R., Montorsi, F., Moul, J.W., Novara, G., Sauter, G., Sulser, T., van der Poel, H.: Downsides of Robot-assisted Laparoscopic Radical Prostatectomy: Limitations and Complications. *European Urology* 57(5), 735–746 (2010), doi:10.1016/j.eururo.2009.12.021
13. Antoniou, G.A., Riga, C.V., Mayer, E.K., Cheshire, N.J.W., Bicknell, C.D.: Clinical applications of robotic technology in vascular and endovascular surgery. *Journal of Vascular Surgery* 53(2), 493–499 (2011), doi:10.1016/j.jvs.2010.06.154
14. Zureikat, A.H., Nguyen, K.T., Bartlett, D.L., Zeh, H.J., Moser, A.J.: Robotic-Assisted Major Pancreatic Resection and Reconstruction. *Archives of Surgery* 146(3), 256–261 (2011), doi:10.1001/archsurg.2010.246
15. Kaushik, D., High, R., Clark, C.J., LaGrange, C.A.: Malfunction of the da Vinci Robotic System During Robot-Assisted Laparoscopic Prostatectomy: An International Survey. *Journal of Endourology* 24(4), 571–575 (2010), doi:10.1089=end.2009.0489
16. Basili, V.R., Caldiera, G., Rombach, H.D.: The Goal Question Metric Approach. In: *Encyclopedia of Software Engineering*. Wiley (1994)
17. Thammarak, K., Intakosum, S.: OPI model: A methodology for development metric based on outcome oriented. In: *8th International Joint Conference on Computer Science and Software Engineering, JCSSE (2011)*, doi:10.1109/JCSSE.2011.5930144
18. Joint Commission on Accreditation of Healthcare Organizations, Robot-assisted abdominal surgery. *Comprehensive Accreditation Manual for Hospitals: The Official Handbook (CAMH)*, Oakbrook Terrace, IL (2002)
19. Vincent, C., Moorthy, K., Sarker, S.K., Chang, A., Darzi, A.W.: System approaches to surgical quality and safety - From concept to measurements. *Annals of Surgery* 239(4), 475–482 (2004), doi:10.1097/01.sla.0000118753.22830.41
20. Pronovost, P.J., Berenholtz, S.M., Goeschel, C.A., et al.: High Reliability in Health Care Organizations. *Serv. Res.* 41(4 pt. 2), 1599–1617 (2006), doi:10.1111/j.1475-6773.2006.00567.x

# System Concept for Collision-Free Robot Assisted Surgery Using Real-Time Sensing<sup>\*</sup>

Jörg Raczkowski, Philip Nicolai, Björn Hein, and Heinz Wörn

Karlsruhe Institute of Technology (KIT)  
Institute for Process Control and Robotics, Karlsruhe, Germany  
joerg.raczkowski@kit.edu

**Abstract.** The introduction of robot assistance into the surgical process yields beside all desired advantages also additional sources of potential risks. This is especially due to the fact that the working space of the robot system is overlapping with the patient and the surgical personnel in a narrow environment around the situs. To enable the usage of partially autonomous robotic system in this field, we propose a novel approach which combines an algorithm for guaranteed collision-free path-planning with a real-time monitoring system of the workspace. This paper details the concept as well as the integration plan, showing first results for both components.

**Keywords:** robot assistance, monitoring, safety, collision avoidance, trajectory planning, sensing, registration.

## 1 Introduction

Nowadays, robot assisted surgery is (again) an upcoming technology for the operating room [1]. For comparatively simple surgical treatments, a robot/manipulator system and some manual support are often sufficient. As treatments grow more complex, there is need for support by additional tools. However, the optimal integration of these tools into the OR set up is cumbersome and time consuming. To make the systems more operable, capable and timesaving for the usage in the OR they would benefit from autonomous execution of some of these functions. This approach will in turn help to introduce robotic systems as routinely used assistance systems. For this reason, the Medical Robotics Group of IPR (MeGI) developed a modular platform for robotic surgery called OP:Sense [2] as depicted in Figure 1. This frame is the basis for research on the complete workflow of the different phases of a surgical treatment.

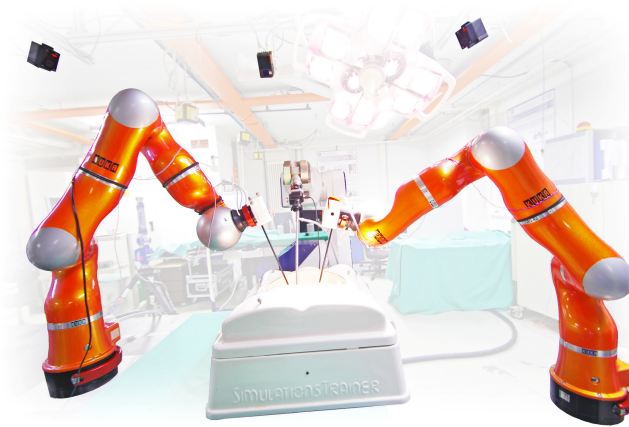
Based on patient data from multiple sources such as imaging and electronic health record (EHR) an individual planning model shall be established. The first phase will be the pre-operative surgical planning which is independent of the tools that will be used during the actual intervention. This plan will then be instantiated and the task

---

<sup>\*</sup> This research work is partially funded by the European Commission within the FP7 project SAFROS under the contract number 248960. The residual equipment was funded within the large equipment program of the German Science Foundation (DFG).

planning for the specifically chosen modules can be performed. An especially important part for robotic surgery is the motion planning of the robotic instruments. This contains trajectories to and from the instruments region of interest. In partially autonomous minimally invasive robotic surgery, these could e.g. be used to automatically position the robot instruments near the trocars and then give full manual control to the surgeon.

In order to plan and guarantee safe robotic trajectories, constraints have to be defined that will lead to a safe execution with included exception handling. This will include supervising the scene for detecting obstacles to be included in the trajectory planning. Path planning itself has to take into account all detected obstacles in order to plan a reliable safe trajectory which doesn't pose any threat to either the patient or the medical staff. The combination of both measures as proposed here leads to robotic movements which are both safe and applicable to the live situation in the operation theatre with many different people and devices interacting in a dynamic way.



**Fig. 1.** OP:Sense platform for robot assisted surgery with two light-weight robots and the operation room supervision system (OSS)

## 2 Planning

### 2.1 Surgical Planning

In the pre-operative planning of the intervention, several medical decisions have to be made. These include the type of intervention (open surgery vs. minimally invasive surgery) as well as decisions more specific to the patients and their anatomy like the position and size of the incision(s) [3]. The latter have to take into account the properties of the surgical robot system being used such as its workspace, degrees of freedom and joint lengths, resulting in a robot configuration optimized for the actual intervention.

During the intervention, the robots positions at the operation table as well as the patient can be assumed to be static, i.e. both the patient and the robot bases are



non-moving. Thus the patient can be regarded as a static obstacle which we propose to initially assess directly before the intervention.

## 2.2 Intra-operative Planning of Safe Zones

To initially assess the patient's geometry and calculate the resulting safe zones in which the robots might work, the sensing system as detailed in III. will be used. After the patient is fully prepared, directly before the intervention itself begins with the first incision, the complete environment is captured by all cameras. This results in a (partially colored) point cloud showing the patient on the OP table which can then be used to build a volumetric model of the scene. Based on the color information the patient's surface can be segmented into an interaction zone (that the robot might collide with) and a safe zone which can't be violated. By combining the zone segmentation with the volumetric model, a safe volume can be constructed which will then be used as an obstacle for trajectory planning.

## 2.3 Trajectory Planning

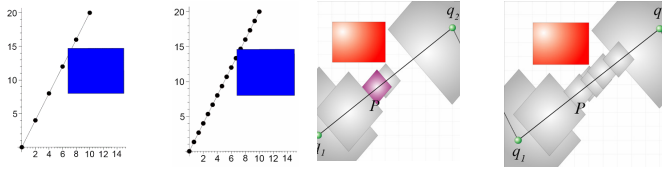
Based on the information given by the paths defined above, we use an automatic path planning algorithm that combines a deterministic local planning algorithm ( $A^*$ ) with a probabilistic global planning algorithm (PRM) in detail presented in [4]. This combination allows to compute collision-free paths in free wide spaces (e.g. from initial position of the robot to the target position) and is especially suited for cluttered and narrow environments (e.g. near the situs). In this connection we use a collision checking approach based on distance computations which allows detecting collisions reliably. This approach is explained in the following.

## 2.4 Collision Checking

As stated before, it is crucial for the targeted application to verify that movements/paths of the robot are collision-free. Typically obstacles detection and movement execution are handled in two different spaces:

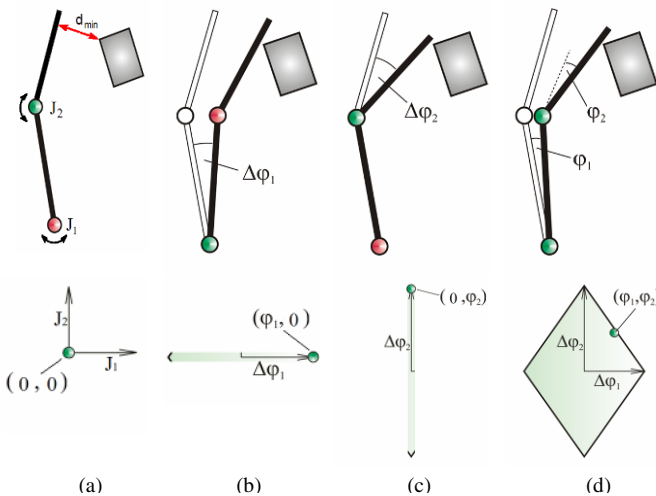
- Work space (W-space): is the space where the robot and the obstacles exist as 3D geometry model.
- Configuration space (C-space): is the mathematical space spanned by all independent joints (only active joints).

Most approaches concerning collision checking of motion paths of robots rely on spot tests which are calculated at discrete positions along the path in C-space. This requires a pre-calculated step-size depending on the minimum size of the expected obstacles. But even if this size is given or can be calculated discrete collision tests can't guarantee that a robot path is collision free (s. Fig. 2). To overcome this deficiency we use an approach to map W-space distances to C-space areas that are collision free [5] (s. Fig. 2) also efficiently used in path planning algorithms [6]. The proposed method eliminates discretisation and resolution based problems when testing a path for collisions.



**Fig. 2.** (left two images) discrete collision checks can fail to detect collisions even if fine sampled, (right two images) our approach transforming W-space distances to C-Space can guarantee that a tested path is collision free

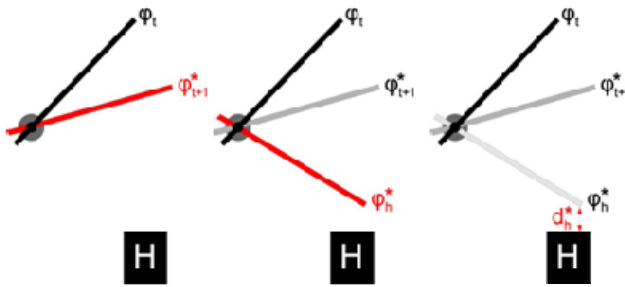
The idea can be summarized in generally determining how much a joint  $J_i$  of a robot can be moved before any affected link collides with an obstacles. To exemplify, consider an obstacle and a device with two rotational joints as shown in Fig. 3. Before moving the device, the minimal distance  $d_{min}$  between the device and obstacles is computed. The distance consumption table translates  $d_{min}$  to a joint value  $\Delta\phi_1$ . After the device moves the joint  $J_1$  by  $\Delta\phi_1$ , the distance between device and obstacle tends to be 0. Due to the fact that  $J_1$  can be moved by  $\Delta\phi_1$  on both directions, a collision free region of  $2\Delta\phi_1$  along the  $J_1$  axis is set (s. Fig 3 (b), C-space). Similar behaviour is observed if only  $J_2$  is actuated. If  $J_1$  and  $J_2$  are rotated at the same time, the joint values are smaller than the ones calculated when moving only one joint ( $\phi_1 < \Delta\phi_1$  and  $\phi_2 < \Delta\phi_2$ ). All joints movements inside the rhombic region are guaranteed to be collision free [BH1]. Based on that, a parameter-free recursive path test can be implemented that is able to securely detect obstacles of arbitrarily size (s. Fig. 3 (right)).



**Fig. 3.** (a) W-space and C-space of a robot with two rotational joints ( $J_1$  and  $J_2$ ). Minimal distance between device and obstacle is computed; (b) only joint  $J_1$  is moved on  $\Delta\phi$  just before it collides; (c) same effect moving joint  $J_2$ ; (d) All robot poses inside the rhombus, in C-space, are collision free.

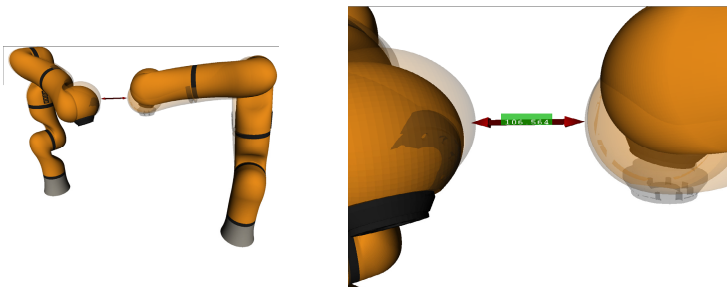
### 2.5 Collision Prediction and Avoidance

The previous two sections describe how collision-free path-planning and safe collision checking of these paths during execution is done. During manual control this concept has to be operator. So during manual control every  $\mathbf{p}$  cycle of the robot controller ( $p*4ms-p*12ms$ ), the future position of the robot is estimated based on the input given by the operator and the current velocity and acceleration (s. Fig. 4).



**Fig. 4.** Left to right: Based on the current input and velocity the future position is estimated at cycle  $t=t+p$ ; based on the maximum possible deceleration the estimated stop position is computed; based on the distance at this position, the robot can be freely moved, is slowed down or – if necessary – stopped

Fig. 5. demonstrates this approach with two KUKA-LBR. The predicted positions are shown in transparent. Between the these virtual robots a distance computation is done. Based on the calculated distance, an appropriate strategy is chosen (emergency stop, reduce velocity, change gear ratio of telecontrol, etc.) . The same principle can be used to emulate virtual fixtures (e.g. guiding channels or virtual workspaces, preventing a tool to leave a defined area). In the proposed approach we favor the use of distance calculations over pure collision checking, offering much more flexibility, even though collision-checking is typically 15 times faster.



**Fig. 5.** Example of two LBR tested for collision in real-time with predicted position (transparent robot) using the approach explained in Fig. 4. In this setup the distance calculation takes  $\sim 1ms$ . For comparison: a pure collision test would only take 0.089 ms.

## 3 Sensing System

### 3.1 Tasks and Requirements

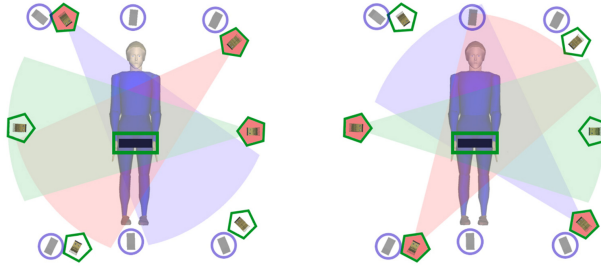
The task of the operation room sensing system (OSS) is to provide real-time scene information of the surroundings of the robots. As the situation in the operation theatre can be highly dynamic and various people surround the operation table, occlusions are a problem which is encountered very often. In the initial phase of an operation, the 3D surface of a patient has to be detected and segmented to determine safe zones, in which the robot might interact with the patient. For laparotomy, colored 3D information is necessary to detect the safe zone where the incision will be performed based on the parts of the patient that are put under medical cloth. In the case of minimally invasive robotic surgery, the locations of the trocars need to be determined. Independent of the type of operation, this step has to be performed only once and is thus not time-critical. During the intervention, the OSS has to provide real-time information about the current environment of the robots for path-planning with collision avoidance (free W-Space). Also, the robots pose has to be redundantly supervised in order to guarantee the correct execution of the planned trajectories and quickly detect deviations. This, of course, should also run in real-time.

### 3.2 Configuration

The OP:Sense setup currently integrates 17 cameras monitoring the scene from different angles, allowing for the supervision of a large working space even with occlusions caused by obstacles and robots. Three different types of cameras are in use to meet the requirements listed above:

- Industrial-standard time-of-flight cameras are used for real-time acquisition of scene information from seven different perspectives. While these offer a high configurability and low latencies, the resulting images only feature a low resolution and no color information.
- A marker-based tracking-system in a six camera configuration provides accurate information for the poses of up to 20 different rigid bodies fitted with highly reflective spheres. These e.g. include the end-effectors of the robots and other instruments.
- MS Kinect cameras offer high-resolution colored 3D-images (RGB-D) at the cost of a comparatively high latency and a lack of configuration options. Recently, the OSS has been expanded to now integrate four Kinect cameras into the OP:Sense setup.

While the combination of these sensors offers many advantages and can thus fulfil the requirements listed above, there are also drawbacks to be dealt with. As all systems rely on active IR signals, it has to be guaranteed that no crosstalk between the cameras distorts the measurements. According experiments showed that the simultaneous usage of multiple PMD S3 cameras leads to major distortions. Thus a shared time- and



**Fig. 6.** Configuration of the OSS with six PMD S3 cameras (green pentagons) and six ARTtrack cameras (blue circles) ceiling-mounted around the workspace as well as one PMD CamCube 2.0 directly over the patient (green rectangle); left: S3 group 1 is triggered, right: S3 group 2 is triggered (different beam colors correspond to different frequencies)

**Table 1.** OSS update rate by different camera types

Sensor / system	Update rates	
	<i>Maximum nominal frame rate</i>	<i>Frame rate achieved in OP:Sense</i>
PMD S3	20 Hz	10 Hz
PMD CamCube	20 Hz	20 Hz
ARTtrack 2.0	60 Hz	20 Hz
MS Kinect	30 Hz	25 Hz
<b>Full scene representation</b>	-	<b>20 Hz</b>

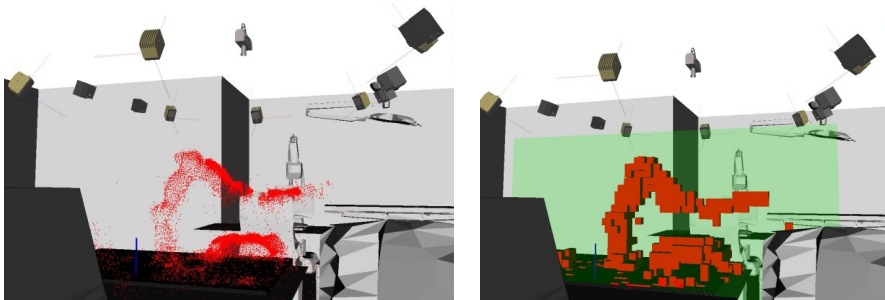
frequency-multiplexing control mechanism has been developed and implemented. The spatial configuration of the cameras as well as the principle of the multiplexing approach is depicted in Fig. 6.

### 3.3 Results

Due to the time-multiplexing control for the S3 cameras, not all cameras can be triggered at their nominal speed, resulting in a lower achievable framerate. In practice, two groups of S3 cameras only run at 10 Hz each instead of their maximum refresh rate of 20 Hz. As all information from the PMD cameras are integrated into one central representation of the scene, this model is nevertheless still with a frame rate of 20 Hz. As in each timestep information from different cameras is integrated, we call this an “interlaced” update rate of 20 Hz. Table 1 shows the detailed results of the current setup.

In a preliminary step for integration of the collision-free path-planning, a voxel carving algorithm has been applied to the scene model. This results in a free space

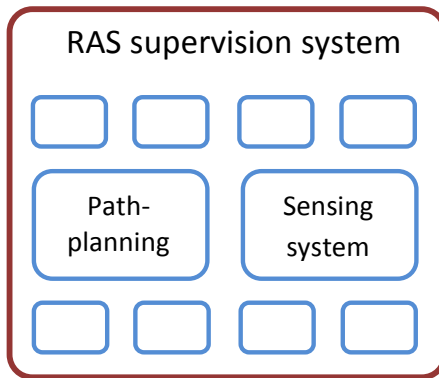
configuration representation which will serve as the input for the collision-free path-planning algorithm. The prototype implementation is running at 4 Hz in a voxel grid of 40 x 20 x 20 cubes as depicted in Fig. 7.



**Fig. 7.** (left) Registered point cloud, (right) resulting voxel space inside the supervised working volume (green)

#### 4 Integration/Plan Execution

The OP:Sense platform contains currently two KUKA LWR4 light weight robots and one Staeubli RX90 robot. The integration of the robotic systems, sensor systems and other embedded devices is done by a high-level control based on Matlab/SimuLink. This central RAS supervision system consists of several modules that each control one part of the intervention, e.g. the OR workflow or the haptic input devices (s. Fig. 8). It will be expanded based on the concept detailed in this paper by the path-planning module which directly interfaces with the sensing system.



**Fig. 8.** Schematic view of the RAS supervision system featuring several components including the new path-planning algorithm and the sensing system

## 5 Summary

The OP:Sense platform offers the opportunity of fast prototyping of surgical robot cells with sensing. Every state during a treatment could be monitored and a decision support is given to the physicians. The paper describes a novel approach for the integration of a path-planning module for collision free trajectories into the platform which is closely coupled to the sensing system, resulting in a reliable and safe human-robot interaction inside the operation room.

**Acknowledgements.** This research work is partially funded by the European Commission within the FP7 project SAFROS under the contract number 248960. The residual equipment was funded within the large equipment program of the German Science Foundation (DFG). The authors thank the EU Commission and the German Science Foundation (DFG) for the support of the research.

## References

1. Taylor, H., Kazanzides, P.: Medical Robotics and Computer-Integrated Interventional Medicine. In: Zelkowitz, M. (ed.) *Advances in Computers*, vol. 73, pp. 217–258. Elsevier (2008)
2. Nicolai, P., Beyl, T., Moennich, H., Raczkowski, J., Wörn, H.: OP: Sense – An Integrated Rapid Development Environment in the Context of Robot Assisted Surgery and Operation Room Sensing. In: *Proceedings of IEEE International Conference on Robotics and Biomimetics (ROBIO 2011)*, Phuket, Thailand, December 7-11, pp. 2421–2422 (2011)
3. Schorr, O., Münchenberg, J., Raczkowski, J., Wörn, H.: KasOp - A Generic System for Pre- and Intraoperative Surgical Assistance and Guidance. In: *Proc. of the 15th Int. Congress and Exhibition of Computer Assisted Radiology and Surgery (CARS)*, Berlin, Germany (2001)
4. Mages, D., Hein, B., Wörn, H.: A Deterministic Hierarchical Local Planner for Probabilistic Roadmap Construction. In: *VDI: VDI Berichte*, vol. 1956, pp. 29–30 (2006)
5. Mages, D., Hein, B., Wörn, H.: Approximated distance conversion from work space to configuration space of robots based on serial kinematics. In: *VDI-Berichte 1841, Robotik* (2002)
6. Hein, B., Wörn, H.: Fast Hierarchical A\* Path Planning for Industrial Robots Based on Efficient Use of Distance Computations. In: *Proc. of 37th International Symposium on Robotics ISR 2006 and 4th German Conference on Robotics Robotik*, Munich (2006)

# Human-Robot Interaction-Based Intention Sharing of Assistant Robot for Elderly People

Jeong-Yean Yang<sup>1</sup>, Oh-Hun Kwon<sup>2</sup>, Chan-Soon Lim<sup>1</sup>, and Dong-Soo Kwon<sup>3</sup>

<sup>1</sup> Human-Robot Interaction Research Center,  
Korea Advanced Institute of Science and Technology,  
291 Daehak-ro, Yuseong-gu, Daejeon 305-701, Korea  
{jyyang, limcs}@robot.kaist.ac.kr

<sup>2</sup> Dept. of Robotics,  
Korea Advanced Institute of Science and Technology,  
291 Daehak-ro, Yuseong-gu, Daejeon 305-701, Korea  
kwonoh@robot.kaist.ac.kr

<sup>3</sup> Dept. of Mechanical Engineering,  
Korea Advanced Institute of Science and Technology,  
291 Daehak-ro, Yuseong-gu, Daejeon 305-701, Korea  
kwonds@kaist.ac.kr

**Abstract.** Intention reading is considered as an important issue in the field of human-robot interaction (HRI). Specifically, for cooperation between a human and a robotic system, a robot should be aware of a given situation and human intention, which necessarily requires high-level knowledge and balanced interaction. For the disabled and elderly, who may not be familiar with manipulating robotic systems, intention sharing between two different agents is preferable for accurate intention reading. In this paper, we focus on the effect of intention sharing of a human user and a robot assistant to improve cooperation.

**Keywords:** intention sharing, human-robot interaction, human-robot cooperation.

## 1 Introduction

With an increasing population of elderly people, assistant robots that help a human user cope with a disability have become a popular area of research in robotics as an extension of rehabilitation robots. Considering the premature development of commercial robotic systems that provides actual service in human's daily life, assistant robots have been focused on its functional purposes: robot-aided guiding for blind people, robot manipulators for disabled people, and so on.

In this paper, we focus on a robot aided vehicle for human walking. As described in Fig. 1, a walker is a common auxiliary device for elderly people who experience difficult walking independently. Its passive mechanical frame assists the human user in standing and walking. While the conventional walker device is human powered, the proposed system, named Smart Walker, has four motorized wheels and the active



mechanism is controlled by a force-feedback joystick and touch-based input command. The active actuation is designed for the medical purpose of rehabilitation: lifting a human from a bed and supporting the human's weight during walking.

On the other hand, the proposed robotic walker operates by human command. In addition, uncertainty from the human's hand becomes more important with typical passive type devices. Misunderstanding of user intention and unnecessary repetition of command changes have greater impacts with regard to safety. In addition, elderly people may not be familiar with informatics devices, and the user group is physically handicapped persons. Thus, the robotic walker is a good example for an assistant robot operated through human-robot interaction (HRI) [1]. The robotic walker cooperates comprehensively with the human while providing refined information considering the difficulties experienced by the user group.

In the proposed system, the feedback of a given command is divided into two areas: informative modalities such as visual, haptic, and auditory feedback, which are common in general robotic systems; and physical movements of the robotic walker by controlling the motor driven wheels. With informative feedback, the cooperation in the robotic walker system enables the elderly user to reach their destination.

Most elderly people born before 1950 feel difficulty with man-machine interfaces. They are not familiar with computer-based operation and furthermore may have less concentration for robot control owing to cognitive impairment. We assume that robot intelligence and smart behavior are preferable for elderly people. Moreover, human intention is not instantly observed. The human intention about where to go depends on the series of joystick commands. However, the robot only recognizes which direction the user wants to go. From the viewpoint of a robotic system, a human's intention about where to go is not explicitly observed by joystick command, and intention reading can be achieved by understanding the relationships among a given situation. Furthermore, the interaction period is another issue. For instance, a human may feel surprise at sudden environmental changes while their happiness slowly changes during a long term period.

Considering the cooperation between an elderly user and a robotic walker, as described in Fig. 1, it is desirable that the robot system be able to read the human's intention and automatically compensate the human's control inputs in a consistent manner. This paper focuses on intention reading by situation awareness and intention sharing by extension of the HRI method. Instead of the autonomous locomotion problem [2], the assistant robot should be aware of the user's destination, and should also be able to estimate human intention properly. Thus, human preference about surrounding objects and a rule-based user model are considered as landmarks for the user's intention regarding his or her destination.

Recalling that the function of the robotic walker is to assist human locomotion, frequent changes of human commands imply that the two agents do not share a common intention. While the control inputs from the joystick correspond to local intention, the ultimate goal is not revealed in every instance. From the results of a reasoning process during interaction, the robot system tries to estimate the human intention. Additionally, assuming that elderly people are novices with respect to using visual feedback, human friendly dialogues are designed to share the reasoning results by confirming the human's intentional goal.

The success of this approach is thus evaluated by the number of input commands that human requires to change direction. A reduction of commands implies that the two agents cooperate with each other to move toward the same goal by sharing intention.

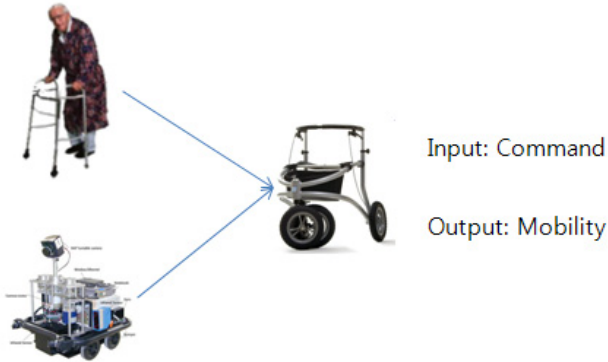


Fig. 1. Overview of the robotic walker

## 2 Intention Reading

The intention reading of the robotic walker is designed by a rule-based reasoning model in the domain of locomotion behaviors [3~5]. An elderly person and a walker perform several tasks such as approaching another human or a specific position, going out of his or her home, and doing exercise for rehabilitation. This domain covers some general aspects of the daily life of elderly people. In Table I, the keywords in the proposed domain are described. The keywords describe the symbolic representation that is possibly recognized by robot sensors with the help of position, emotion detection, touch sensors, and vision system-based object detection.

For instance, the going out task can be refreshing but it also increases the possibility of getting cold. The objects at home include a pet, cellphone, bed, and table, each of which is related to other features. A cellphone provides the possibility of more talking, which implicitly increases refreshment and reduces boredom. A pet contributes to the human's happiness while it requires laborious work that affects health. The network among keywords is directly or indirectly connected in a complex manner and its effects for human's preferences are calculated by the following connections. In this case, about one hundred rules are applied to describe the proposed domain.

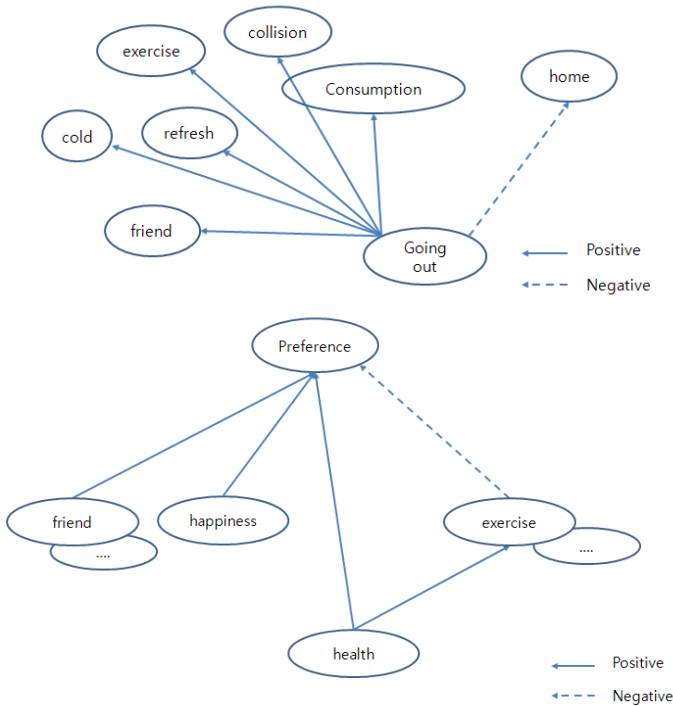
Each rule increases or decreases design factors such as emotion and preference. Exercise supports better health but elderly people have the tendency of cutting it. In the proposed model, elderly people feel the highest preference from the keyword family and the lowest from hunger. All rules are defined for inducing other cases between  $[-1, 1]$ ; minus one is the maximum negative value and one is the maximum positive value. While the direct relationship of keywords has a value between  $-1$  and

1, the indirect relationship diminishes, as in the manner of a probabilistic network model. These rule sets are recursively calculated for estimating which and how much each keyword induces other keywords.

In Fig. 2, the role of direct and indirect relationships is described. Health directly increases the preference and exercise respectively while the exercise itself decreases the overall preference. In other words, health is one of the preferable keywords for elderly people however exercise decreases the preference even though health increases exercise. In our experiments, the left side of the table in Fig. 3 has a set of keywords and the right side corresponds to the queue for describing the current situation. For example, the keywords home, pet, and friend indicate a preference level over 4.

**Table 1.** Keywords defined in the proposed domain

Keywords
cellphone, talking, friend, pet, labor, dirt, touch, hit, friend, son, family, health, approach, collision, anger, hunger, boring, passenger, wallet, bed, sleep, table, wait, hunger, consumption, wealth, going out, refresh, satisfaction, tv, cold, home, happiness, exercise, fatigue, meal, noon, evening, supper, breakfast, lunch,



**Fig. 2.** Examples of rule sets for “going out” and “preference”

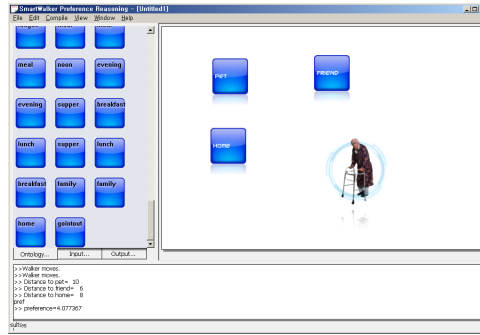


Fig. 3. Icon-based intention reading software environment

### 3 Intention Sharing

In the right part of Fig. 3, the robotic walker estimates the current intention of the user by observing the surrounding situation. The observed keywords are stored into a reasoning queue for estimating the current preference value. In an actual case, the effects of the symbolic keywords gradually diminish by passing through the given environment. While the robotic walker moves by keeping pace with the human user, the roles of surrounding symbols are dependent on elapsed time and observation, as they are forgotten by the human user.

The time constant has a maximum value when symbols are observed. As the robot walker moves, the observed symbols are added into the queue and their time dependent effects are also estimated by a reasoning process. When the symbol disappears, its effect starts to decrease. The effect of the time constant is multiplied by the positive and negative values of keywords. A lower time constant implies smaller effect of keywords stored in the reasoning queue. Thus, the keyword meal has the longest time duration and it slowly increases hunger.

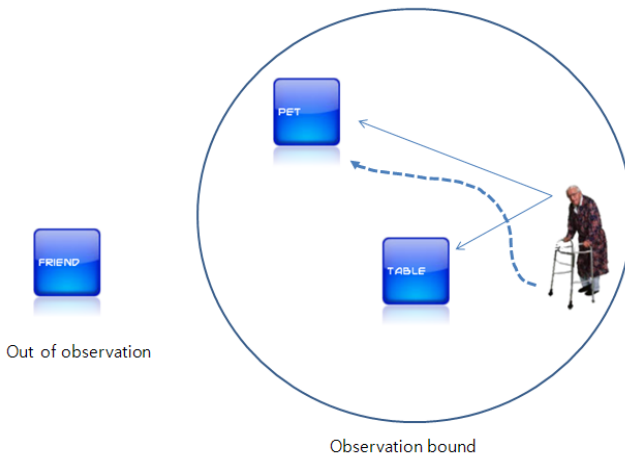
Assuming that the radius of observation is limited, as described in Fig. 4, the surrounding symbols are related to the distance metric. The inputs by the joystick control the moving direction of the robotic walker and the preference values from observed symbols are calculated in order to estimate the current human intention. As depicted in this example, the preference within the observation bound indicates that the human intends to move toward the pet by avoiding collision with the table. In this manner, the robot system estimates the user's current intention from the surrounding objects and situations, and finally the maximum value is estimated to be the user's intention.

As a result, the user's command is affected by the results of intention reading estimated for maximizing preference. Our basic assumption implies that frequent meaningless commands can be reduced by intention sharing. When an obstacle is detected, the robotic walker determines to avoid it slowly by a reasoning process. Abrupt changes of the trajectory increase the user's confusion, and consequently the direction vector is gradually changed toward higher position. This is a suitable scenario because the human and walker system are designed for slow locomotion.

For walking toward a distant position, a small effect of avoidance is continuous before near target positional symbol. However, the intention reading is restricted within the observation bound. The intention reading depends on the observation queue, which stores the symbols in order of appearance. This indicates that intention during long-term operation cannot be estimated. For example, if the friend in Fig. 4 is the ultimate goal, the proposed intention reading does not explain the long-term locomotion.

On the other hand, intention sharing is designed for improving interactivity between both agents [6]. A vector difference between the human's intentional direction and the robot's estimated direction implies disagreement over intention. When the human starts the going out task in order to meet his friend, the goal of the task cannot be precisely estimated owing to the limitation of unseen observation. Thus, the vector difference becomes larger when disagreement between the agent's intention grows. In the proposed rule-based model, the going out keyword induces positivity of friend, exercise, health, and collision. Therefore, the reasoning model creates queries about the real reason for the current task. The candidates are chosen by the user's additional input, and then a preference model is reconstructed by considering the friend at a remote site and the objects in the local site. The subtask that generates a query of the hidden intention contributes to reducing the difference in the vector direction related to disagreement of intention. This is described in Fig. 5.

The reasoning of the long-term task is related to the history of the state transitions. Thus, in many cases, the reasoning of the symbolic queue in the proposed method is not sufficient for history-based reasoning. However, the superiority of reasoning in a long-term period is still questionable. Considering that the major purpose of the walker is not to carry people but rather to assist their walking, interaction at the local site appears to be more desirable for practical purposes.



**Fig. 4.** Observation bound with respect to the relative distance from human to surrounding symbols

- Which one is your goal?
- 1) Friend
  - 2) Exercise
  - 3) Health
  - 4) Collision

**Fig. 5.** Query example for sharing real intention. The current task “going out” induces the above four candidates.

## 4 Experimental Results

The experimental system is composed of a PC-based architecture for controlling the robotic system and reasoning in a given environment. The movement of the robotic walker is controlled by touch-based joystick inputs. For convenience, the motion is changed by up, down, left, and right commands. In addition, an Android-based small tablet that provides touch, display, and sound generation is used for displaying dialogue and transferring the user’s command inputs. These two different types of systems are connected with a USB cable and a TCP/IP-based network connection is used with the help of a bridge function.

The reasoning process of metaphysical features is estimated as described in Table II. In this case, the keyword family has the maximum preference. Among a hundred rules, the keyword family is the most positive keyword to induce happiness, talking, reduced boredom, and so on.

In table III, many keywords for positive and negative preferences are introduced. They are suitably matched with user’s daily life. While the keyword going out increases refreshment and possibility of meeting friend, it also increases diseases and consumption. In the rule set, hunger induces anger and decreases health. Therefore, it has the lowest preference in this domain.

In table IV, the effect of intention sharing is shown. When the overall moving area is not large and the area is nearly the same as the observation area, the effect of intention sharing is not evidently shown. However, in the case of a three-fold increase relative to the the observation area, intention sharing shows better performance for estimating the ultimate goal. Considering that our test does not guarantee global locomotion, the effect of querying the ultimate goal contributes to reduced occurrences of disagreement.

Unfortunately, in this preliminary research the role of elapsed time is not clearly shown. We speculate that the working period with a given test space is too short for the time-based interaction to produce sufficient effects for intention reading.

**Table 2.** Reasoning results

Keywords	Most Positive		Most Negative	
Preference	Family	3.1	Hunger	-3.1
Happiness	Home	7.8	Dirt	-0.7
			Consumption	-0.7
Health	Meal	6.2	Hunger	-2.0
			Sleep	0.4
Exercise	Going out	3.0	Wait	0.4
			Fatigue	0.4
Boring	Home	5.2	Going out	0.5
			Pet	0.5
Hunger	Going out	6.4	Wealth	-1.0
			Sleep	0.4
Talking	Home	5.6	Wait	0.4
			Fatigue	0.4

**Table 3.** Preference results

Keywords	Most Positive
Friend	1.34
Going out	-0.53
Exercise	-2.24
Talking	0.49
Pet	0.74
Family	3.06
Hunger	-3.1
Home	1.99

**Table 4.** Reasoning results

	Average Number of Abrupt Input Changes in a Local Area (Within Observation Bound)	Average Number of Abrupt Input Changes in a Wide Area
Preference	13/100	42/100
Happiness	9/100	19/100

## 5 Conclusion

In this paper, we focused on the effects of intention reading and sharing issues. Human intention is estimated by a typical rule-based reasoning process of the surrounding observed objects. The results of the recognized situation are shared by a query system in order to confirm the walking direction. In many cases, intention reading is an appropriate solution for a robot to be aware of the current situation. Therefore, a robot can understand the human's real intention and can determine a better interaction method.

In the proposed experiment, we focused on intention sharing and how it improves interaction performance. The disagreement between two agents was evaluated by counting abrupt command changes. The comparison between the human command and the robot's estimated direction readily showed interaction failures. We tried to improve the interaction and focused on when the agents disagree with each other.

As expected, intention sharing is a more efficient method for specifically cooperative behaviors. This system still has the possibility of amplifying user's confusion caused by robot autonomy. In further works, intention sharing in multiple stages will be an interesting area of research for determining questions that will be more effective during overall interaction.

**Acknowledgements.** This project is financially supported by the Ministry of Knowledge and Economics through the study of "ADL support system for the elderly and disabled people".

## References

1. Cao, Y.U., Fukunaga, A.S., Kahng, A.B.: Cooperative Mobile Robotics: Antecedents and Directions. *Autonomous Robots* 4, 1–13 (1997)
2. Li, J., Pan, Q., Hong, B.: A New Approach of Multi-Robot Cooperative Pursuit Based on Association Rule Data Mining. *International Journal of Advanced Robotic Systems*, 329–336 (2010)
3. de A. Barreto, G., Araujo, A.F.R., de O. Rosa, M.: A Reactive Rule –based System for Trajectory Planning of a Mobile Robot. In: Proc. of the IASTED/AAAI Conference on Artificial Intelligence and Soft Computing, pp. 419–422 (1998)
4. Ciliz, M.K., Isik, C.: Fuzzy rule-based Motion Controller for an Autonomous Mobile Robot. *Robotica* 7, 37–42 (1989)
5. Kim, D.J., Song, W.K., Han, J.S., Bien, Z.Z.: Soft Computing-based Intention Reading Technique as a Means of Human-Robot Interaction for Human Centered System. *Soft Computing- A Fusion of Foundations, Methodologies, and Applications* 7(3), 160–166 (2003)
6. Tomasello, M., Carpenter, M., Call, J., Behne, T., Moll, H.: Understanding and Sharing Intentions: The Origins of Cultural Cognition. *Behavioral and Brain Sciences* 28, 675–735 (2005)



# Author Index

- Ang, Wei Tech 341  
Ang Jr., Marcelo H. 297  
Aye, Yan Naing 341
- Baeg, SeungHo 263  
Barsi, Attila 369  
Basso, Filippo 53  
Beetz, Michael 127  
Behnke, Sven 5  
Bleuler, Hannes 369  
Bulavintsev, Sergey 105
- Campoy, Pascual 165  
Cervera, Enric 149  
Chang, Stephen K.Y. 297  
Cho, Kuk 263  
Chui, Chee Kong 297  
Courbon, Jonathan 29, 43
- Dario, P. 323  
Deniša, Miha 137  
Dillmann, Rüdiger 309  
Di Marco, Daniel 117  
Dombre, E. 323  
Domínguez, Salvador 5
- Fiorini, Paolo 381
- García-Bermejo, Jaime Gómez 5  
Gasperotti, Lorenza 381  
Göktoğan, Ali Haydar 193  
Gu, Dongbing 17  
Guerrero, J.J. 95
- Han, Jae-Hung 277  
Häussermann, Kai 117  
Hein, Björn 391  
Hong, Sung Kyung 181  
Hu, Huosheng 17  
Huang, Weimin 297
- Jäkel, Rainer 309  
Jörg, Stefan 357
- Kang, Byeong Ho 333  
Khampitak, Kovit 351  
Kim, Chi Yen 333  
Kim, Eun-Gyung 235  
Kim, Hee-Je 253  
Kim, Hyuk 105  
Kim, Jong-Hwan 243  
Kim, Sunghoon 209  
Klodmann, Julian 357  
Konietschke, Rainer 357  
Korrapati, Hemanth 29, 43  
Kwon, Dong-Soo 401  
Kwon, Oh-Hun 401  
Kyung, Jinho 217
- Le, M.Q. 323  
Lee, Han Nam 253  
Lee, Hyunjin 225  
Lee, Jun-Seong 277  
Lee, Min Cheol 333  
Lee, Sang Cheol 181  
Lee, Sukhan 69  
Levi, Paul 117  
Lim, Chan-Soon 401  
Liu, C. 323

- Liu, Hao 287  
 Liu, Jiang 297  
 López-Nicolás, G. 95  
  
 Maeda, Masateru 287  
 Martinet, Philippe 149  
 McDonald-Maier, Klaus 17  
 Meissner, Pascal 309  
 Mejias, Luis 165  
 Mellado-Bataller, Ignacio 165  
 Mencassi, A. 323  
 Menegatti, Emanuele 53  
 Mezouar, Youcef 29, 43  
 Michieletto, Stefano 53  
 Morandi, Angelica 381  
 Moughlbay, Amine Abou 149  
 Munaro, Matteo 53  
  
 Neadsanga, Wathanyu 351  
 Nicolai, Philip 391  
 Noda, Ryusuke 287  
  
 Ogay, Dmitriy 105, 235  
 Oleari, Elettra 381  
 Olivares-Mendez, Miguel A. 165  
 Omedes, Jason 95  
  
 Pagello, Enrico 53  
 Park, Chanhum 217  
 Park, Jang-Sik 105  
 Park, Jonghyun 209  
 Park, Joongki 209  
 Park, Sangdeok 263  
 Park, Yongjin 69  
 Pestana, Jesús 165  
 Poignet, P. 323  
 Pongpimon, Thantakorn 351  
  
 Rabenoroso, K. 323  
 Raczkowsky, Jörg 391  
  
 Ribeiro, David 369  
 Romahn, Fabian 309  
 Ryu, Jee-Hwan 105, 235  
 Ryuh, Young Sun 181, 225  
  
 Sanchez, L. Alonso 323  
 Schmidt-Rohr, Sven R. 309  
 Sengül, Ali 369  
 Shee, Cheng Yap 341  
 Sikander, Sakura 253  
 Sohn, Juchan 209  
 Su, Yi 297  
 Sukkarieh, Salah 193  
  
 Taechajedcadarung-sri, Sirivit 351  
 Tahk, Min-Jea 193  
 Tenorth, Moritz 117, 127  
 Theodoridis, Theodoros 17  
  
 Ude, Aleš 137  
  
 Verga, Monica 381  
  
 Won, Dae-Yeon 193  
 Wörn, Heinz 391  
 Worst, Rainer 5  
  
 Yang, Gi-Hun 225  
 Yang, Jeong-Yean 401  
 Yang, Liangjing 297  
 Yang, Tao 297  
 Yoo, Jeong-Ki 243  
 Yoo, Tae Suk 181  
 Yoon, Sung Min 333  
 Yun, Dongwon 217  
  
 Zalama, Eduardo 5  
 Zemitì, N. 323  
 Zhao, Su 341  
 Zweigle, Oliver 117