James L. Crowley
Bruce A. Draper
Monique Thonnat (Eds.)

LNCS 6962

Computer Vision Systems

8th International Conference, ICVS 2011
Sophia Antipolis, France, September 2011
Proceedings

Springer

Lecture Notes in Computer Science 6962

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Editorial Board

David Hutchison
Lancaster University, UK

Takeo Kanade
Carnegie Mellon University, Pittsburgh, PA, USA

Josef Kittler
University of Surrey, Guildford, UK

Jon M. Kleinberg
Cornell University, Ithaca, NY, USA

Alfred Kobsa
University of California, Irvine, CA, USA

Friedemann Mattern
ETH Zurich, Switzerland

John C. Mitchell
Stanford University, CA, USA

Moni Naor
Weizmann Institute of Science, Rehovot, Israel

Oscar Nierstrasz
University of Bern, Switzerland

C. Pandu Rangan
Indian Institute of Technology, Madras, India

Bernhard Steffen
TU Dortmund University, Germany

Madhu Sudan
Microsoft Research, Cambridge, MA, USA

Demetri Terzopoulos
University of California, Los Angeles, CA, USA

Doug Tygar
University of California, Berkeley, CA, USA

Gerhard Weikum
Max Planck Institute for Informatics, Saarbruecken, Germany

James L. Crowley Bruce A. Draper
Monique Thonnat (Eds.)

Computer Vision Systems

8th International Conference, ICVS 2011
Sophia Antipolis, France, September 20-22, 2011
Proceedings

Springer

Volume Editors

James L. Crowley
INRIA Grenoble Rhône-Alpes Research Centre
655 Avenue de l'Europe, 38330 Montbonnot, France
E-mail: james.crowley@inrialpes.fr

Bruce A. Draper
Colorado State University, Department of Computer Science
Fort Collins, CO 80523, USA
E-mail: draper@cs.colostate.edu

Monique Thonnat
INRIA Sophia Antipolis
2004 route des Lucioles, BP 93, 06902 Sophia Antipolis, France
E-mail: monique.thonnat@inria.fr

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-23967-0 e-ISBN 978-3-642-23968-7
DOI 10.1007/978-3-642-23968-7
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2011936033

CR Subject Classification (1998): I.5, I.4, I.3, I.2.10, I.5.4, C.3

LNCS Sublibrary: SL 1 – Theoretical Computer Science and General Issues

© Springer-Verlag Berlin Heidelberg 2011
This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law.
The use of general descriptive names, registered names, trademarks, 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.

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

Computer Vision is the science and technology of machines that see. The dominant scientific conferences in computer vision, such as ICCV, CVPR and ECCV, concentrate on theories and models for obtaining information from images and image sequences. The intensely competitive nature of these conferences leaves little room to address the systems and engineering science issues necessary for transforming vision theory into practical vision systems. The International Conference on Vision Systems, ICVS, was created to fill this gap.

The first ICVS was organized in December 1999 at Las Palmas in the Canary Islands to provide a forum for research on systems architecture, benchmarks and performance evaluation. Since that time, the field of computer vision has made impressive progress, with the emergence of reliable techniques for interest point detection and matching, image indexing, category learning, object detection, recognition and classification. Meanwhile, computing power has become much less of a barrier to building vision systems. Desktop computers have evolved from machines with 100-MHz clocks and a few megabytes of memory to multi-core architectures with multiple GHz clock processors and gigabytes of memories. This progress has been reflected in the emergence of techniques documented in ICVS conferences in Toronto in 2001, Vancouver in 2002, Graz in 2003, New York in 2005, Santorini in 2007, and Liège in 2009. We continued this tradition with the 8th International Conference on Vision System in Sophia Antipolis.

The conference Program Committee received 58 submitted papers. Each paper was assigned to three reviewers from among the 31 members of the review committee, leading to the selection of 22 papers for oral presentation at the conference. These were organized into seven sessions, showcasing recent progress in the areas performance evaluation, activity recognition, control of perception, and knowledge-directed vision. The program was completed by presentations from three invited speakers, exploring areas of particularly high potential for impact on the engineering science of vision systems.

The emergence of mobile computing has led to a revolution in computer vision systems. The ubiquitous nature of cameras on mobile telephones and tablets has enabled new applications that combine vision and mobility with ubiquitous access to information over the Internet. However, the limited computing and electrical power of mobile platforms has limited these systems. This is set to change with the emergence of low-power GPUs specifically designed to support computer vision and graphics on mobile devices. The invited talk by Joe Stam of NVIDIA described the emerging use of GPUs as a hardware platform for vision systems on personal computers and described the new generation of devices for mobile platforms such as cameras and tablets.

The use of open source systems has had a profound impact on all areas of informatics, including computer vision. Our second invited speaker, Gary Bradski, has been an early champion of open source software for computer vision. Gary launched the OpenCV library as a standard open source repository for computer vision in the late 1990s. Since that time OpenCV has emerged as the standard source for vision software, enabling rapid prototyping of computer vision systems for an increasingly diverse variety of applications. In our second invited talk, Gary Bradski retraced the emergence of OpenCV as a standard, and presented the wide scope of applications that OpenCV has made possible.

Video surveillance and monitoring is currently one of the largest and most active application areas for Computer Vision systems. In our third invited talk, Mubarak Shah discussed the software engineering aspects, as well as the vision techniques used for object detection, tracking and activity recognition in both ground-based and aerial surveillance systems.

We wish to thank all authors who submitted papers for the program, as well as the Program Committee for providing reviews as well as feedback to the authors. We would also like to thank the local organizers for the local arrangements. Most importantly, we thank the participants, for providing discussion and debates that stimulate progress in the science and technology of computer vision systems.

July 2011

James L. Crowley
Bruce Draper
Monique Thonnat

Organization

General Chair

Monique Thonnat

Program Co-chairs

James L. Crowley	INRIA Grenoble and Grenoble INP, France
Bruce Draper	Colorado State University, USA

Program Committee

Helder Araujo	University of Coimbra, Portugal
Bob Bolles	SRI Research, USA
David Bolme	Colorado State University, USA
Alain Boucher	IFI, Vietnam
Francois Bremond	INRIA, France
Jorge Cabrera	University of Las Palmas, Gran Canaria, Spain
Henrik Christensen	Georgia Tech, USA
Regis Clouard	GREYC Laboratory, France
Patrick Courtney	Perkin Elmer Life Sciences, UK
David Demirdjian	MIT, USA
Frederic Devernay	INRIA, France
Bob Fisher	University of Edinburgh, UK
Simone Frintrop	Rheinische Friedrich-Wilhelms-Universität Bonn, Germany
Vaclac Hlavac	Czech Technical University in Prague, Czech Republic
Jesse Hoey	University of Dundee, UK
Ales Leonardis	University of Ljubljana, Slovenia
James Little	University of British Columbia, Canada
Yui Man Lui	Colorado State University, USA
Giorgio Metta	University of Genoa, Italy
Bernd Neuman	University of Hamburg, Germany
Lucas Paletta	Joanneum Research, Austria
Justus Piater	University of Innsbruck, Austria
Fiora Pirri	University of Rome "La Sapienza", Italy
Ioannis Pratikakis	Democritus University of Thrace, Greece
Sajit Rao	Co57, Cambridge, MA, USA
John Alexander Ruiz Hernandez	INRIA, France

Gerhard Sagerer	University of Bielefeld, Germany
Bernt Schiele	Max Plank Institute, Saarbrücken, Germany
Arcot Sowmya	University of New South Wales, Australia
Muhammed Nayeem Teli	Colorado State University, USA
Marc Van Droogenbroeck	University of Liege, Belgium
Sergio Velastin	Kingston University, UK
Markus Vincze	Vienna University of Technology, Austria
Sven Wachsmuth	University of Bielefeld, Germany

Local Organization and Arrangements

Agnès Cortell	INRIA Sophia Antipolis, France

Table of Contents

Vision System

Control of Perception (I)

Performance Evaluation (II)

Activity Recognition

Control of Perception (I)

Knowledge Directed Vision

Control of Perception (II)

Knowing What Happened - Automatic Documentation of Image Analysis Processes

Birgit Möller, Oliver Greß, and Stefan Posch

Institute of Computer Science, Martin Luther University Halle-Wittenberg,
Von-Seckendorff-Platz 1, 06120 Halle/Saale, Germany
{birgit.moeller,oliver.gress,stefan.posch}@informatik.uni-halle.de

Abstract. Proper archiving or later reconstruction and verification of results in data analysis requires thorough logging of all manipulative actions on the data and corresponding parameter settings. Unfortunately such documentation tasks often enforce extensive and error prone manual activities by the user. To overcome these problems we present `Alida`, an approach for fully automatic documentation of data analysis procedures. Based on an unified operator interface all operations on data including their sequence and configurations are registered during analysis. Subsequently these data are made explicit in XML graph representations yielding a suitable base for visual and analytic inspection. As example for the application of `Alida` in practice we present `MiToBo`, a toolbox for image analysis implemented on the basis of `Alida` and demonstrating the advantages of automatic documentation for image analysis procedures.

Keywords: automatic documentation, meta data, XML, processing graph, image analysis.

1 Introduction

In many fields of application the amount of data that needs to be analyzed is constantly growing. Thus, manual analysis of large datasets gets impossible and the need for automatic analysis procedures arises. A further driving force behind automatic data analysis is its potential to produce objective and reproducible results compared to the subjective outcome of human inspection of data. However, besides the results per se, it is often also of interest how the results where achieved. Hence, monitoring data analysis yields a documentation of the process and facilitates verification later-on, as well as long-term archival storage.

Documentation of analysis processes can, by nature, be formalized more easily for automatic procedures, as the input data and the algorithmic operations performed including parameter settings determine the output in a deterministic way in most cases. However, to manually create a detailed documentation is tedious and error prone. In addition, it has to be taken into account that software usually evolves over time. While the development itself is tracked by revision control systems, the execution of programs and generation of actual results is usually not linked to these systems.

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 1–10, 2011.
© Springer-Verlag Berlin Heidelberg 2011

In this paper we present our approach `Alida`[1] for fully automatic documentation of analysis processes to ease the generation of documentation, automatic reconstruction and verification of analysis results. The documentation extracted includes all input and output objects involved, manipulations performed with all relevant parameters, the flow of data, and also software versions used. All this information is summarized in the *processing graph* which is implicitly defined by any analysis process, and made explicit by `Alida`.

Our approach is based on two fundamental building blocks. To monitor all data manipulations of an analysis process, manipulations are realized in terms of *operators* providing unified interface definitions and following clearly specified invocation procedures. Secondly, all objects ever manipulated are registered within the system and linked to manipulating operators resulting in an implicit representation of the processing graph. For each output object this graph can subsequently be made explicit and stored in terms of an XML representation. This allows for convenient visualization, reconstruction and verification of results at a later point in time, and also for long-term archiving, e.g., in databases. `Alida` enforces minimal restrictions for users and programmers to automatically generate the documentation, interfering as little as possible with usual software development cycles, and resulting in automatic documentation with a minimum of overhead.

One important area where the automatic analysis of data has become an indispensable tool during the last decades are applications dealing with data from optical sensors and digital cameras, or relying on the analysis of visual information in general. In particular, in biomedical image analysis the amounts of data are growing fast, and objectivity of results is an important issue as the variation within manual analysis results of humans is often immense [13].

We demonstrate `Alida`'s practical relevance regarding this field using as an example `MiToBo`, a toolbox for biomedical image analysis. It makes use of `Alida`'s documentation capabilities to allow for parameter logging and process documentation in biomedical research as well as in image analysis algorithm development.

The remainder of this paper is organized as follows. In Section 2 we briefly review related work, while Section 3 presents a detailed description of the automatic documentation concept and its realization in Java. In Section 4 we deal with the visualization of the processing history, describe practical experiences applying `Alida` in Section 5, and make some concluding remarks in Section 6.

2 Related Work

Documentation of analysis processes and data manipulation actions is crucial in many fields of applications. In particular computer-based data management and analysis often requires detailed logging of what happens to given data, e.g., to reconstruct manipulative actions later-on or to verify data analysis results. Obviously such logging procedures should be done automatically as explicit manual documentation by the user is cumbersome and error prone.

[1] **A**utomatic **L**ogging of Process **I**nformation in **D**ata **A**nalysis, `http://www.informatik.uni-halle.de/alida`

The levels on which automatic logging and documentation may be accomplished vary significantly, starting from journaling file systems and logging procedures on the operating system level (e.g., [3,8]), continuing with automatic tracing of user interactions (e.g., [11]), and ending up with explicit extraction of semantic process meta information [10]. In particular the latter option, however, sometimes requires introspection of applied software. In the scientific community proper documentation of data analysis procedures and scientific result extraction is essential for scientific authenticity and progress. Thereby the documentation is supposed to subsume information about the data itself and its acquisition, analysis and manipulation, and of course the results accomplished [6]. In several fields, e.g., in bioinformatics (microarrays, proteomics), activities have evolved to define common standards for data and process documentation [2,12].

With regard to biomedical imaging the Open Microscopy Environment (OME)[2] is working on standardized documentation of microscope image data based on its own XML format *OME-XML* [6]. Besides detailed acquisition device data and image meta data, also options for documenting analysis steps are intended [9]. However, while device and image meta data can often be extracted automatically, gathering of analysis procedures' meta information usually requires explicit manual activities from user side.

3 Automatic Documentation

Here we introduce the concept of operators as the only place of data manipulation leading to the interpretation of an analysis process as a processing graph. Then we give some details of our Java implementation of the automatic documentation framework `Alida` and the external representation of the processing history.

3.1 Operators

The background for automatic documentation is the concept of operators, being the only places where data are processed and manipulated (see, e.g., [5]). The data to be processed are considered as input objects of an operator. The types of these objects are application dependent, e.g., in computer vision images or sets of segmented regions are common. An operator receives zero or more input objects, and its behavior is controlled or configured via parameters. Typical examples for parameters are sizes of kernels, structuring elements, filter masks or weighting constants. The application of an operator produces zero or more output objects. The types of these objects are the same as for the inputs as in virtually all cases an operator output may act as the input to other operators.

The basic assumptions of the documentation framework are as follows. The output objects resulting from the application of an operator depend only on

- the values of the input objects,
- parameters of the operator, and
- the software version of the operator upon invocation.

[2] `http://www.openmicroscopy.org/site`

These assumptions are probably easy to agree upon. Note that the first two state principles of reasonable software design avoiding, e.g., side effects or dependencies on global variables. However, these basic assumptions underscore the benefits of proper documentation of operators. If we know these bits of information for a data object, e.g., an image stored in a file, we are able to understand and reproduce its content completely.

3.2 Processing Graph

In almost all cases a single operator will not be sufficient to produce desired results. Several operators will act sequentially or in parallel on the same or on different data, and an operator may invoke further operators to accomplish its goals. Such a sequence of operator calls may be understood as a directed acyclic graph (DAG) which we call *processing graph*. If each invocation of an operator is realized by a method call, the processing graph is a subgraph of the dynamic call graph of the analysis process. To understand the process this DAG may also be interpreted as a hierarchical graph, where the invocation of an operator is represented as a nested child of the calling operator as shown in Fig. 1 for an example graph. Each invocation of an operator is depicted as a rectangle. There are further nodes in the graph representing the events where new data objects are generated. These are shown as triangles and are denoted as *data ports* in contrast to *input* and *output ports* of operators defined below. Typical examples are reading data from file, cloning data or generating data from scratch.

Graph nodes are connected by edges indicating the flow of data. Each operator features input and output ports each corresponding to one of the input and output data objects. These ports may be conceived as the entry or exit points of data into and out of the operator. Ports are depicted as filled ellipses in light gray (input ports) and dark gray (output ports), respectively.[3]

Besides the graph structure, describing the operators applied and the data flow between them, further required information are the parameters' values upon invocation and the software version of each operator. Continuing the argument given above for a single operator, given this information along with the processing graph allows to completely understand and reproduce the resulting data, given knowledge of the initial input objects to the top most operator. This argument is valid for the final output of the processing pipeline and also for temporary results of processing. In the latter case a subgraph of the complete processing graph contains all relevant information and may easily be extracted.

Obviously it is of interest to extend documentation of analysis beyond the execution of single processes, as often intermediate results are persistently stored, e.g., in files, and further processes perform additional operations on the data. Such documentation across processes becomes feasible if not only the output data itself are stored, but are accompanied by an external representation of the processing graph including parameter values and software version. This allows to retrace the complete processing history of a given data object back to the point

[3] Colour codes for different entities have been converted to b/w in the proceedings.

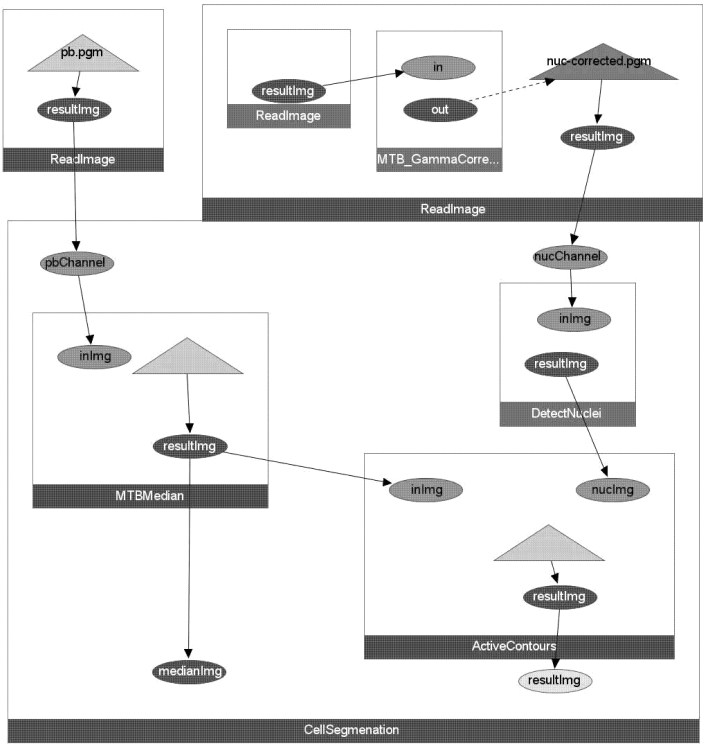


Fig. 1. Example processing graph representing the history for the data object shown as bright ellipse. Each operator call refers to a rectangle, which is in lighter gray if temporarily collapsed. Light and dark gray ellipses are input and output ports of operators, triangles represent newly generated data objects. If read from file, the triangle is tagged by the filename. If in addition a processing graph of a former analysis procedure was read the triangle is coloured in dark gray and both graphs are connected by a dashed edge.

of generation by a sensor and, thus, to reproduce the complete analysis pipeline. In Fig. 1 the events of reading data from persistent memory are depicted as triangles tagged with names showing, e.g., the name of a file read. Data objects internally created from scratch are displayed by a gray triangle without name. If a former processing history is associated with the data imported, its processing graph is read as well and linked to the processing graph of the currently executing process via a dashed edge. In this case the triangle is displayed in dark gray.

3.3 Implementation

We implemented this concept of operators and process documentation in Java, but the concept can be transfered to other object-oriented languages as well. In our implementation each operator has to extend the abstract class `ALDOperator`. This is necessary to automatically generate the processing history upon

```
public class ALDCalcMean extends ALDOperator {
    @Parameter( label= "data", required = true,
        type = Type.INPUT, explanation = "Input data")
    private Double [] data;

    @Parameter( label= "mean",type = Type.OUTPUT,explanation = "Mean value")
    private Double mean = null;

    @Parameter( label="trim", type=Type.PARAMETER,
        explanation = "Fraction of data to be trimmed")
    private float trim = 0.0f;

    protected ALDCalcMean(Double [] _data) throws ALDOperatorException {
                this.data = _data;  }
    protected void operate() { // code your operation here }

     // getter and setter methods come here
}
```

Fig. 2. Code fragment implementing a simple operator

invocation without any efforts for the programmer. Basically two issues have to be taken care of when implementing an operator, namely defining the interface of the operator and implementing the operation or functionality itself.

The interface of an operator comprises the input and output objects and the parameters of the operator. Each of these items is realized by a member variable of the class. The annotation mechanism of Java is used to define

- a name or label,
- the role, e.g., as a parameter or output object,
- whether a parameter or input object is required or optional, and
- an optional explanatory text.

A simple example is shown in Fig 2.

In addition, an operator may use supplemental arguments, e.g., to define variables to control output or debugging information or to return intermediate results. The output of an operator is expected to be independent of the values of these supplemental arguments, hence, they are not stored in the history. These supplemental parameters are not tied to the operator concept in a strict sense, as they could be realized by not annotated fields of a class. But, they are included to facilitate code generation and graphical programming in the future.

The functionality of the operator is supplied by implementing the abstract method `operate()`. Obviously, this method has access to all input and output objects, as well as parameters, as these are realized as member variables.

To actually invoke an operator, an instance of the operator class is created, the input objects and parameters of this object are set, and subsequently the method `runOp()` supplied by `ALDOperator` is called. Upon return from `runOp()` the output objects can be retrieved from the operator instance (see Fig. 3).

```
ALDCalcMean mOp =
 new ALDCalcMean(MyData);
mOp.setTrim( 0.1f);
mOp.runOp();
Double out = mOp.getMean();
```

Fig. 3. Code fragment showing how to invoke an operator

The method `runOp()` realizes the core of the documentation. Upon call it creates an instance of the class `ALDOpNode` which represents this activation of the operator in the processing graph. Each `ALDOpNode` is automatically linked to its parent in the graph, unless it is a top-level `ALDOpNode` having no parent. Thus, the processing graph is incrementally built on-the-fly as operators are invoked. To deliberately hide an operator invocation from the processing history, `runOp(true)` may be used. In addition, `runOp()` links the input ports of all non-null input objects to their current origin in the processing graph. This origin may be an input or data port of the parent `ALDOpNode` or an output port of a sibling `ALDOpNode`. For each data object this originating port is stored in a global weak hash map and updated as the data object is passed to or from further operator calls during subsequent processing. As a consequence only uniquely identifiable objects are allowed as inputs and outputs, which excludes only primitive data types, interned strings, and cached numerical objects.

Supplementing the construction of the graph, upon invocation the method `runOp()` retrieves the current values of all parameters via the Java reflection mechanism and stores these in the `ALDOpNode` created. As the last necessary information the current software version is acquired. To this end `Alida` defines an abstract class `ALDVersionProvider` specifying methods for software version retrieval. Concrete implementations of this class can be passed to the operator mechanism at runtime. A factory infers the desired implementation via environment variables or JVM properties and creates corresponding objects. As default implementation `Alida` supplies the programmer with class `ALDVersionProviderCmdLine` reading version data from the environment. Other options are straightforward, e.g., querying SVN repositories is implemented in `MiToBo`.

3.4 External Representation of the Processing History

To make use of the implicitly constructed processing graph, at any point of processing the subgraph associated to a particular data object may explicitly be extracted. To this end the originating port of the data object is queried from the hash map mentioned above and the processing graph is built by traversing the port links in a bottom-up fashion to collect all relevant `ALDOpNodes` including parameter values and software version. For an external representation we serialize the constructed processing graph to XML using `graphML` [1] as the basic tool. We extended its schema description to satisfy our application specific needs like the representation of properties, and use *xmlbeans*[4] for code generation. Although `graphML` supports the concepts of ports we prefer to model ports as child nodes of an operator node in `graphML` because we need to attach more complex information to ports than provided by `graphML` and as this gives more flexibility regarding the visualization of processing graphs.

[4] `http://xmlbeans.apache.org`

4 Exploring the Processing History: `Chipory`

One important use of the processing history is the *visual* inspection of operators including their sequence of invocation, and to scrutinize the parameter settings.

To support these needs we extended `Chisio`[5] to handle the specific extensions of the documentation framework yielding `Chipory` (**Chi**sio for **P**rocessing hist**ory**). `Chisio` is a free editing and layout tool for compound or hierarchically structured graphs. In `Chipory` most editing and display functionality was conserved, however, is not required for inspecting a processing history. `Chisio` offers several layout algorithms where `Chipory` chooses *Sugiyama* as default as this is most adequate for the hierarchical graph structure of processing histories.

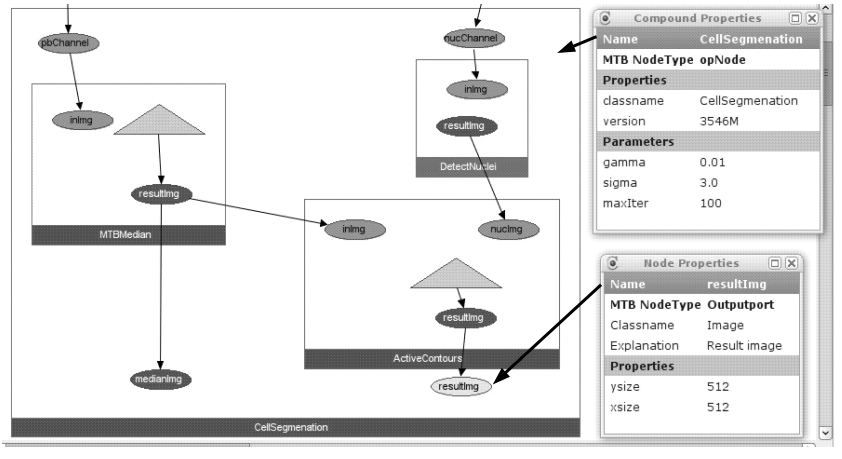


Fig. 4. Screen shot of `Chipory` with details for the operator `CellSegmentation` and the output port `resultImage` of the same operator

The processing graph displayed in Fig. 1 was in fact created using `Chipory`. One important extension of `Chipory` with respect to `Chisio` is its support to *collapse* operator nodes. Collapsing a node makes all enclosed operator and data nodes invisible, thus, only the ports of collapsed operators are shown. In the example there are three collapsed nodes, depicted with their names on lighter gray background, namely `MTB_GammaCorrection`, `DetectNuclei`, and one invocation of `ReadImage`. If a node is uncollapsed later on, all enclosed nodes are made recursively visible again, until a previously collapsed node is encountered.

More details for operators and ports may be inspected using the *Object properties* of `Chipory`'s nodes. These are displayed in a separate window which can be popped up for any selected operator node. Information displayed includes

- name of the operator or port,
- type of the node, e.g., `opNode` for operators,
- for operators the parameter values at time of invocation, their class names and software versions,

[5] `http://sourceforge.net/projects/chisio`

- for input and output ports the Java class of the data object as it is passed into or out of the operator, along with the explanatory text of this port,
- for output ports additionally the properties recorded if the data object resulting from the invocation of the operator is derived from the class `ALDData`.

In Fig. 4 this is shown for the operator `CellSegmentation` and the output port with name `resultImage` of this operator.

5 `Alida` in Practice: The `MiToBo` Image Processing Toolbox

We use `Alida` within our research on automatic analysis of microscope images (see, e.g., [7,4]). In particular, we have developed a software toolbox called `MiToBo`[6] using `Alida` as the core for automatic documentation. `MiToBo` is based on and extends the widely-used Java image analysis framework ImageJ[7]. Within `MiToBo` more than 100 operators are already implemented including basic image processing operations like filtering or thresholding, segmentation employing active contours, and also specific applications intended for the use by life scientists.

The integrated documentation capabilities of `Alida` significantly ease algorithm development in `MiToBo` as well as scientific data preparation for publication. Automatic documentation simplifies parameter tuning in testing phases, releasing the developer from sometimes cumbersome explicit parameter logging. Once an algorithm has reached a stable status and, e.g., its results on specific data are to be published, `Alida` guarantees for complete process logging and easy reproduction of the results without enforcing additional efforts. Note that this is independent of the scientist actually producing the results as no special knowledge is needed to produce the documents.

Besides these benefits another important feature of `Alida` is the larger flexibility it induces for the biologists in their daily work when seeking for suitable software to analyze new kinds of data. Given the documentation of operations and parameters with which earlier results on similar data were produced, existing algorithms can easily be tested by the biologists given the formerly optimized set of parameters as a reasonable starting point. This not only helps biologists to quickly obtain prototypical results, but it also yields valuable information for computer scientists if adaptations of algorithms become necessary.

6 Conclusion

The integrated and automated documentation concept of `Alida` releases programmers as well as users from time-consuming and error prone manual documentation tasks during algorithm development and scientific data analysis. For each generated result data item the complete trace of its processing

[6] **M**icroscope **I**mage Analysis **To**ol**Bo**x, `http://www.informatik.uni-halle.de/mitobo`

[7] `http://rsb.info.nih.gov/ij/index.html`

history is available as well in an easy to interpret graphical representation as in wide-spread XML format. Thus analysis, verification and reconstruction of the analysis process can easily be accomplished even in the course of long-term archival.

While for the moment `Alida` is mainly focused on the documentation task, there are straightforward extensions towards automatic code generation from formerly extracted processing histories. Also foundations for an integration of the process documentation with graphical programming frameworks have been established by well-defined operator interfaces and will be further investigated.

References

1. Brandes, U., Eiglsperger, M., Herman, I., Himsolt, M., Marshall, M.: GraphML progress report structural layer proposal. In: Goodrich, M.T., Kobourov, S.G. (eds.) GD 2002. LNCS, vol. 2528, pp. 109–112. Springer, Heidelberg (2002)
2. Brazma, A., Hingamp, P., Quackenbush, J., Sherlock, G., Spellman, P., et al.: Minimum information about a microarray experiment (MIAME) - toward standards for microarray data. Nature Genetics 29(4), 365–371 (2001)
3. Buchholz, F., Spafford, E.: On the role of file system metadata in digital forensics. Digital Investigation 1(4), 298–309 (2004)
4. Glaß, M., Möller, B., Zirkel, A., Wächter, K., Hüttelmaier, S., Posch, S.: Scratch Assay Analysis with Topology-Preserving Level Sets and Texture Measures. In: Vitrià, J., Sanches, J.M., Hernández, M. (eds.) IbPRIA 2011. LNCS, vol. 6669, pp. 100–108. Springer, Heidelberg (2011)
5. Konstantinides, K., Rasure, J.: The Khoros software development environment for image and signal processing. IEEE Trans. on Image Processing 3(3), 243–252 (1994)
6. Linkert, M., Rueden, C., et al.: Metadata matters: access to image data in the real world. The Journal of Cell Biology 189, 5777–5782 (2010)
7. Möller, B., Stöhr, N., Hüttelmaier, S., Posch, S.: Cascaded segmentation of grained cell tissue with active contour models. In: Proceedings International Conference on Pattern Recognition (ICPR), pp. 1481–1484 (2010)
8. Narayanasamy, S., Pereira, C., Patil, H., Cohn, R., Calder, B.: Automatic logging of operating system effects to guide application-level architecture simulation. In: Proc. of the Joint Int. Conf. on Measurement and Modeling of Computer Systems, SIGMETRICS 2006/Performance 2006, pp. 216–227. ACM, New York (2006)
9. Open Microscopy Environment: OME schemas (2011), `http://www.openmicroscopy.org/Schemas` (accessed March 17, 2011)
10. Pedrinaci, C., Lambert, D., Wetzstein, B., van Lessen, T., Cekov, L., Marin, D.: SENTINEL: a semantic business process monitoring tool. In: Proc. of 1st Int. Workshop on Ontology-supported Business Intelligence, OBI 2008 (2008)
11. Satoh, K., Okumura, A.: Documentation know-how sharing by automatic process tracking. In: Proc. of 4th Int. Conf. on Intelligent User Interfaces, pp. 49–56 (1999)
12. Taylor, C., Paton, N., Lilley, K., et al.: The minimum information about a proteomics experiment (MIAPE). Nat. Biotechnol. 25, 887–893 (2007)
13. Warfield, S., Zou, K., Wells, W.: Validation of image segmentation by estimating rater bias and variance. Phil. Trans. R. Soc. A 366(1874), 2361–2375 (2008)

Efficient Use of Geometric Constraints for Sliding-Window Object Detection in Video

Patrick Sudowe and Bastian Leibe

UMIC Research Centre
RWTH Aachen University, Germany

Abstract. We systematically investigate how geometric constraints can be used for efficient sliding-window object detection. Starting with a general characterization of the space of sliding-window locations that correspond to geometrically valid object detections, we derive a general algorithm for incorporating ground plane constraints directly into the detector computation. Our approach is indifferent to the choice of detection algorithm and can be applied in a wide range of scenarios. In particular, it allows to effortlessly combine multiple different detectors and to automatically compute regions-of-interest for each of them. We demonstrate its potential in a fast *CUDA* implementation of the *HOG* detector and show that our algorithm enables a factor 2-4 speed improvement on top of all other optimizations.

1 Introduction

Object detection has become a standard building block for many higher-level computer vision tasks. Current detectors reach sufficient detection accuracies [1] to support complex mobile scene analysis and multi-person tracking applications [2,3,4], and there is a strong call to make this performance available for automotive and robotics applications.

Even though first CPU [5] and GPU [6,7] implementations of object detectors have already been proposed that operate at several frames per second, the pressure to develop more efficient algorithms does not subside. This is because object detection is only part of a modern vision system's processing pipeline and needs to share computational resources with other modules. In addition, practical applications often require not just detection of a single object category, but of many categories seen from multiple viewpoints [8]. Consequently, efficient object detection is a very active field of research. Many approaches have been proposed in recent years to speed up detection, including detection cascades [5,9,10], efficient approximative feature representations [11,12], and alternatives to the sliding-window search strategy [13].

The use of scene geometry enables computational speedups which are orthogonal to the approaches mentioned above. It is well-known that in many street-scene scenarios, objects of interest can be expected to occur in a corridor of locations and scales on the ground plane [14,3]. Approaches targeted at automotive scenarios have used such constraints for a long time. Surprisingly, though,

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 11–20, 2011.
© Springer-Verlag Berlin Heidelberg 2011

the employed geometric constraints are often defined rather heuristically, sampling a few 3D locations and scales to be processed by the detector [15]. Such an approach is feasible for single-class detection scenarios with limited camera motion, but it quickly becomes impractical when multiple object class detectors shall be combined or when the camera may undergo stronger motion (*e.g.* for automatic sports video analysis or pan/tilt/zoom surveillance cameras).

In this paper, we derive a general solution for this problem that is applicable to *any camera*, *any ground plane*, and *any object detector or combination of detectors* (as long as perspective distortion is small enough such that objects can still be detected upright in the image). Starting from geometric principles, we analyze the space of sliding-window locations that correspond to geometrically valid object detections under the constraints of a ground plane corridor. We show that for a given detector scale, this space corresponds to an image region that is bounded by two parabolas, and we give a practical formula to efficiently compute the corresponding ROI. Based on this, we propose a sliding-window algorithm that touches the minimal set of pixels to return all valid detections.

Our approach is flexible. It does not rely on a precomputed ground plane corridor, but provides a principled algorithm to *recompute the ROI for every frame* based on an estimate of the camera motion. The only information it requires is the current ground plane homography, the projection of the ground plane normal vector (both of which can be obtained either by structure-from-motion [3] or homography tracking [16]), the height of the detection bounding box in the image, and the real-world size range of the objects of interest.

In particular, this makes it possible to combine multiple object detectors with minimum effort. It does not matter whether those detectors have been trained on different resolutions [17] or for different viewpoints or real-world object sizes [18]. Everything that is needed is each detector's bounding box height in the image and the target object's real-world size range. We demonstrate this by performing experiments for single-class pedestrian detection [19] and for multi-viewpoint car detection (similar to [3,18]). In all cases, we show the validity of our approach and quantify the resulting detector speed-ups. In order to perform a fair quantitative evaluation of those speed-ups, it is important to apply our algorithm to an already efficient detector implementation. We therefore combine it with a fast CUDA implementation of HOG, which closely follows the original HOG pipeline from [19]. Our resulting *groundHOG* pedestrian detector runs at 57*fps* for a street scene scenario without loss in detection accuracy.

Related Work. Several recent approaches have been proposed to integrate scene geometry and detection [14,4,20,21,22]. Their main goal is to increase precision by selecting consistent detections. However, scene geometry offers additionally a potential speed increase, if one limits the detector's search region. Many automotive applications therefore employ a fixed, precomputed ground plane corridor (*e.g.* [2]), which is deliberately left a bit wider than necessary in order to compensate for changing camera pitch. Other approaches try to stabilize the camera image by detecting the horizon line [23] or fit a ground plane to stereo measurements [15]. A common approach is to then sample a fixed set of

ROIs in 3D and to process the corresponding image regions with an AdaBoost classifier [15]. Such an approach is possible when dealing with a single object category, but it quickly becomes both inefficient and cumbersome when multiple categories with different real-world sizes shall be detected simultaneously. Such a scenario requires a more principled solution.

From the computational side, there are two major cost items in the design of a sliding-window classifier: the evaluation of the window classifier itself and the computation of the underlying feature representation. The success of the Viola-Jones detector [5] has shown that for certain object classes such as faces or cars, relatively simple Haar wavelet features are sufficient. This has been used in the design of AdaBoost based detectors which evaluate the features for each test window independently. For more complex object categories, Histograms of Oriented Gradients (HOG) [19] have become the dominant feature representation [1]. Unfortunately, HOG features are expensive to compute. Looking at highly optimized CUDA implementations of the HOG detection pipeline [6,7], they typically account for 60-70% of the total run-time of a single-class classifier. It is therefore more efficient to precompute the features and to reuse them for different evaluation windows [24,25], as is common practice in the design of SVM-based detectors [19,18]. When combining several classifiers for different object aspects or categories, the relative importance of the shared feature computation decreases, but it still imposes a lower bound on the effective run-time.

In this paper, we therefore address both problems together: (1) Our proposed algorithm automatically computes the ROIs in which each employed classifier needs to be evaluated for each detection scale, such that it only considers geometrically valid detections. (2) In addition, it returns the minimal set of pixels that need to be touched for all detectors together, so that feature computation can be kept efficient.

The paper is structured as follows. We first derive a general formulation for the problem and analyze the space of sliding-window locations that correspond to geometrically valid object detections (Sec. 2). Based on this analysis, we propose a general algorithm for incorporating ground plane constraints directly into the detector design (Sec. 3). Finally, Sec. 4 presents detailed experimental results evaluating the approach's performance in practice.

2 The Space of Valid Object Detections

The central question we address in this paper is: What is the space of all valid detections? That is, if we only consider detection bounding boxes that correspond to objects on the ground plane whose real-world size is within a range of $S_{obj} \in [S_{min}, S_{max}]$, what is the region in the image in which those bounding boxes can occur? More concretely, we consider a sliding-window detector that processes the image at a discrete set of scale levels. At each scale, a fixed-size bounding box of height s_{img} pixels slides over the image. We are interested in the positions of the bounding box foot point y_b that lead to valid detections.

Geometric Derivation. In the following, we address this problem in the general case. We use the notation from [26], denoting real-world quantities by

upper-case letters and image quantities by lower-case letters. Let us assume that we have a calibrated camera with projection matrix $\mathtt{P} = \mathtt{K}\,[\mathtt{R}|\mathbf{t}]$ watching a scene containing the ground plane $\boldsymbol{\pi}$ with normal vector $\mathbf{N}$(Fig. 1). We can define a local coordinate system on the ground plane by an origin $\mathbf{Q}_0$ and two orthogonal basis vectors $\mathbf{Q}_1, \mathbf{Q}_2$. The (homogenous) world coordinates $\mathbf{X} = [X, Y, Z, 1]^\mathsf{T}$ of a point $\mathbf{U} = [U, V, 1]^\mathsf{T}$ on the ground plane are then given by the transformation

$$\mathbf{X} = \mathtt{Q}\mathbf{U} = \begin{bmatrix} \mathbf{Q}_1 & \mathbf{Q}_2 & \mathbf{Q}_0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{U} \tag{1}$$

and their projection on the image plane is given by the homography $\mathtt{H}_{\boldsymbol{\pi}} = \mathtt{PQ}$.

We now want to find an object with real-world height S_{obj} that is located on or above ground plane position $\mathbf{U}$ and which extends from height S_b to height $S_t = S_b + S_{obj}$. The projections $\mathbf{x} = [x, y, w]^\mathsf{T}$ of the object's bottom and top points $\mathbf{X}_b$ and $\mathbf{X}_t$ into the image are given by

$$\mathbf{x}_b = \mathtt{P}\mathbf{X}_b = \mathtt{P}\left(\mathtt{Q}\mathbf{U} + S_b\mathbf{N}\right) = \mathtt{H}_{\boldsymbol{\pi}}\mathbf{U} + S_b\mathtt{P}\mathbf{N} \tag{2}$$

$$\mathbf{x}_t = \mathtt{P}\mathbf{X}_t = \mathtt{P}\left(\mathtt{Q}\mathbf{U} + S_t\mathbf{N}\right) = \mathtt{H}_{\boldsymbol{\pi}}\mathbf{U} + S_t\mathtt{P}\mathbf{N}. \tag{3}$$

Writing $\mathbf{h}_j^\mathsf{T} = [h_{j1}, h_{j2}, h_{j3}]$ for the row vectors of $\mathtt{H}_{\boldsymbol{\pi}}$ and using $\mathbf{n} = [n_1, n_2, n_3]^\mathsf{T} = \mathtt{P}\mathbf{N}$, we can compute the y coordinates of the corresponding image pixels as

$$y_b = \frac{\mathbf{h}_2^\mathsf{T}\mathbf{U} + S_b n_2}{\mathbf{h}_3^\mathsf{T}\mathbf{U} + S_b n_3}\,, \qquad y_t = \frac{\mathbf{h}_2^\mathsf{T}\mathbf{U} + S_t n_2}{\mathbf{h}_3^\mathsf{T}\mathbf{U} + S_t n_3}\,. \tag{4}$$

We can now express the constraint that the projected object height in the image should exactly correspond to the height of the sliding window given by s_{img}:

$$y_t = y_b + s_{img} \tag{5}$$

$$\frac{\mathbf{h}_2^\mathsf{T}\mathbf{U} + S_t n_2}{\mathbf{h}_3^\mathsf{T}\mathbf{U} + S_t n_3} = \frac{\mathbf{h}_2^\mathsf{T}\mathbf{U} + S_b n_2 + s_{img}\left(\mathbf{h}_3^\mathsf{T}\mathbf{U} + S_b n_3\right)}{\mathbf{h}_3^\mathsf{T}\mathbf{U} + S_b n_3}$$

$$\left(\mathbf{h}_2^\mathsf{T}\mathbf{U}+S_t n_2\right)\left(\mathbf{h}_3^\mathsf{T}\mathbf{U}+S_b n_3\right) = \left(\mathbf{h}_2^\mathsf{T}\mathbf{U}+S_b n_2+s_{img}\left(\mathbf{h}_3^\mathsf{T}\mathbf{U}+S_b n_3\right)\right)\left(\mathbf{h}_3^\mathsf{T}\mathbf{U}+S_t n_3\right)$$

The set of all ground plane locations $\mathbf{U}$ for which this constrained is fulfilled is then given by the conic section $\mathtt{C}$ with

$$\mathbf{U}^\mathsf{T}\mathtt{C}\mathbf{U} = 0 \tag{6}$$

$$\begin{bmatrix} U & V & 1 \end{bmatrix} \left[\mathbf{h}_3\mathbf{h}_3^\mathsf{T} + \begin{bmatrix} 0 & 0 & a \\ 0 & 0 & b \\ a & b & c+d \end{bmatrix} \right] \begin{bmatrix} U \\ V \\ 1 \end{bmatrix} = 0\,, \quad \text{where}$$

$$\begin{bmatrix} 2a & 2b & c \end{bmatrix} = \frac{1}{s_{img}} \left[(S_t - S_b)(n_3\mathbf{h}_2^\mathsf{T} - n_2\mathbf{h}_3^\mathsf{T}) + s_{img}(S_t + S_b)n_3\mathbf{h}_3^\mathsf{T} \right] \tag{7}$$

$$d = S_t S_b n_3^2\,. \tag{8}$$

It can easily be seen that the discriminant of the conic (*i.e.*, the determinant of its upper-left 2×2 matrix) is 0, since $\mathbf{h}_3\mathbf{h}_3^\mathsf{T}$ has only rank 1. The equation therefore represents a parabola, whose projection into the image is given by

$$\mathbf{x}^\mathsf{T}\mathtt{D}\mathbf{x} = \mathbf{x}^\mathsf{T}\mathtt{H}_{\boldsymbol{\pi}}^{-\mathsf{T}}\mathtt{C}\mathtt{H}_{\boldsymbol{\pi}}^{-1}\mathbf{x} = 0\,. \tag{9}$$

Analysis. In our sliding-window detection scenario, we are interested in finding objects which have a real-world height in the range $S_{obj} \in [S_{min}, S_{max}]$. From the above derivation, it follows that the only windows at which those objects can be found are located in the space between the two curves defined by $\mathtt{D}$ for $S_t = S_b + S_{min}$ and $S_t = S_b + S_{max}$. In the following, we analyze the detailed shape of those curves further. If the camera viewing direction is exactly parallel to the ground plane, then eq. (9) degenerates and defines a pair of lines (one of which will be behind the camera). In order to analyze the remaining cases, we perform the variable substitution

$$\begin{bmatrix} \bar{U} \\ \bar{V} \end{bmatrix} = \begin{bmatrix} h_{31} & h_{32} \\ -h_{32} & h_{31} \end{bmatrix} \begin{bmatrix} U \\ V \end{bmatrix} \tag{10}$$

and obtain the ground plane locations of the curve points on the parabola

$$\bar{V} = \frac{h_{31}^2+h_{32}^2}{2(h_{32}a-h_{31}b)}\bar{U}^2 + \frac{h_{33}(h_{31}^2+h_{32}^2)+h_{31}a+h_{32}b}{h_{32}a-h_{31}b}\bar{U} + \frac{(h_{33}^2+c+d)(h_{31}^2+h_{32}^2)}{2(h_{32}a-h_{31}b)} \ .$$

Of particular interest is the factor in front of the quadratic term $\bar{U}^2$. In a more detailed analysis we found that the factor is negligible for most practically relevant cases in automotive or mobile robotics scenarios, unless a wide-angle camera is used. This means the parabola can be approximated by a line.

Obtaining the Ground Constraints. In the above derivation, we assumed an internally and externally calibrated camera, as well as knowledge about the ground plane. In an automotive or mobile robotics setup, this information can be obtained by structure-from-motion and dense stereo measurements (*e.g.* [3,4]). However, looking at the components of $\mathtt{D}$ in eq. (9), it becomes clear that the curve is already fully specified if we know the ground plane homography $\mathtt{H}_{\boldsymbol{\pi}}$ and the projection of the normal vector $\mathbf{n} = \mathtt{P}\mathbf{N}$. This makes the approach also attractive for other applications, such as sports broadcasts or surveillance, where landmark points on the ground plane can be tracked to maintain calibration.

The homography $\mathtt{H}_{\boldsymbol{\pi}}$ can be estimated from at least four image points with known ground plane coordinates (*e.g.* using the DLT algorithm [26]). The projection of the normal can also easily be obtained from two or more points with known heights above the ground plane. Let $\mathbf{X}_i$ be a point with height S_i above its known ground plane footpoint $\mathbf{U}_i$. According to eq. (4), the corresponding image coordinates are given by

$$x_i = \frac{\mathbf{h}_1^\mathsf{T}\mathbf{U}_i + S_i n_1}{\mathbf{h}_3^\mathsf{T}\mathbf{U}_i + S_i n_3} \ , \qquad y_i = \frac{\mathbf{h}_2^\mathsf{T}\mathbf{U}_i + S_i n_2}{\mathbf{h}_3^\mathsf{T}\mathbf{U}_i + S_i n_3} \ . \tag{11}$$

From this, we get an equation system, with two constraints per measured point:

$$\begin{bmatrix} -S_i & 0 & S_i x_i \\ 0 & -S_i & S_i y_i \\ \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} n_1 \\ n_2 \\ n_3 \end{bmatrix} = \begin{bmatrix} (\mathbf{h}_1^\mathsf{T} - x_i\mathbf{h}_3^\mathsf{T})U_i \\ (\mathbf{h}_2^\mathsf{T} - y_i\mathbf{h}_3^\mathsf{T})U_i \\ \vdots \end{bmatrix} \tag{12}$$

$$\mathtt{A}\mathbf{n} = \mathbf{b} \ , \tag{13}$$

resulting in the least-squares solution $\mathbf{n} = \mathtt{A}^\dagger\mathbf{b}$ if at least two points are given.

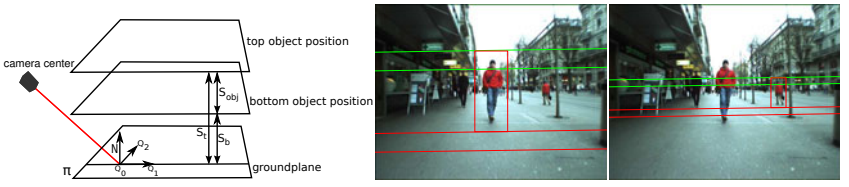


Fig. 1. (left) Visualization of the employed coordinate system and notation. (middle & right) Ground plane corridor at scales $\sigma = 1.75$ (middle) and $\sigma = 0.65$ (right). Two valid detections within the corridor are shown. The selected region-of-interest (ROI) is delimited by the uppermost and lowermost lines.

Extension to Multiple Scales. Until now, we have assumed that the image is scanned with a fixed-size detection window with height s_{img}. In order to detect objects at different scales, a sliding-window detector processes downscaled versions of the input image at fixed scale intervals σ. We achieve the same effect by adapting the internal camera calibration matrix $\mathtt{K}$. If we assume a camera with zero skew, this results in the following matrix for scale level k:

$$\mathtt{K}_k = \begin{bmatrix} \alpha_x/\sigma^k & 0 & x_0/\sigma^k \\ 0 & \alpha_y/\sigma^k & y_0/\sigma^k \\ 0 & 0 & 1 \end{bmatrix} . \tag{14}$$

Propagating this change to the ground plane homography and the projection of the normal vector, we can see that those entities are obtained as

$$\mathtt{H}_{\boldsymbol{\pi},k} = \begin{bmatrix} \mathbf{h}_1^\mathsf{T}/\sigma^k \\ \mathbf{h}_2^\mathsf{T}/\sigma^k \\ \mathbf{h}_3^\mathsf{T} \end{bmatrix} , \qquad \mathbf{n}_k = \begin{bmatrix} n_1/\sigma^k \\ n_2/\sigma^k \\ n_3 \end{bmatrix} . \tag{15}$$

3 Detection Algorithms

Putting all the pieces together, we can now formulate a general algorithm for geometrically constrained object detection, as shown in Alg. 1. For each scale level, we first compute the corresponding $\mathtt{D}$ matrices for the minimum and maximum object size. We then create a rectangular ROI by inserting the x coordinates of the left and right image borders into eq. (9) and taking the minimum and maximum of the resulting y coordinates. As derived above, only the window locations inside this region correspond to geometrically valid object detections. Since we compute the region for each scale independently, this allows us to *restrict all rescaling and feature computation steps* to those regions.

Multi-Class/Multi-viewpoint Detection. A straightforward extension to multiple classes or viewpoints of objects is to apply several specialized classifiers on the precomputed features. This approach can be easily augmented with our geometric constraints formulation. For each individual classifier one precomputes the *ROI*. The *HOG* features are then computed for a minimal region encompassing all *ROIs* that are active at each scale. Each classifier can then be evaluated

Algorithm 1. The proposed algorithm

Compute $\mathtt{H}_{\boldsymbol{\pi}}$ and $\mathbf{n}$.
for all scale levels k **do**
  Compute $\mathtt{H}_{\boldsymbol{\pi},k}$ and $\mathbf{n}_k$ according to eq. (15).
  Compute $\mathtt{D}_{S_{min}}$ and $\mathtt{D}_{S_{max}}$ according to eq. (9) using $\mathtt{H}_{\boldsymbol{\pi},k}$.
  Set x_{min} and x_{max} to the left and right image borders.
  Compute y_{min} and y_{max} by solving eq. (9) for x_{min} and x_{max} using $\mathtt{D}_{S_{min}}$ and $\mathtt{D}_{S_{max}}$.
  Process the ROI $(x_{min}, y_{min}, x_{max}, y_{max})$ with the detector:
  • *Only* up-/downscale the image pixels inside the ROI.
  • *Only* compute features inside the ROI.
  • *Only* apply the sliding-window classifier to the window locations in the ROI.
end for

on its respective region. Note that only the height of each object class and the classifiers' window sizes are necessary. No error-prone manual process is required.

Different Detector Resolutions and Part-Based Models. A common method to improve detection performance is to use specialized classifiers for distinct scale ranges [10]. Our formulation naturally adapts to the *ROI* multi-resolution case. Here, the benefit is that the system can determine automatically if at some scale only a subset of classifiers can return viable detections. It is *not necessary* to fine tune any further parameters. If no valid *ROI* is found for a classifier, it is automatically skipped for the current scale. Similarly, our algorithm can be used with the popular part-based detection approach by Felzenszwalb *et al.* [18].

4 Experimental Results

We quantitatively evaluate our proposed ground plane constraints. In order to demonstrate the advantage our algorithm can achieve *on top of all other optimizations*, we combine it with a highly optimized *CUDA* implementation of the *HOG* detector (in the following called *cudaHOG*). Our code is publicy available at `http://www.mmp.rwth-aachen.de/projects/groundhog`.

Baseline Detection Performance. First, we establish that our baseline system *cudaHOG* achieves the same detection performance as two other published HOG-based systems: the original Dalal HOG Detector [19] and fastHOG [7]. Fig. 3(left) compares the performance on the INRIA pedestrian dataset [19]. We plot recall *vs.* false positives per image (fppi) using the standard PASCAL VOC criterion [27]. The plot shows that our baseline *cudaHOG* implementation is competitive.

Effect of Ground Plane Constraints. Next, we investigate the effects of the ground plane constraints in detail. For the experiments in this section, we use the Bahnhof sequence from the Zurich Mobile Pedestrian corpus [4], and employ ground planes estimated by SfM. The sequence consists of 999 frames of size 640×480, containing 5,193 annotated pedestrians with a height > 60 pixels.

We start by evaluating computational effort on the first 100 frames. We vary the start scale and ground plane corridor size and report the number of blocks and SVM windows evaluated, as well as the average run-time per frame (Tab. 1). Our baseline *cudaHOG* runs at roughly 22 fps for the start scale 1.0. By adopting

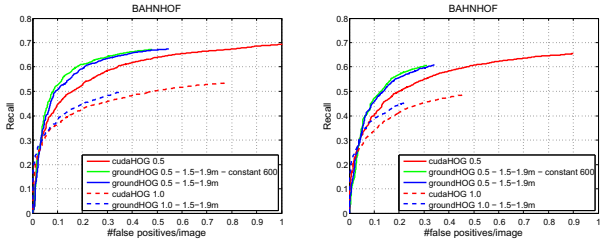


Fig. 2. *cudaHOG vs. groundHOG* for scale steps 1.05 (left) and 1.20 (right). In both cases, we plot the performances when starting at scale 1.0 and when upscaling the image to twice its original resolution (scale 0.5). For the upscaled version we also plot the performance for a bounded ROI width of maximal 600 pixels.

a ground plane corridor of $[1.5m, 1.9m]$, we can more than double the speed to 57 fps.

In addition, we investigate how detection performance is affected by the start scale. As observed by several authors [1,3,4], the HOG detection performance can be considerably improved by upscaling the input images to twice their original resolution (start scale $\sigma = 0.5$ instead of $\sigma = 1.0$). The results shown in Fig. 2 verify this performance improvement (*e.g.*, recall increases by 10% at 0.2 fppi). Usually, the upscaling step comes at considerable additional cost. *groundHOG* can achieve significant computational savings here, since it can limit the upscaling operation to a (relatively small) band around the horizon line. As Tab. 1 shows, *groundHOG* can still process the upscaled images at 20 fps (23 fps if the width of the detection corridor is also bounded to 600 pixels), effectively the same run-time as the unconstrained detector on the original images. Hence, our algorithm achieves *significantly higher recall* in the *same computation time*.

Finally, we investigate the effect of increasing the scale step factor σ from its default value of 1.05 to 1.20 (as also explored in [6]). As shown in Tab. 1 and Fig. 2, this results in a significant speedup to 222 fps without and 87 fps/104 fps with upscaling at a moderate loss of recall (about 5% at 0.5 fppi).

Multi-Class/Multi-viewpoint Detection. As a proof-of-concept experiment for multi-viewpoint detection, we have trained a basic car detector for five viewpoints. We perform a bounding box based non-maximum-suppression step on the individual detector outputs to combine them into a single detection response. While this basic setup cannot achieve the absolute detection rates of more sophisticated setups, it is suitable to demonstrate the effects of a ground plane constraint. We evaluate on the LEUVEN sequence [3] that contains 1175 images at 720×576 pixel resolution. Fig. 3 shows that detection performance benefits significantly, as fewer false positives are encountered by *groundHOG*. When incorporating the ground plane constraints, detection takes only 94 ms compared to originally 339 ms, representing a 3.6-fold speedup.

Table 1. HOG blocks & SVM windows evaluated per frame on the BAHNHOF sequence when applying *groundHOG* with the corridor $[S_{min}, S_{max}] = [1.5m, 1.9m]$. max w refers to a maximal ROI width of 600 pixels. (CPU: Core2Quad Q9550, GPU: GTX 280).

	scale step 1.05					scale step 1.2				
	start 1.0		start 0.5			start 1.0		start 0.5		
	cuda	ground	cuda	ground	max w	cuda	ground	cuda	ground	max w
HOG blocks	53,714	21,668	215,166	52,230	40,781	16,305	6,334	65,404	15,142	11,460
SVM windows	31,312	4,069	162,318	11,132	8,208	9,801	1,243	50,110	3,289	2,341
run-time (ms)	43.78	17.28	183.60	49.80	43.49	12.44	4.50	50.35	11.45	9.58
run-time (fps)	22	57	5	20	23	80	222	19	87	104

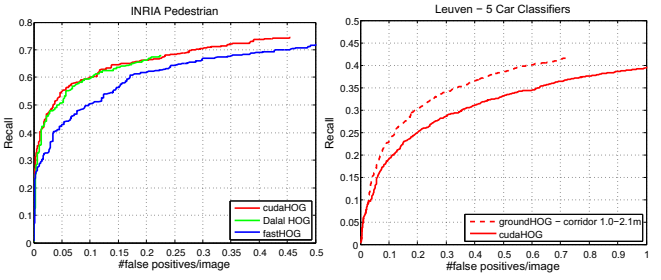


Fig. 3. (left) Baseline comparison on *INRIAPerson* dataset. (right) Results of a 5-view car detector on LEUVEN sequence, demonstrating the performance gains through our geometric constraints. The individual results are merged by a simple NMS scheme.

5 Conclusion

We have systematically explored how geometric ground plane constraints can be used to speed up sliding-window object detection. As a result of this analysis, we have presented a general algorithm that enforces a detection corridor, while taking maximum advantage of the sliding window detection scheme. We have demonstrated this approach in a CUDA implementation of the *HOG* detector. As verified in our experiments, the resulting *groundHOG* algorithm achieves at least the same detection accuracy as the original *HOG* detector in a range of detection scenarios, while allowing significant speedups.

Acknowledgments. This project has been funded, in parts, by the EU project EUROPA (ICT-2008-231888) and the cluster of excellence UMIC (DFG EXC 89).

References

1. Dollar, P., Wojek, C., Schiele, B., Perona, P.: Pedestrian Detection: A Benchmark. In: CVPR (2009)
2. Gavrila, D., Munder, S.: Multi-Cue Pedestrian Detection and Tracking from a Moving Vehicle. IJCV 73(1), 41–59 (2007)
3. Leibe, B., Schindler, K., Van Gool, L.: Coupled Object Detection and Tracking from Static Cameras and Moving Vehicles. PAMI 30(10), 1683–1698 (2008)

4. Ess, A., Leibe, B., Schindler, K., Van Gool, L.: Robust Multi-Person Tracking from a Mobile Platform. PAMI 31(10), 1831–1846 (2009)
5. Viola, P., Jones, M.: Robust Real-Time Face Detection. IJCV 57(2) (2004)
6. Wojek, C., Dorkó, G., Schulz, A., Schiele, B.: Sliding-windows for rapid object class localization: A parallel technique. In: Rigoll, G. (ed.) DAGM 2008. LNCS, vol. 5096, pp. 71–81. Springer, Heidelberg (2008)
7. Prisacariu, V., Reid, I.: fastHOG – a Real-Time GPU Implementation of HOG. Technical Report 2310/09, Dept. of Eng. Sc., Univ. of Oxford (2009)
8. Torralba, A., Murphy, K., Freeman, W.: Sharing Features: Efficient Boosting Procedures for Multiclass Object Detection. In: CVPR (2004)
9. Vedaldi, A., Gulshan, V., Varma, M., Zisserman, A.: Multiple kernels for object detection. In: ICCV (2009)
10. Felzenszwalb, P., Girshick, R., McAllester, D.: Cascade Object Detection with Deformable Part Models. In: CVPR (2010)
11. Dollar, P., Tu, Z., Perona, P., Belongie, S.: Integral Channel Features. In: BMVC (2009)
12. Dollar, P., Belongie, S., Perona, P.: The Fastest Pedestrian Detector in the West. In: BMVC (2010)
13. Lampert, C., Blaschko, M., Hofmann, T.: Efficient Subwindow Search: A Branch and Bound Framework for Object Localization. PAMI 31(12), 2129–2142 (2009)
14. Hoiem, D., Efros, A., Hebert, M.: Putting Objects Into Perspective. In: CVPR (2006)
15. Geronimo, D., Sappa, A., Ponsa, D., Lopez, A.: 2D-3D-based On-Board Pedestrian Detection System. CVIU 114(5), 583–595 (2010)
16. Choi, W., Savarese, S.: Multiple target tracking in world coordinate with single, minimally calibrated camera. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 553–567. Springer, Heidelberg (2010)
17. Park, D., Ramanan, D., Fowlkes, C.: Multiresolution models for object detection. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 241–254. Springer, Heidelberg (2010)
18. Felzenszwalb, P., McAllester, D., Ramanan, D.: A Discriminatively Trained, Multiscale, Deformable Part Model. In: CVPR (2008)
19. Dalal, N., Triggs, B.: Histograms of Oriented Gradients for Human Detection. In: CVPR (2005)
20. Breitenstein, M., Sommerlade, E., Leibe, B., Van Gool, L., Reid, I.: Probabilistic Parameter Selection for Learning Scene Structure from Video. In: BMVC (2008)
21. Bao, Y., Sun, M., Savarese, S.: Toward Coherent Object Detection And Scene Layout Understanding. In: CVPR (2010)
22. Sun, M., Bao, Y., Savarese, S.: Object Detection with Geometrical Context Feedback Loop. In: BMVC (2010)
23. Bombini, L., Cerri, P., Grisleri, P., Scaffardi, S., Zani, P.: An Evaluation of Monocular Image Stabilization Algorithms for Automotive Applications. Intel. Transp. Syst (2006)
24. Schneiderman, H.: Feature-Centric Evaluation for Efficient Cascaded Object Detection. In: CVPR (2004)
25. Lehmann, A., Leibe, B., Van Gool, L.: Feature-Centric Efficient Subwindow Search. In: ICCV (2009)
26. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision. Cambridge University Press, Cambridge (2000)
27. Everingham, M., et al. (34 authors): The 2005 PASCAL Visual Object Classes Challenge. In: Quiñonero-Candela, J., Dagan, I., Magnini, B., d'Alché-Buc, F. (eds.) MLCW 2005. LNCS (LNAI), vol. 3944, pp. 117–176. Springer, Heidelberg (2006)

A Method for Asteroids 3D Surface Reconstruction from Close Approach Distances

Luca Baglivo[1], Alessio Del Bue[2], Massimo Lunardelli[1], Francesco Setti[1], Vittorio Murino[2,3], and Mariolino De Cecco[1]

[1] Department of Mechanical and Structural Engineering, University of Trento
via Mesiano 77, 38123 Trento, Italy
{luca.baglivo,mariolino.dececco,lunardem}@ing.unitn.it
[2] Istituto Italiano di Tecnologia (IIT)
Via Morego, 30 16163 Genova, Italy
{alessio.delbue,vittorio.murino}@iit.it
[3] Department of Computer Science, University of Verona
Strada Le Grazie 15, 37134 Verona, Italy

Abstract. We present a procedure for asteroids 3D surface reconstruction from images for close approach distances. Different from other 3D reconstruction scenario from spacecraft images, the closer flyby gave the chance to revolve around the asteroid shape and thus acquiring images from different viewpoints with a higher baseline. The chance to have more information of the asteroids surface is however paid by the loss of correspondences between images given the larger baseline. In this paper we present a procedure used to reconstruct the 3D surface of the asteroid 21 Lutetia encountered by Rosetta spacecraft on July the 10^{th} of 2010 at the closest approach distance of 3170 Km. It was possible to reconstruct a wider surface even dealing with strong ratio of missing data in the measurements. Results show the reconstructed 3D surface of the asteroid as a sparse 3D mesh.

Keywords: Astronomy, Structure from Motion, Asteroid 3D reconstruction.

1 Introduction

The inference of the physical properties of asteroids (size, shape, spin, mass, density, etc.) is of primary importance for understanding the planetary formation processes of the Solar System. Although many analysis can be carried out on ground stations, with large telescopes and adaptive optics, the best accuracy of parameters, other than the validation of on-ground techniques [3], are obtained from spacecrafts' imaging systems. A fair example is the main belt asteroid 21 Lutetia, the largest to have been visited by a spacecraft.

Rosetta ESA cornerstone mission, two years after the flyby with the asteroid 2867 Steins, encountered Lutetia on its way to the comet 67P/Churyumov-Gerasimenko with an expected rendez vous in 2014. Rosetta flew by Lutetia on the 10^{th} July of 2010 at a 15 km/s velocity. Within about 10 hours, the two OSIRIS (Optical, Spectroscopic, and Infrared Remote Imaging System) imaging cameras [9] took 462 images. The 234 images from Narrow-Angle Camera (NAC, 717 mm focal length, angular resolution 5

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 21–30, 2011.
© Springer-Verlag Berlin Heidelberg 2011

arcsec/px,) and the 228 images from Wide-Angle Camera (WAC, 135 mm focal length, angular resolution 22 arcsec/px) covered more than 50% of the asteroid surface. The highest spatial scale in the NAC was around 60 m/px whereas the whole 120 km maximum length of the asteroid was imaged by the 2048x2048 pixel CCD. All 24 filters of the two cameras, extending from 240 to 980 nm, have been used to investigate on shape, volume, spin, surface characteristics, and a number of other aspects.

The shape reconstruction goes surprisingly beyond its intrinsic scientific importance: an accurate shape model can improve the quality of the spacecraft orbit, geometrical calibration of the cameras, albedo map (by subtracting the illumination due to the shape and to the reflectance model), and hopefully proving gravitational waves [10]. Volume estimation is an important product of the shape reconstruction. The volume has to be obtained by fusing the estimate of the 50% surface visible from OSIRIS with the occluded one which is based on a lightcurve inversion technique (both from ground [2,3] and from previous observations during Rosetta cruise [12]).

This work reports on the authors' first results on the 3D reconstruction of the Lutetia side visible from the OSIRIS imagers. This 3D reconstruction is obtained using a pipeline for structure from motion tailored to the imaging scenario in deep space. The approach is divided in several stages. Starting from Section 2, we describe the feature detection stage and how to form image trajectories lying onto the asteroid surface by rejecting outliers and points at the asteroid limb. The outlier rejection stage is also improved with the known mission data of the prevailing approaching movement of the asteroid. Such filtered image trajectories may yet contain missing data because of occlusions and feature detection failures. Thus, in Section 3 incomplete trajectories are fed to a robust Structure from Motion algorithm that can deal with large amounts of missing data. The experimental Section 4 shows the final reconstructed surface of the asteroid as a 3D mesh.

2 Feature Detection

Our pipeline for 3D reconstruction begins from a sparse reconstruction of uncalibrated images. The first crucial task is to obtain an accurate images features detection and matching over multiple frames. The frames selected were 9 among those obtained by the NAC during the closest approach phase. During the flyby, using a dedicated navigation camera, the attitude control was pointing the spacecraft imagers towards the centroid of the asteroid. The first set of selected images for the reconstruction are a compromise between a high spatial resolution and a stable set of features, considering that in the last phase of the closest approach images had the highest resolution but also a very high rate of occlusions due to the fast relative rotation of the asteroid. We use a multi-step procedure. First, we use the Hessian-based, rotation and scale invariant feature detection algorithm SURF [1] (see Figure 1). The features too close to the limb of the asteroid (Figure 1) are filtered out, thus preventing likely outliers due to the limb variation with the asteroid rotation.

A first matching step is the one proposed by the classical SURF algorithm. In addition, we propose a custom checking phase that filters out the features that do not have a one-to-one match after comparing the matches between two consecutive frames in both

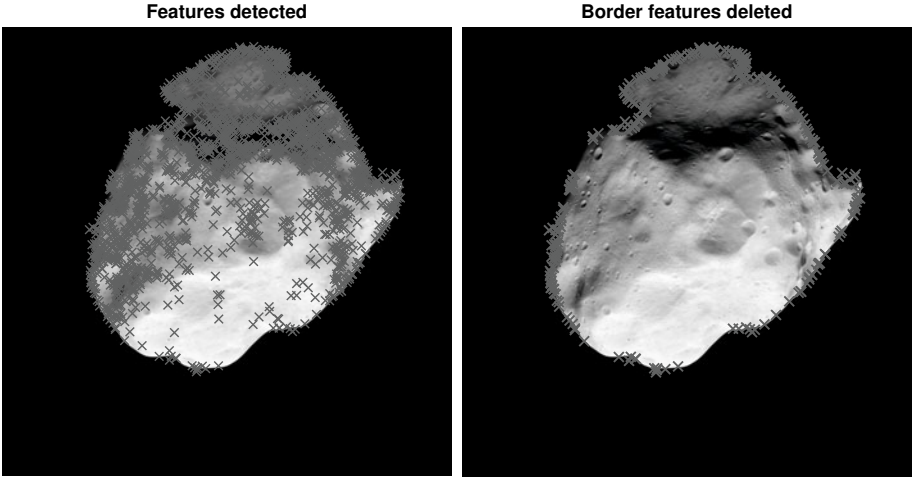


Fig. 1. The left figure shows in red all the features extracted using the SURF features detector. The right figure shows the features detected as lying on the limb of the asteroid. Such features are rejected since likely to be occluded given the approach motion.

forth and back directions. The matched points are then subjected to a RANSAC outliers detection algorithm. RANSAC is a RANdom SAmple Consensus algorithm that rejects samples which do not fit the chosen observation model [7]. In our case, the natural choice given the imaging device is the fundamental matrix model to reject outliers [11]. According to the camera model used in the first reconstruction, as explained in the following sections, an affine fundamental matrix is implemented in the RANSAC fit. As a result, we obtain a set of 2D image trajectories. Short trajectories (i.e. the features tracked in a number of frames under a fixed threshold) are discarded and the remaining outliers are eliminated relying upon a heuristic algorithm which is based upon a trajectory consistency constraint. This final stage filters out the last outliers which are not consistent with the prevailing approaching movement of the asteroid. The relative 3D positions and rotations between object and camera has the affect to create new 2D image trajectories as result of an object rotation around an apparent spin axis. This axis is parallel to the image columns (since spacecraft and asteroid trajectoris are almost in the same plane during the flyby) and far outside the object. The trajectories of the features are well described by a simple linear model where the displacement of a feature along the columnns is proportional to the column distance from the apparent spin axis (check Figure 2).

This multi-step approach reveals to be robust enough to proceed with the surface reconstruction. The features are then stored in a $\mathtt{W}$ matrix which contains the image coordinates as:

$$\mathtt{W} = \begin{bmatrix} \mathbf{w}_{11} & \dots & \mathbf{w}_{1p} \\ \vdots & \ddots & \vdots \\ \mathbf{w}_{f1} & \dots & \mathbf{w}_{fp} \end{bmatrix} = \begin{bmatrix} \mathtt{W}_1 \\ \vdots \\ \mathtt{W}_f \end{bmatrix}, \tag{1}$$

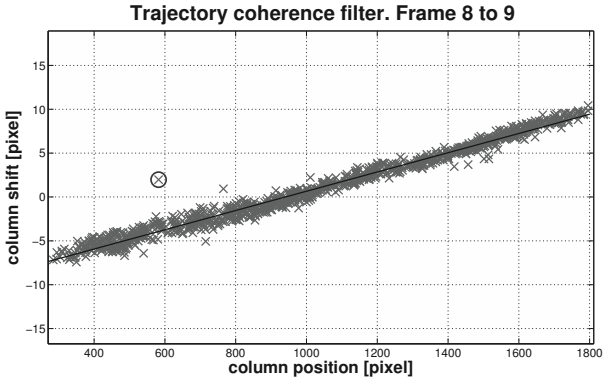


Fig. 2. Features column displacement versus column position is well linearly fitted for this particular frames selection representing Lutetia zooming in and rotating around an apparent spin axis outside the asteroid while it is approaching Rosetta. Crosses are inliers and circles outliers.

where the 2-vector $\mathbf{w}_i j$ represents the image coordinates for frame i and point j. The sub-block matrix $\mathtt{W}_i$ of size $2 \times p$ contains all the coordinates for a single frame. Notice that by forming the matrix we will have some image trajectories that are not viewed in all the f frames. Given the rotational motion of the asteroid, some feature points will appear or disappear during the matching. Thus, the matrix $\mathtt{W}$ will have some missing entries which must be taken into account during 3D reconstruction.

We aim also at estimating the 3D final uncertainty of the reconstruction. A good starting point is to estimate the feature covariance as the observations uncertainty and to propagate it over the reconstruction process by embedding the extrinsic and intrinsic parameters uncertainties [5]. We use the method provided by Zeisl et al. [17] to calculate the correlated uncertainty of SURF feature position estimation. As Zeisl et al. demonstrate experimentally, the directional uncertainty provides a more accurate result, instead of using an isotropic uncertainty, when a weighted 3D reconstruction optimization process like Bundle Adjustment [16] is used. Furthermore, the uncertainty of feature locations could be exploited in the matching phase to select better reference features. In Figure 3 we report a sample of the estimated covariances for some points lying on the asteroid surface.

3 Surface Reconstruction

The feature detection algorithm provides a set of image trajectories of the moving body. The next task is to find the precise localisation of the 3D points lying on the surface. The Structure from Motion (SfM) framework is used to extract the 3D point position given the image coordinates motion. Rigid structure from motion techniques are fairly consolidated in the computer vision literature. With a reliable set of 2D features tracked through time, Bundle Adjustment (BA) is the *Gold Standard* method for 3D reconstruction. However, it has been observed that other methods may perform better than

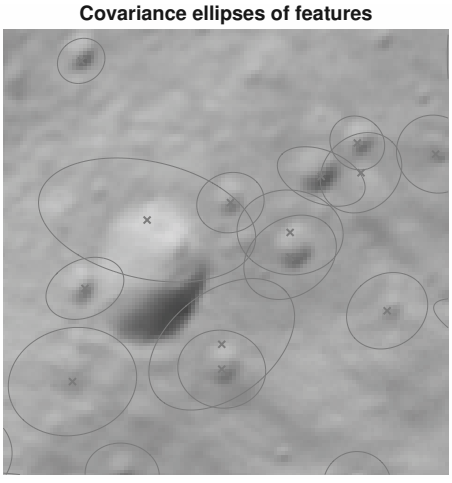


Fig. 3. An example of estimated covariance of the SURF features' location obtained with the covariance estimator in [17]

standard approaches [13] when the amount of missing entries in the measurements increases. Given the amount of missing data and the possible approximation of the camera model, the Lutetia 3D reconstruction task can be tackled with such approach.

Given the fly-by motion of the spacecraft revolving around the asteroid, we have a particular distribution of the computed 2D points from the previous stage. The revolution around the body gives a set of interrupted point trajectories since the surface of the asteroid becomes occluded given the motion. This mostly appears near the border of the imaged shape. This percentage of missing points amounts to about 50% of the overall trajectories for the Lutetia experiment. In general, such percentage of missing features can cause problems to reconstruction algorithms. However, if we approximate the camera model to affine, we have at disposal modern optimisation tolls that are able to stand high percentages of missing data.

3.1 Affine Camera Approximation

The crux of using full perspective camera models are the non-linearities implicit in the optimisation problem solving for the Structure and Motion of the imaged object. However, in a spacecraft flight approach the assumption of a full perspective models is exact but not entirely necessary to provide an initial reconstruction. The object shape relief would always be smaller of some order of magnitude compared to the spacecraft distance from the body. Thus, an affine approximation of the camera model may hold in practice. Experimentally, we have verified for the Lutetia 3D reconstruction that the reprojection error for BA [16] and a state of the art algorithm with affine cameras [6] were very similar. Another advantage of using affine cameras is that such approaches may deal with vast amounts of missing data in the measurements (higher then 60% and up to 90% of the whole measurements). Also, in general, initialising an optimisation

algorithm using affine cameras is far easier then the projective counter-part. Notice that it is rather common using an incremental procedure to SfM where each increment represents the use of a more complex camera model as in [4].

3.2 Affine Structure from Motion

Given the image trajectory stored in a matrix $\mathtt{W}$ of size $2f$ x p as in Eq. (1) we have that the image coordinates are given at each frame i by:

$$\mathtt{W}_i = \mathtt{M}_i \ \mathtt{S} \tag{2}$$

where $\mathtt{M}_i$ is a 2×4 affine camera matrix that projects the overall 3D shape defined by the $4 \times p$ matrix $\mathtt{S}$ in homogeneous coordinates. The chosen camera model for $\mathtt{M}_i$ is the scaled orthographic camera model giving:

$$\mathtt{M}_i = \left[s_i \mathtt{R}_i \middle| \mathbf{t}_i \right] \qquad \mathtt{R}_i \mathtt{R}_i^T = \mathtt{I}_2 \tag{3}$$

with s_i being a scalar which model the scale of the shape given the distance, $\mathtt{R}_i$ a 2×3 camera with two orthonormal rows (i.e. a rotation matrix without the last row) and $\mathbf{t}_i$ a 2D translation vector. The matrix $\mathtt{I}_2$ is a 2×2 identity matrix. Such model represents a reasonable alternative to full-perspective models when the distance from the object is high in comparison with the relative depth of the shape – a fitting constrain for space image analysis. If we stack Eq. (3) frame-wise we can form a global formulation for the imaging projection equations giving:

$$\mathtt{W} = \begin{bmatrix} \mathtt{M}_1 \\ \vdots \\ \mathtt{M}_f \end{bmatrix} \mathtt{S} = \begin{bmatrix} \left[s_1 \mathtt{R}_1 \middle| \mathbf{t}_1 \right] \\ \vdots \\ \left[s_f \mathtt{R}_f \middle| \mathbf{t}_f \right] \end{bmatrix} \mathtt{S} = \mathtt{M} \ \mathtt{S}. \tag{4}$$

It is straightforward to notice that the image coordinates are in bilinear form with the matrix $\mathtt{S}$ storing the 3D coordinates in homogeneous coordinates and each camera projection matrix stacked in $\mathtt{M}$. Moreover, matrix $\mathtt{M}$ contains non-linear constraints given by the orthonormal rows of $\mathtt{R}_i$ i.e. the matrix $\mathtt{M}$ lies on a specific matrix manifold. Such manifold in the case of structure from motion problems is called *motion manifold* [13].

3.3 Optimisation with Augmented Lagrange Multipliers (ALM)

The optimisation structure of bilinear problems with matrix manifold constraints can be optimised with a general tool called BALM recently introduced in [6]. This algorithm is favored in respect to previous approaches such as [13] because the smoother convergence properties given by the ALM. The cost function optimised by BALM for our reconstruction problem is the following:

$$\min_{\mathtt{R}_i \mathtt{R}_i^T = \mathtt{I}_2} \ \left\| \mathtt{D} \odot (\mathtt{W} - \mathtt{M} \ \mathtt{S}) \right\|^2 \tag{5}$$

where $\mathtt{D}$ is a $2f \times p$ mask matrix with either 1 if the 2D point is present or 0 if it is missing (the symbol $\odot$ denotes the element-wise Hadamard product). Given the missing

entries, it is not possible to solve for such cost function in closed form. The algorithm is generic in its formulation but it can achieve state of the art results for several bilinear optimisation problems [6]. Given our Structure from Motion problem, the algorithm requires an initialisation (i.e. the values for $\mathtt{M}^0$ and $\mathtt{S}^0$ at iteration 0) and a projector, preferably optimal, to the respective *motion manifold* of orthographic camera matrices.

Initialisation. To initialise the matrices $\mathtt{M}^0$ and $\mathtt{S}^0$ we first fill each missing entries for each trajectory with the respective mean value of the trajectory. Computationally it means to replace the $\mathtt{W}$ matrix with missing entries with a complete matrix $\mathtt{W}^0$. Since the matrix is now complete, it is possible to run standard Tomasi and Kanade factorization algorithm for rigid Structure from Motion [15]. The closed form solution can find an initial decomposition that complies with Eq. (4).

Projector. The BALM algorithm optimises iteratively the bilinear components $\mathtt{M}$ and $\mathtt{S}$ while enforcing the specific manifold constraints of the problem (check [6] for more details). In the proposed ALM framework, enforcing the manifold constraints signifies projecting the current estimate of $\mathtt{M}$ without constraints and without considering the translation component $\mathbf{t}_i$, called $\tilde{\mathtt{M}}$. This results in a simplified optimisation sub-problem at each frame as:

$$\min_{\mathtt{R}_i\mathtt{R}_i^T=\mathtt{I}_2} \left\|\tilde{\mathtt{M}}_i - s_i\mathtt{R}_i\right\|^2 \tag{6}$$

where $\tilde{\mathtt{M}}_i$ are the sub-block of $\tilde{\mathtt{M}}$ as presented in Eq. (3) for $\mathtt{M}$. This projection is solved in closed form as:

$$\mathtt{R}_i = \mathtt{U}\mathtt{V}^T \quad \text{and} \quad s_i = (\sigma_1 + \sigma_2)\backslash 2 \tag{7}$$

where $\tilde{\mathtt{M}}_i = \mathtt{U}\mathtt{C}\mathtt{V}^T$ is the SVD of the matrix not satisfying the constraints and σ_d for $d = 1, 2$ are the singular values stored in $\mathtt{C}$.

4 Results

Figure 4 and Figure 5 show the results (in texture mapping form) of the ALM algorithm application on part of the visible surface. The underlying sparse mesh is made by 1245 points. The RMS reprojection error is 0.17 pixels thus achieving a low error even using an approximated scaled orthographic camera model. The 3D reconstruction allows to appreciate the 3D relief of the asteroid body. By inspecting the side view in Figure 5, it is possible to notice the various elevations of the 3D shape.

The large and completely shaded areas, mostly in crater regions, have not been reconstructed in details as well as the large homogeneous areas because of the lack of feature points in these regions. We plan to use photometric, shading and dense methods techniques in order to increase the reconstruction accuracy in these regions. Similarly to the sparse 3D reconstruction case, such methods will be tailored to the spacecraft data and the albedo information provided by the mission.

The Lutetia reconstruction of its part visible from the Osiris imagers, which is about 50% of the total surface can not be cross-validated with other measurements except for the silhouette part, which should be consistent with that of the occluded part of the asteroid reconstructed thanks to a lightcurve inversion technique [2, 3].

Fig. 4. Asteroid texture mapping – frontal view

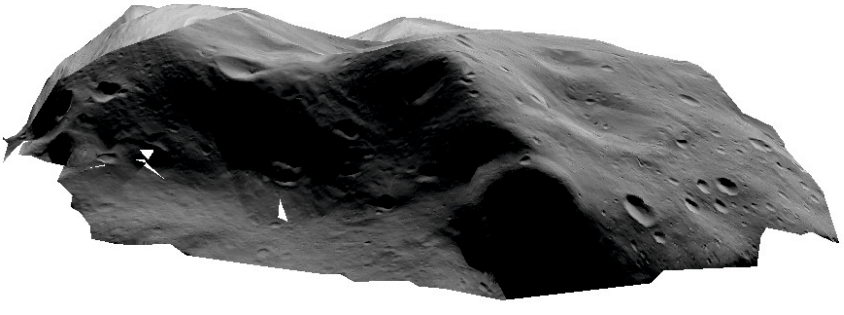


Fig. 5. Asteroid texture mapping – side view

5 Conclusions and Future Work

We presented a 3D reconstruction pipeline for asteroids shape reconstruction from spacecrafts flyby images. In particular we obtained our first results on the Lutetia asteroid using the images taken from OSIRIS data servers. The ALM optimization approach applied on the affine structure from motion problem with a large fraction of missing data as in this case, confirmed to be fairly good in providing an accurate 3D reconstruction. Such initial 3D shape reconstruction can be aligned to the real observation parameters in order to estimate the exact scale and to fix the reference system to absolute astronomical coordinates. In such regard, we may use Schweighofer and Pinz algorithm [14] to align 3D reconstruction with Rosetta spacecraft data. After that, a final BA algorithm can refine the results using the full parameters of the imaging model. At this stage we

will implement the covariance propagation starting by the feature covariance and using a standard analytical method [8].

Future work will be directed toward the dense and precise reconstruction of largely shaded areas. In a similar manner as for the sparse 3D reconstruction case, we aim to include in the pipeline robust photometric techniques which may also account for missing entries given the asteroid rotation and relief self-occlusion. This stage can be improved by making an explicit use of the a priori information given by the surface albedo and illumination sources localisation. In this way, we plan to create a system that consistently joins photometric information to the proposed 3D reconstruction pipeline.

Acknowledgement. OSIRIS was built by a consortium led by the Max-Planck-Institut für Sonnensystemforschung, Katlenburg-Lindau, Germany, in collaboration with CISAS, University of Padova, Italy, the Laboratoire d'Astrophysique de Marseille, France, the Instituto de Astrofisica de Andalucia, CSIC, Granada, Spain, the Research and Scientific Support Department of the European Space Agency, Noordwijk, The Netherlands, the Instituto Nacional de Tecnica Aeroespacial, Madrid, Spain, the Universidad Politechnica de Madrid, Spain, the Department of Astronomy and Space Physics of Uppsala University, Sweden, and the Institut für Datentechnik und Kommunikationsnetze der Technischen Universität Braunschweig, Germany. The support of the national funding agencies of Germany (DLR), France (CNES), Italy (ASI), Spain (MEC), Sweden (SNSB), and the ESA Technical Directorate is gratefully acknowledged. We thank the Rosetta Science Operations Centre and the Rosetta Mission Operations Centre for the successful flyby of (21) Lutetia.

References

1. Bay, H., Ess, A., Tuytelaars, T., Gool, L.V.: Speeded-up robust features (SURF). Computer Vision and Image Understanding 110(3), 346–359 (2008); similarity Matching in Computer Vision and Multimedia
2. Carry, B., et al.: Physical properties of the ESA Rosetta target asteroid (21) Lutetia. II. Shape and flyby geometry. Astronomy and Astrophysics 523, A94+ (2010)
3. Carry, B., et al.: The KOALA Shape Modeling Technique Validated at (21) Lutetia by ESA Rosetta Mission. Bulletin of the American Astronomical Society, 1050–+ (2010)
4. Christy, S., Horaud, R.: Euclidean shape and motion from multiple perspective views by affine iterations. IEEE Transactions on Pattern Analysis and Machine Intelligence 18(11), 1098–1104 (1996)
5. De Cecco, M., Pertile, M., Baglivo, L., Lunardelli, M.: A unified framework for uncertainty, compatibility analysis, and data fusion for multi-stereo 3-d shape estimation. IEEE Trans. Instr. Meas. 59(11), 2834–2842 (2010)
6. Del Bue, A., Xavier, J., Agapito, L., Paladini, M.: Bilinear factorization via augmented lagrange multipliers. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 283–296. Springer, Heidelberg (2010)
7. Fischler, M., Bolles, R.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24(6), 381–395 (1981)
8. Hartley, R.I., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, Cambridge (2000); ISBN: 0521623049

9. Keller, H.U., et al.: OSIRIS The Scientific Camera System Onboard Rosetta. Space Science Review 128, 433–506 (2007)
10. Kochemasov, G.G.: Waiting for 21-Lutetia "Rosetta" images as a final proof of structurizing force of inertia-gravity waves. In: EGU General Assembly 2010, Vienna, Austria, May 2-7, p. 4070 (May 2010)
11. Kovesi, P.D.: MATLAB and Octave functions for computer vision and image processing. Centre for Exploration Targeting, School of Earth and Environment, The University of Western Australia, http://www.csse.uwa.edu.au/~pk/research/matlabfns/
12. Lamy, P.L., Faury, G., Jorda, L., Kaasalainen, M., Hviid, S.F.: Multi-color, rotationally resolved photometry of asteroid 21 Lutetia from OSIRIS/Rosetta observations. Astronomy ad Astrophysics 521, A19+ (2010)
13. Marques, M., Costeira, J.P.: Estimating 3d shape from degenerate sequences with missing data. Computer Vision and Image Understanding 113(2), 261–272 (2009)
14. Schweighofer, G., Pinz, A.: Globally Optimal O(n) Solution to the PnP Problem for General Camera Models. In: Proc. British Machine Vision Conference, pp. 1–8 (2008)
15. Tomasi, C., Kanade, T.: Shape and motion from image streams under orthography: A factorization approach. International Journal of Computer Vision 9(2) (1992)
16. Triggs, B., McLauchlan, P.F., Hartley, R.I., Fitzgibbon, A.W.: Bundle adjustment – A modern synthesis. In: Triggs, B., Zisserman, A., Szeliski, R. (eds.) ICCV-WS 1999. LNCS, vol. 1883, pp. 298–375. Springer, Heidelberg (2000), citeseer.nj.nec.com/triggs00bundle.html
17. Zeisl, B., Georgel, P., Schweiger, F., Steinbach, E., Navab, N.: Estimation of location uncertainty for scale invariant feature points. In: British Machine Vision Conference, BMVC (2009)

RT-SLAM: A Generic and Real-Time Visual SLAM Implementation

Cyril Roussillon[1,2,3], Aurélien Gonzalez[1,2],
Joan Solà[1,2,4], Jean-Marie Codol[1,2,5], Nicolas Mansard[1,2],
Simon Lacroix[1,2], and Michel Devy[1,2]

[1] CNRS, LAAS, 7 avenue du colonel Roche, F-31077 Toulouse, France
[2] Université de Toulouse, UPS, INSA, INP, ISAE, LAAS, F-31077 Toulouse, France
[3] Funded by the Direction Générale de l'Armement (DGA), France
[4] Ictineu Submarins, Industria 12, 08980 St. Feliu de Llobregat, Barcelona, Catalonia
[5] NAV ON TIME, 42 avenue du Général De Croutte, 31100 Toulouse, France
{firstname.name}@laas.fr

Abstract. This article presents a new open-source C++ implementation to solve the SLAM problem, which is focused on genericity, versatility and high execution speed. It is based on an original object oriented architecture, that allows the combination of numerous sensors and landmark types, and the integration of various approaches proposed in the literature. The system capacities are illustrated by the presentation of an inertial/vision SLAM approach, for which several improvements over existing methods have been introduced, and that copes with very high dynamic motions. Results with a hand-held camera are presented.

1 Motivation

Progresses in image processing, the growth of available computing power and the proposition of approaches to deal with bearings-only observations have made visual SLAM very popular, particularly since the demonstration of a real-time implementation by Davison in 2003 [4]. The Extended Kalman Filter (EKF) is widely used to solve the SLAM estimation problem, but it has recently been challenged by global optimization methods that have shown superior precision for large scale SLAM. Yet EKF still has the advantage of simplicity and faster processing for problems of limited size [14], and can be combined with global optimization methods [6] to take the best of both worlds.

Monocular EKF-SLAM reached maturity in 2006 with solutions for initializing landmarks [7] [12] [9]. Various methods for landmark parametrization have been analyzed [10], and the literature now abounds with contributions to the problem.

This article presents RT-SLAM[1], a software framework aimed at fulfilling two essential requirements. The first one is the need for a generic, efficient and flexible development tool, that allows to easily develop, evaluate and test various

[1] RT-SLAM stands for "Real-Time SLAM".

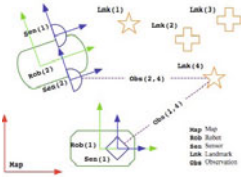
J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 31–40, 2011.
© Springer-Verlag Berlin Heidelberg 2011

approaches or improvements. The second one is the need for practical solutions for live experiments on robots, for which localization and mapping require real-time execution and robustness. RT-SLAM is a C++ implementation based on Extended Kalman Filter (EKF) that allows to easily change robot, sensor and landmark models. It runs up to 60 Hz with 640×480 images, withstanding highly dynamic motions, which is required for instance on humanoid or high speed all terrain robots. RT-SLAM is available as *open source* software at `http://rtslam.openrobots.org`.

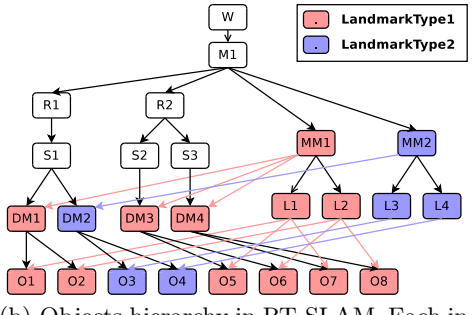
Section 2 details the architecture of RT-SLAM and section 3 presents some of the techniques currently integrated within, to define an efficient inertial/visual SLAM approach. Section 4 analyzes results obtained with a hand-held system assessed thanks to ground truth, and section 5 concludes the article by a presentation of prospects.

2 Overall Architecture

Fig. 1 presents the main objects defined in RT-SLAM. They encompass the basic concepts of a SLAM solution: the *world* or environment contains *maps*; maps contain *robots* and *landmarks*; robots have *sensors*; sensors make *observations* of landmarks. Each of these objects is abstract and can have different implementations. They can also contain other objects that may themselves be generic.



(a) Main objects in a SLAM context. Different robots `Rob` have several sensors `Sen`, providing observations `Obs` of landmarks `Lmk`. States of robots, sensors and landmarks are stored in the stochastic map `Map`. There is one observation per sensor-landmark pair.



(b) Objects hierarchy in RT-SLAM. Each individual map `M` in the world `W` contains robots `R` and landmarks `L`. A robot has sensors `S`, and an observation `O` is created for every pair of sensor and landmark. In order to allow full genericity, map managers `MM` and data managers `DM` are introduced.

Fig. 1. The main objects in RT-SLAM

Map. The maps contain an optimization or estimation engine: for now RT-SLAM uses a standard formulation of EKF-SLAM. Since this solution is very well documented in the literature [5], it is not detailed in depth here. Indirect

indexing within Boost's ublas C++ structures is intensively used to exploit the sparsity of the problem and the symmetry of the covariances matrices.

Robot. Robots can be of different type according to the way their state is represented and their prediction model. The latter can be either a simple kinematic model (constant velocity, constant acceleration, ...) or a proprioceptive sensor (odometric, inertial, ...), as illustrated section 3.3. The proprioceptive sensor is an example of generic object contained in robot objects, as different hardware can provide the same function.

Sensor. Similarly to robots, sensors can also have different models (perspective camera, panoramic catadioptric camera, laser, ...), and contain a generic exteroceptive sensor hardware object (firewire camera, USB camera, ...). In addition, as sensors belong to the map, their state can be estimated: this opens the possibility for estimating other parameters such as extrinsic calibration, time delays, biases and gain errors, and the like.

Landmark. Landmarks can be of different type (points, lines, planes, ...), and each type can have different state parametrization (Euclidean point, inverse depth point, ...). Moreover the parametrization of a landmark can change over time, as explained section 3.2. A landmark also contains a descriptor used for data association, which is a dual description to the state representation.

As shown Fig. 1(a), it is worth noticing that landmark objects are common to the different sensors, all of them being able to observe the same landmark (provided they have compatible descriptors for this landmark of course). This allows to greatly improve the observability of landmarks compared to a system where the sensors are strictly independent. In the particular case of two cameras for instance, landmarks can be used even if they are only visible from one camera or if they are too far away for a stereovision process to observe their depth (this process was introduced in [11] as *BiCam* SLAM).

Observation. In RT-SLAM, the notion of observation plays a predominant role. An observation is a real object containing both methods and data. One observation is instantiated for every sensor-landmark pair, regardless of the sensor having actually observed the landmark or not, and has the same lifetime as the associated landmark. The methods it contains are the conventional projection and back-projection models (that depend on the associated sensor and landmark models), while the stored data consist of results and intermediary variables such as Jacobian matrices, predicted and observed appearances, innovations, event counters and others, that allow to greatly simplify and optimize computations.

Managers. In order to achieve full genericity *wrt* landmark types, in particular to allow the concurrent use of different landmark types for one sensor, two different manager objects are added: *data manager* and *map manager*. Their implementations define a given management strategy, while their instantiations are dedicated to a certain landmark type. The data manager processes the sensors

raw data, providing observations of the external landmarks. For this purpose, it exploits some raw data processors (for feature detection and tracking), and decides which observations are to be corrected and in which order, according to the quantity of information they bring and their quality. For example it can apply an active search strategy and try to eliminate outliers as described in section 3.1. The map manager keeps the map clean, with relevant information, and at a manageable size, by removing landmarks according to their quality and the given policy (*e.g.* visual odometry where landmarks are discarded once they are not observed, or multimap slam where maps are "closed" according to given criteria). These managers communicate together: for example, the data manager may ask the map manager if there is enough space in the map to start a new initialization process, and to allocate the appropriate space for the new landmark.

3 Inertial/Visual SLAM within RT-SLAM

3.1 Active Search and One-Point RANSAC

The strategy currently implemented in RT-SLAM's data manager to deal with observations is an astute combination of *active search* [5] and outliers rejection using *one-point RANSAC* [3].

The goal of active search is to minimize the quantity of raw data processing by constraining the search in the area where the landmarks are likely to be found. Observations outside of this 3σ observation uncertainty ellipse would be anyway considered incompatible with the filter and ignored by the *gating* process. In addition active search gives the possibility to decide anytime to stop matching and updating landmarks with the current available data, thus enabling *hard real-time* constraints. We extended the active search strategy to landmark initialization: each sensor strives to maintain features in its whole field of view using a randomly moving grid of fixed size, and feature detection is limited to empty cells of the grid. Furthermore the good repartition of features in the field of view ensures a better observability of the motions.

Outliers can come from matching errors in raw data or mobile objects in the scene. Gating is not always discriminative enough to eliminate them, particularly right after the prediction step when the observation uncertainty ellipses can be quite large – unfortunately at this time the filter is very sensitive to faulty corrections because it can mistakenly make all the following observations incompatible. To prevent faulty observations, outliers are rejected using a one-point RANSAC process. It is a modification of RANSAC, that uses the Kalman filter to obtain a whole model with less points than otherwise needed, and provides a set of *strongly compatible* observations that are then readily corrected. Contrary to [3] where data association is assumed given when applying the algorithm, we do the data association along with the one-point RANSAC process: this allows to look for features in the very small strongly compatible area rather than the whole observation uncertainty ellipse, and to save additional time for raw data processing.

3.2 Landmark Parametrizations and Reparametrization

In order to solve the problem of adding to the EKF a point with unknown distance and whose uncertainty cannot be represented by a Gaussian distribution, point landmarks parametrizations and initialization strategies for monocular EKF-SLAM have been well studied [4] [8] [9]. The solutions now widely accepted are undelayed initialization techniques with *inverse depth* parametrization. Anchored Homogeneous Point [10] parametrization is currently used in RT-SLAM.

The drawback of inverse depth parametrizations is that they describe a landmark by at least 6 variables in the stochastic map, compared to only 3 for an Euclidean point $(x\ y\ z)^T$. Memory and temporal complexity being quadratic with the map size for EKF, there is a factor of 4 to save in time and memory by *reparametrizing* landmarks that have converged enough [2]. The map manager uses the linearity criterion proposed in [9] to control this process.

3.3 Motion Prediction

The easiest solution for EKF-SLAM is to use a robot kinematic model to do the filter prediction, such as a constant velocity model:

$$\mathcal{R} = (\mathbf{p}\ \mathbf{q}\ \mathbf{v}\ \mathbf{w})^T$$

where $\mathbf{p}$ and $\mathbf{q}$ are respectively the position and quaternion orientation of the robot, and $\mathbf{v}$ and $\mathbf{w}$ are its linear and angular velocities.

The advantage of such a model is that it does not require complicated hardware setup (only a simple camera), but its strong limitation is that the scale factor is not observable. A second camera with a known baseline can provide a proper scale factor, but one can also use a proprioceptive sensor for the prediction step. Furthermore it usually provides a far better prediction with smaller uncertainties than a simple kinematic model, which brings several benefits:

1. search areas for matching are smaller, so processing is faster,
2. linearization points are more accurate, so SLAM precision is better, or one can reduce the framerate to decrease CPU load for equivalent quality,
3. it allows to withstand high motion dynamics.

In the case of an Inertial Measurement Unit (IMU), the robot state is then:

$$\mathcal{R} = (\mathbf{p}\ \mathbf{q}\ \mathbf{v}\ \mathbf{a_b}\ \mathbf{w_b}\ \mathbf{g})^T$$

where $\mathbf{a_b}$ and $\mathbf{w_b}$ are the accelerometers and gyrometers biases, and $\mathbf{g}$ the 3D gravity vector. Indeed it is better for linearity reasons to estimate the direction of $\mathbf{g}$ rather than constraining the robot orientation to enforce $\mathbf{g}$ to be vertical.

A special care has to be taken for the conversion of the noise from continous time (provided by the manufacturer in the sensor's datasheet) to discrete time. As the perturbation is continuous white noise, the variance of the discrete noise grows linearly with the integration period.

3.4 Image Processing

Point extraction. Point extraction is based on Harris detector with several optimizations. Some of them are approximations: a minimal derivative mask $[-1, 0, 1]$ is used, as well as a square and constant convolution mask, in order to minimize operations. This allows the use of *integral images* [15] to efficiently compute the convolutions. Additional optimizations are related to active search (section 3.1): only one new feature is searched in a small region of interest, which eliminates the costly steps of thresholding and sub-maxima suppression.

Point matching. Point matching is based on Zero-mean Normalized Cross Correlation (ZNCC), also with several optimizations. Integral images are used to compute means and variances, and a hierarchical search is made (two searches at half and full resolution are sufficient). We also implemented *bounded partial correlation* [13] in order to interrupt the correlation score computation when there is no more hope to obtain a better score than the threshold or the best one up to now. To be robust to viewpoint changes and to track landmarks longer, tracking is made by comparing the initial appearance of the landmark with its current predicted appearance [5].

4 Results

Fig. 2 shows the hardware setup that has been used for the experiments. It is composed of a firewire camera rigidly tied to an IMU, on which four motion capture markers used to measure the ground truth are fixed.

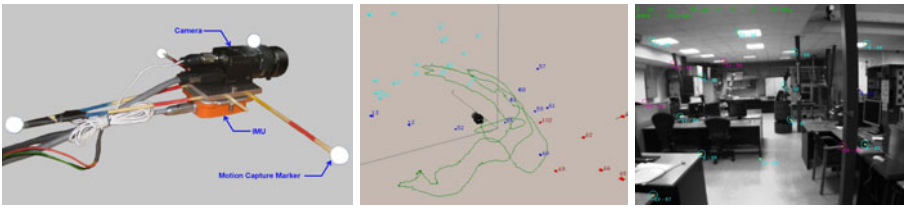


Fig. 2. The experimental setup composed of a Flea2 camera and an XSens MTi IMU, and screen captures of the 3D and 2D display of RT-SLAM

Two different sequences are used, referred to as *low dynamic* and *high dynamic* sequences. Both were acquired indoor with artificial lights only, with an image framerate of 50 Hz synchronized to the inertial data rate of 100 Hz. The motion capture markers are localized with a precision of approximately 1 mm, so the ground truth has a precision of $\sigma_{xyz} = 1\,\text{mm}$ in position and $\sigma_{wpr} = 0.57°$ in angle (baseline of 20 cm).

Movies illustrating a run of every experiment are provided at:
`http://rtslam.openrobots.org/Material/ICVS2011`.

4.1 Constant Velocity Model

The inertial data is here not used, and the prediction is made according to a constant velocity model. Fig. 3 presents the estimated trajectory and the ground truth for the *low dynamic* sequence, and Fig. 4 shows the absolute errors for the same run.

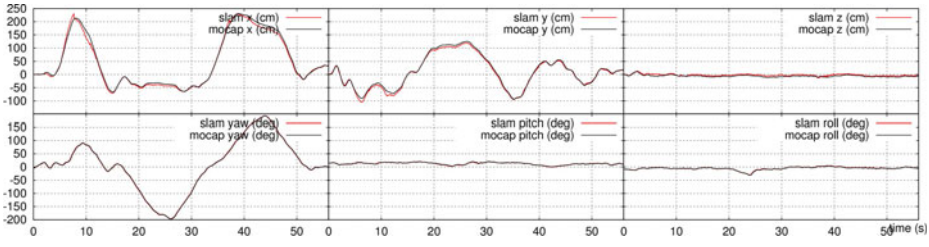


Fig. 3. Illustration of low dynamic trajectory, constant velocity SLAM (with scale factor of 2.05 manually corrected). Estimated trajectory parameters and ground truth, as a function of time (in seconds).

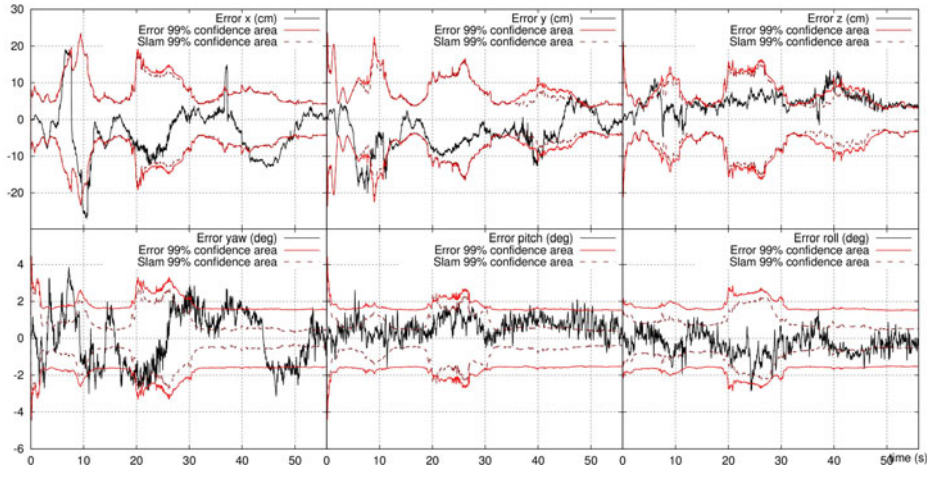


Fig. 4. Errors of the estimated parameters of the trajectory shown Fig. 3. The 99% confidence area corresponds to 2.57σ bounds. The SLAM 99% confidence area, that does not include ground truth uncertainty, is also shown.

Besides the global scale factor which is manually corrected, the camera trajectory is properly estimated. The movie shows that loops are correctly closed, even after a full revolution.

The position error raises up to 10 cm after quick rotations of the camera: this is due to a slight *drift* of the scale factor caused by the integration of newer

landmarks (when closing loops, the scale factor is restimated closer to its initial value). Also, uncertainty ellipses remain relatively large throughout the sequence: these issues are solved by the use of an IMU to predict the motions.

4.2 Inertial/Visual SLAM

Fig. 5 shows the trajectory estimated with the *high dynamic* sequence, and Fig. 6 and 7 show the behavior of inertial SLAM. Here, all of the 6 degrees of freedom are successively excited, then y and yaw are excited with extreme dynamics: the yaw angular velocity goes up to 400 deg/s, the rate of change of angular velocity exceeds 3,000 deg/s^2, and accelerations reach $3\,g$. It is interesting to note that the time when SLAM diverges (around $t = 65$ s) corresponds to motions for which the angular velocity exceeds the limit of the IMU (300 deg/s) and where its output is saturated.

The IMU now allows to observe the scale factor, and at the same time reduces the observation uncertainty ellipses and thus eases the active search procedure. Conversely, the divergence of the SLAM process at the end of the sequence illustrates what happens when vision stops stabilizing the IMU: it quickly diverges because the biases and the gravity cannot be estimated anymore.

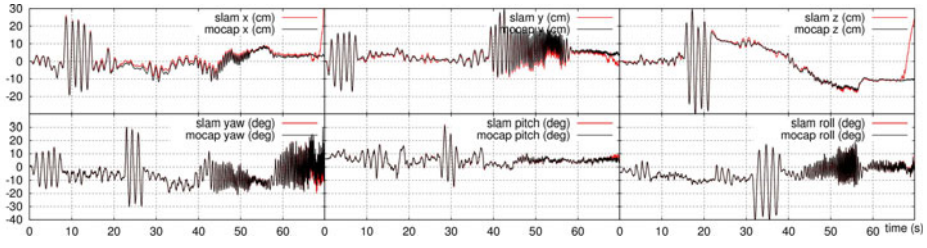


Fig. 5. Illustration of high dynamic trajectory, inertial/visual SLAM. Estimated trajectory parameters and ground truth, as a function of time (in seconds).

5 Outlook

We have presented a complete SLAM architecture whose genericity and performance make it both a useful experimentation tool and efficient solution for robot localization. The following extensions are currently being developed: using a second camera to improve landmarks observability, a multimap approach to cope with large scale trajectories and multirobot collaborative localization, and the use of line landmarks complementarily to points to provide more meaningful maps and to ease map matching for loop closure.

The architecture of RT-SLAM allows to easily integrate such developments, and also to consider additional motion sensors: to increase the robustness of the system, it is indeed essential to consider the various sensors usually found on board a robot (odometry, gyrometers, GPS). Eventually, it would be interesting to make RT-SLAM completely generic *wrt* the estimation engine as well, in order to be able to use global optimization techniques in place of filtering.

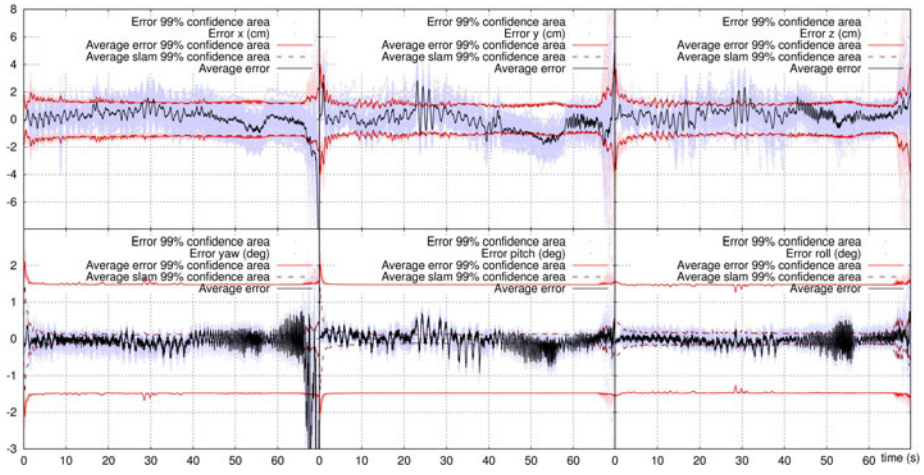


Fig. 6. Errors of inertial/visual SLAM estimated over 100 runs on the dynamic sequence, as a function of time (in seconds) – the difference between each run is due to the randomized landmark selection, see section 3.1. All the runs remain in the neighborhood of the 99% confidence area. The angular ground truth uncertainty is not precisely known: its theoretical value is both overestimated and predominant over SLAM precision in such a small area.

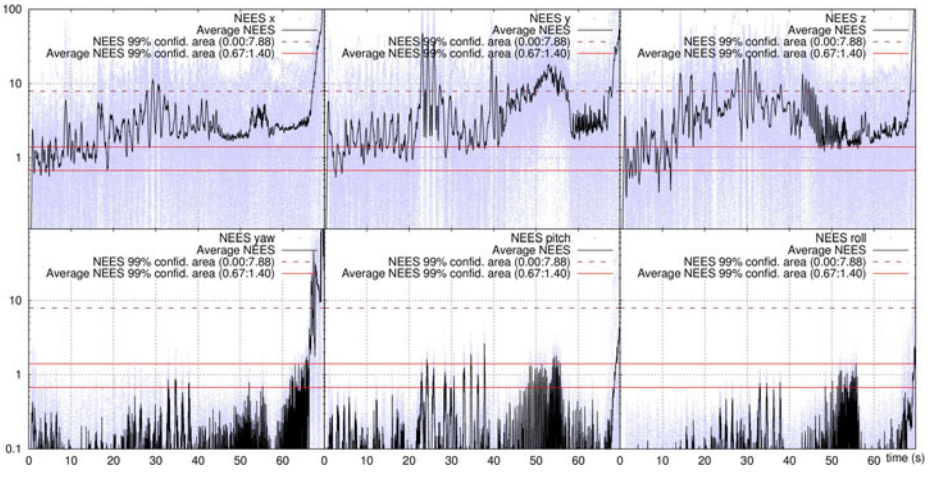


Fig. 7. Inertial/visual SLAM NEES [1] over 100 runs on the dynamic sequence. The average NEES is quickly out of the corresponding 99% confidence area, but as the 100 runs were made on the same sequence they are not independant and the average NEES should rather be compared to the simple NEES confidence area. The filter appears to be very conservative with angles but this is due to the overestimated ground truth uncertainty as explained in Fig. 6.

References

1. Bailey, T., Nieto, J., Guivant, J., Stevens, M., Nebot, E.: Consistency of the ekf-slam algorithm. In: Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3562–3568 (October 2006)
2. Civera, J., Davison, A.J., Montiel, J.M.M.: Inverse Depth to Depth Conversion for Monocular SLAM. In: Proc. of IEEE International Conference on Robotics and Automation (ICRA), pp. 2778–2783 (April 2007)
3. Civera, J., Grasa, O.G., Davison, A.J., Montiel, J.M.M.: 1-point ransac for ekf-based structure from motion. In: Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3498–3504 (October 2009)
4. Davison, A.J.: Real-time simultaneous localisation and mapping with a single camera. In: Proc. of IEEE International Conference on Computer Vision (ICCV), pp. 1403–1410 (October 2003)
5. Davison, A.J., Reid, I.D., Molton, N.D., Stasse, O.: Monoslam: Real-time single camera slam. IEEE Trans. Pattern Anal. Mach. Intell. 29, 1052–1067 (2007)
6. Estrada, C., Neira, J., Tardos, J.D.: Hierarchical slam: Real-time accurate mapping of large environments. IEEE Trans. on Robotics 21(4), 588–596 (2005)
7. Kwok, N.M., Dissanayake, G.: An efficient multiple hypothesis filter for bearing-only slam. In: Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), vol. 1, pp. 736–741 (2004)
8. Lemaire, T., Lacroix, S., Sola, J.: A practical 3d bearing-only slam algorithm. In: Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2449–2454 (August 2005)
9. Montiel, J.M.M.: Unified inverse depth parametrization for monocular slam. In: Proc. of Robotics: Science and Systems (RSS), pp. 16–19 (2006)
10. Sola, J.: Consistency of the monocular ekf-slam algorithm for three different landmark parametrizations. In: Proc. of IEEE International Conference on Robotics and Automation (ICRA), pp. 3513–3518 (May 2010)
11. Sola, J., Monin, A., Devy, M.: Bicamslam: Two times mono is more than stereo. In: Proc. of IEEE International Conference on Robotics and Automation (ICRA), pp. 4795–4800 (2007)
12. Sola, J., Monin, A., Devy, M., Lemaire, T.: Undelayed initialization in bearing only slam. In: Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2499–2504 (2005)
13. Di Stefano, L., Mattoccia, S., Tombari, F.: Zncc-based template matching using bounded partial correlation. Pattern Recognition Letters 26(14), 2129–2134 (2005)
14. Strasdat, H., Montiel, J.M.M., Davison, A.J.: Real-time monocular slam: Why filter? In: Proc. of IEEE International Conference on Robotics and Automation (ICRA), pp. 2657–2664 (May 2010)
15. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proc. IEEE Conf. Comput. Vis. Pattern Recog., pp. 511–518 (2001)

A Quantitative Comparison of Speed and Reliability for Log-Polar Mapping Techniques

Manuela Chessa, Silvio P. Sabatini, Fabio Solari, and Fabio Tatti

Department of Biophysical and Electronic Engineering, University of Genoa
Via all'Opera Pia 11a - 16145 Genova - Italy
manuela.chessa@unige.it
www.pspc.dibe.unige.it

Abstract. A space-variant representation of images is of great importance for active vision systems capable of interacting with the environment. A precise processing of the visual signal is achieved in the fovea, and, at the same time, a coarse computation in the periphery provides enough information to detect new saliences on which to bring the focus of attention. In this work, different techniques to implement the blind-spot model for the log-polar mapping are quantitatively analyzed to assess the visual quality of the transformed images and to evaluate the associated computational load. The technique with the best trade-off between these two aspects is expected to show the most efficient behaviour in robotic vision systems, where the execution time and the reliability of the visual information are crucial.

1 Introduction

The vision modules of robotic systems, which continuously interact with the environment by purposefully moving the eyes to bring the interesting objects into the foveas [1,2], require mechanisms that simultaneously provide a wide field-of-view, a high spatial resolution in the region of interest, and a reduction of the amount of data to be processed. This can be achieved through a variable resolution (or space-variant) image acquisition stage, inspired by the primate visual system. Indeed, the distribution of the photoreceptors in the mammals' retina is denser in the central region, the fovea, whereas it is sparser in the periphery. In particular, the projection of the photoreceptor array into the primary visual cortex can be well described by a log-polar mapping [3].

In the literature, several authors apply the log-polar transform to solve different visual processing tasks in computer vision and robotic applications, e.g. vergence control [4], and binocular tracking [5] in active vision systems. An approach for the computation of binocular disparity is described in [6], and in [7] the author faces the estimation of the optic flow in cortical images. Moreover, the log-polar imaging has been applied in several pattern recognition tasks, such as object detection [8], and facial feature detection [9]. For recent reviews of the different applications of the log-polar mapping approach see [10,11].

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 41–50, 2011.
© Springer-Verlag Berlin Heidelberg 2011

Different models for mapping the Cartesian images into the log-polar domain have been proposed [12,13,14]. Among them, the blind-spot model is particularly interesting, since its implementation is easier and computationally less demanding than that of the other models. Moreover, two interesting properties come with it: rotation and scale invariance. Several techniques are present in the literature to implement the blind-spot model, nevertheless a systematic analysis and a comparison among them is still lacking, but see [15].

The aim of this paper is to analyze the different techniques developed to implement the blind-spot model. To this end, we perform a quantitative comparison that takes into account the computational load, and the amount of visual information preserved in the transformation. Specifically, we compute different image quality indexes [16,17] for assessing the degradation of the visual information in the transformed image. It is worth noting that we are interested in how the space-variant subsampling of the images is achieved, since we wonder if the different techniques yield an undesired loss of details in the periphery, in addition to the one due to the mapping itself.

2 Log-Polar Mapping: The Blind-Spot Model

The log-polar mapping is a non linear transformation that maps each point of the Cartesian domain (x, y) into a cortical domain of coordinates (ξ, η). For the blind spot model [14], the transformation is described by the following equations:

$$\begin{cases} \xi = \log_a \left(\frac{\rho}{\rho_0} \right) \\ \eta = \theta \end{cases} \tag{1}$$

where a parameterizes the non-linearity of the mapping, ρ_0 is the radius of the central blind spot, and $(\rho, \theta) = (\sqrt{x^2 + y^2}, \arctan(y/x))$ are the polar coordinates derived from the Cartesian ones. All points with $\rho < \rho_0$ are ignored.

In order to deal with digital images, given a Cartesian image of $M \times N$ pixels, and defined $\rho_{max} = 0.5 \min(M, N)$, we obtain an $R \times S$ (rings $\times$ sectors) discrete cortical image of coordinates (u, v) by taking:

$$\begin{cases} u = \lfloor \xi \rfloor \\ v = \lfloor q\eta \rfloor \end{cases} \tag{2}$$

where $\lfloor \cdot \rfloor$ denotes the integer part, $q = S/(2\pi)$, and $a = \exp(\ln(\rho_{max}/\rho_0)/R)$. Figure 1 shows the transformations through the different domains. The retinal area that refers to a given cortical pixel defines its *receptive field* (RF). By inverting Eq. 1 the centers of the RFs can be computed, and these points present a non-uniform distribution through the retinal plane, as in Figure 2a. The optimal relationship between R and S is the one that optimizes the pixel aspect ratio, making it as close as possible to 1. It can be shown that, for a given R, the optimal rule is $S = 2\pi/(a - 1)$ [14].

The shape of the RFs affects both the quality of the transformation and its computational burden. In the following, we consider four common techniques, each characterized by a different shape for the RFs.

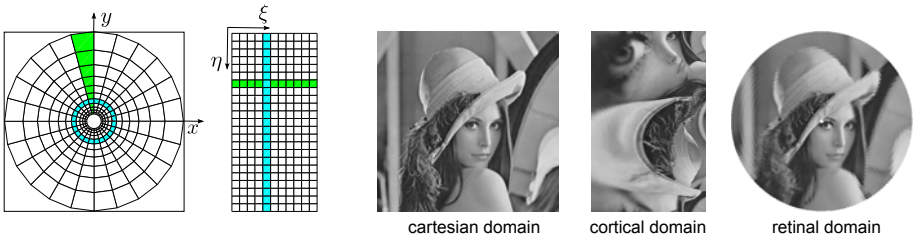


Fig. 1. Left: the cyan circle and the green sector in the Cartesian domain map to vertical and horizontal stripes, respectively, in the cortical domain. Right: an example of image transformation from the Cartesian to the cortical domain, and backward to the retinal domain. The specific choice of the parameters is: $R = 100$, $S = 157$, $\rho_0 = 5$, and $\rho_{max} = 256$. The cortical image is scaled to improve the visualization.

Nearest Pixel. In this technique (see Figure 2a), the RFs are single points. Each RF is located, in the most general case, between four adjacent Cartesian pixels. This technique assigns to the cortical pixel that refers to a given RF, the value of the closest Cartesian pixel. The inverse transformation follows the same principle.

Bilinear Interpolation. Similarly to the previous technique, the RFs are single points, as shown in Figure 2a. The difference is that, in this case, the value of a desired cortical pixel is obtained through a bilinear interpolation of the values of the four nearest neighbouring Cartesian pixels to the center of the RF [18]. The same principle is applied to the inverse transformation.

Overlapping Circular Receptive Fields. This biological plausible technique is a modified implementation of the one proposed in [12,15]. The Cartesian plane is divided in two regions: the *fovea* and the *periphery*. The periphery is defined as the part of the plane in which the distance between the centers of two RFs on the same radius is greater than 1 (undersampling). The fovea (oversampling) is handled by using the bilinear interpolation technique described above, whereas in the periphery we use the overlapping Gaussian circular RFs shown in Figure 2b.

The standard deviation of the RF Gaussian profile is a third of the distance between the centers of two consecutive RFs, and the spatial support is six times the standard deviation. As a consequence of this choice, adjacent RFs overlap [12]. A cortical pixel C_i is computed as a Gaussian weighted sum of the Cartesian pixels P_j in the i-th RF:

$$C_i = \sum_j w_{ij} P_j \tag{3}$$

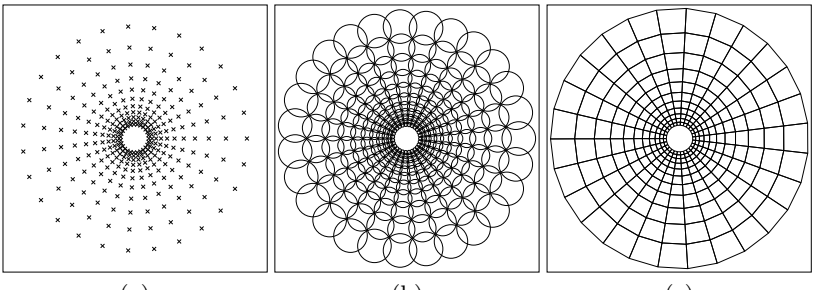
where the weights w_{ij} are the values of a normalized Gaussian centered on the i-th RF. A similar approach is used to compute the backward transformation.

Adjacent Receptive Fields. In this technique [14], all the Cartesian pixels, whose coordinates in the cortical domain share the same integer part, are

assigned to the same RF, see Figure 2c. The value of a cortical pixel C_i can be calculated through a weighted sum of the values of the pixels that belong to the corresponding RF, normalized with respect to the total sum of weights. The precision of the boundaries of the RF can be improved by breaking each pixel into subpixels and assigning each of them to the correct RF. Consequently, the weight of each Cartesian pixel P_j is the fraction A_{ij} of the pixel area that belongs to the i-th RF:

$$C_i = \frac{1}{\sum_j A_{ij}} \sum_j A_{ij} P_j \tag{4}$$

If no pixel subdivision is applied, the sum reduces to a simple average. In the following we consider a pixel subdivision equals to 1/4. The cortical image can be backward transformed by assigning to each retinal pixel the value of its corresponding cortical pixel by following an approach similar to the one of the forward transformation.



(a) (b) (c)

Fig. 2. The RFs for the different implementations of the blind-spot model, represented on the Cartesian image (the square frame): (a) nearest pixel and bilinear interpolation (the centers of the RFs); (b) overlapping circular RFs (plotted at twice the value of the standard deviation); (c) adjacent RFs.

3 Comparison and Results

The four different techniques, described in the previous Sections, are here compared in terms both of their associated computational load (i.e. execution time), and of the quality of the resulting transformed images.

3.1 Computational Load

It is important to consider that, in the vision modules of a robotic system, all the image processing steps must be performed at a given frequency in order to meet the real-time constraints. Thus, the log-polar mapping should be much faster than the usual 40 ms/frame. Figure 3 shows the execution times for transforming the images into the log-polar domain, for different sizes of the Cartesian input image and of the target cortical image (the latter defined by different R and with

S computed in order to keep the aspect ratio of the log-polar pixel equals to 1). The transformations are implemented in C++, the codes are compiled in release version, and run on an Intel Core i7 2.8 GHz. The results show how the nearest pixel and the bilinear interpolation techniques are ten times faster than the solutions based on RFs. In particular, these two techniques are slightly affected

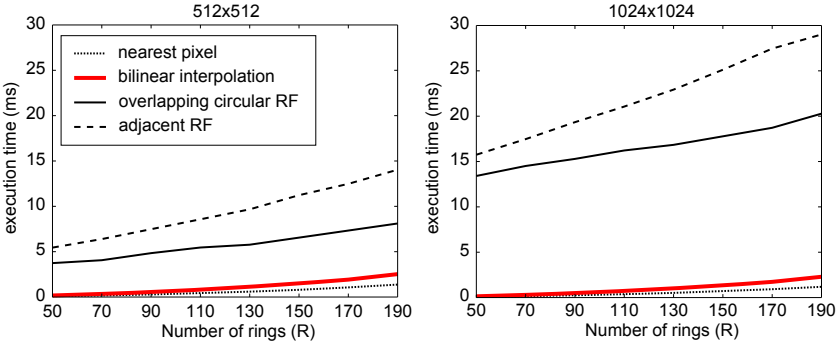


Fig. 3. Execution times (ms) with respect to R of the cortical image for the transformation of two Cartesian images (512×512 and 1024×1024 pixels) into the log-polar domain by using the four techniques of the blind-spot model with $\rho_0 = 5$.

by the size of the input image, since they are based on point-wise processing, and, consequently, the number of online computations is $O(K)$, where K is the number of cortical pixels. The other two techniques, based on RFs, require $O(KH)$ operations, where H denotes the number of Cartesian pixels. For these reasons, we have decided not to implement the matrix approach proposed in [15]. It is worth noting that we have neglected to consider the inverse transformation (i.e. from the cortical to the retinal domain), since robotic vision systems should use the information computed in the cortical domain directly. In the following, we will consider the backward transformed retinal images only for the sake of comparison with the reference Cartesian ones.

3.2 Image Quality Indexes

In the literature, several authors address the problem of quality image assessment in the field of lossy image compression. A major problem in evaluating lossy techniques is the extreme difficulty in describing the type and the amount of degradation in the reconstructed images. Thus, to obtain a reliable image quality assessment, it is common to consider the results obtained with different indexes. The same approach is here followed to evaluate the degradation of the transformed images in the log-polar domain, for different compression ratios.

Several quantitative measures appeared in the literature [19]: the simplest and most widely used full-reference quality metric is the mean squared error, along with the related quantity of Peak Signal-to-Noise Ratio (PSNR). These

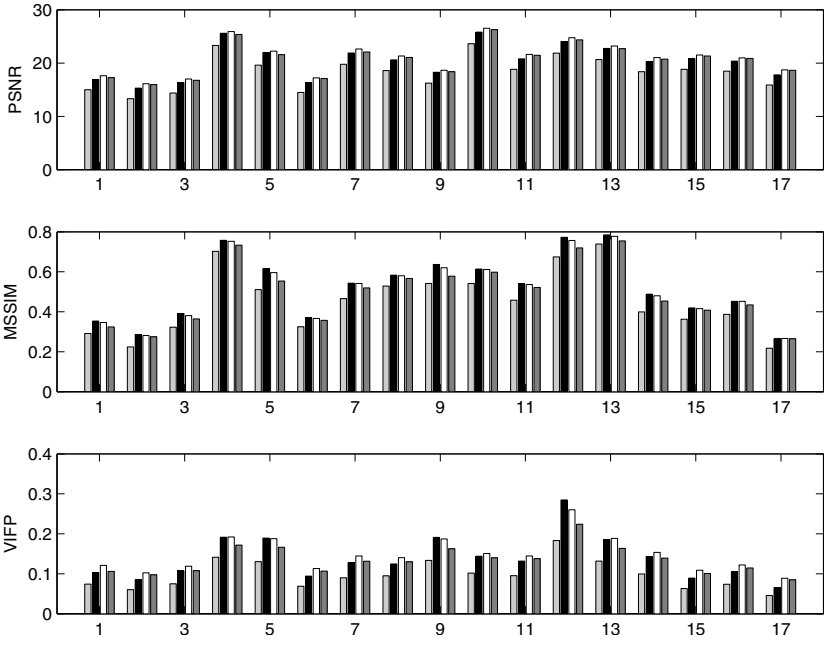


Fig. 4. Quality indexes (PSNR, MSSIM, and VIFP) computed on the considered image subset, numbered from 1 to 17, of the LIVE database. The bars represent the four different techniques to implement the blind-spot model: light gray for the nearest pixel, black for the bilinear interpolation, white for the overlapping circular RF, and dark gray for adjacent RF, respectively. The parameters of the log-polar transformations are: $R = 70$, $S = 109$, and $\rho_0 = 5$.

quantities are easy to compute and have a clear physical meaning, though they are not very well suited to match the perceived visual quality. In [16] the authors proposed the use of the Mean Structural SIMilarity (MSSIM) to design an image quality measure that takes advantage of the known characteristics of the human visual system. Along this line, in [17] the Visual Information Fidelity (VIF) index is introduced, which is derived from a statistical model for natural scenes and from a model of image distortions, and it also takes into account considerations on the human vision system.

The results presented in this Section are obtained by using the following quality indexes: PSNR, the MSSIM implemented by [16], and the pixel domain version of visual information fidelity (VIFP) implemented by [17]. The four implementations of the blind-spot model have been compared by using 17 images (512×512 pixels) from the LIVE Image Quality Assessment Database [20]. The considered images are: *bikes, building2, buildings, caps, coinsinfountain, flowersonih35, house, lighthouse2, monarch, ocean, paintedhouse, parrots, plane, rapids, sailing1, sailing4, stream*, in the following numbered from 1 to 17.

We have chosen two different values of cortical rings ($R = 70$ and $R = 150$) in order to obtain two different compression ratios: 1:34 and 1:7, respectively. In

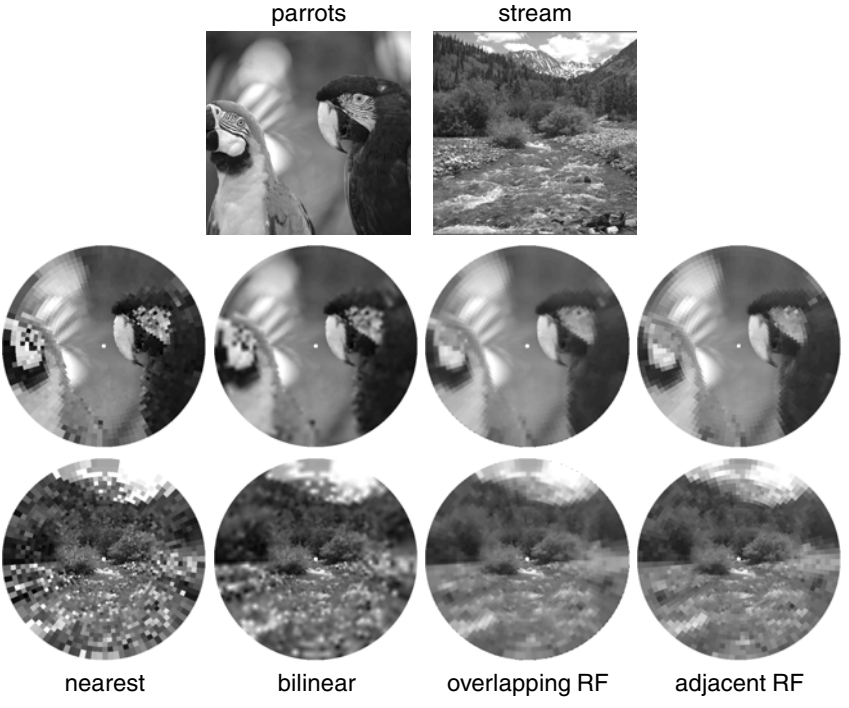


Fig. 5. Examples of log-polar transformations with the four considered techniques for two images of the LIVE database. The square images are the Cartesian reference ones. The circles below represent the corresponding transformed images, by the different techniques analyzed in this paper. The used parameters are the same of Fig. 4.

general, the quality indexes have higher values for cortical images with dimension of 150×237 pixels, due to the lower compression ratio. Figure 4 shows the indexes computed for the 17 selected images for $R = 70$. It is worth noting the variability of the index values, with respect to the different images. This can be explained by the different properties of the images: complex textures are indeed affected by the decreased resolution in the periphery, thus leading to a lower quality of the transformed images. Figure 5 shows the transformations obtained for two significant cases: images 12 (*parrots*) and 17 (*stream*), since they present the highest and the lowest quality indexes, respectively. In general, it is worth nothing that the bilinear interpolation technique (black bar in Fig. 4) has better performances, with respect to the other solutions, with images characterized by gross uniform textures, thus yielding higher quality indexes (especially, for the MSSIM and the VIFP). On the contrary, for images characterized by fine details, thus more degraded by the log-polar transformation, the adjacent RF and the overlapping RF techniques (dark gray and white bars, respectively) perform slightly better. Nevertheless, considering both the quality indexes separately and their mean values and standard deviation (see Table 1) we can conclude that the bilinear interpolation technique is characterized by quality indexes comparable

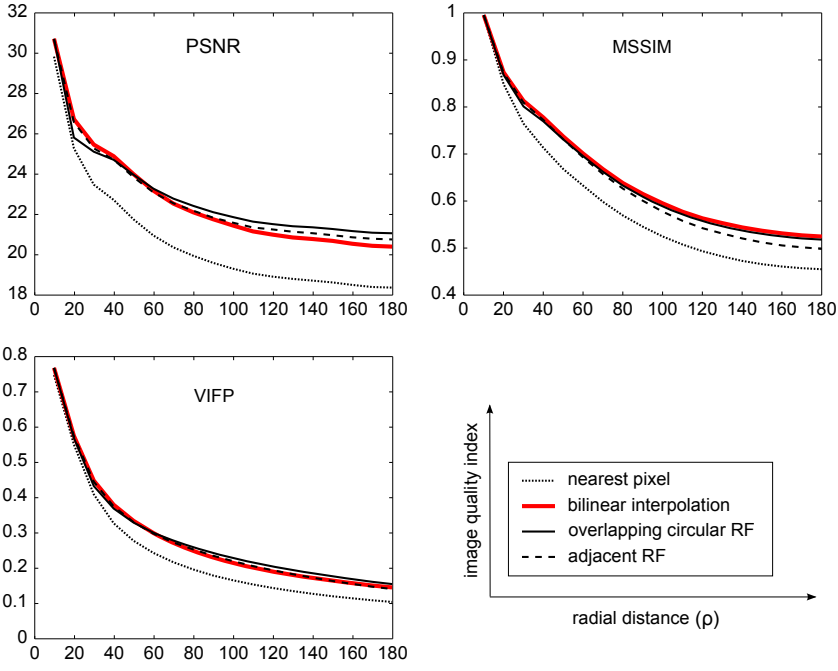


Fig. 6. The average quality indexes (PSNR, MSSIM, and VIFP) as a function of the radial position, for the four solutions of the blind-spot model, by considering the chosen subset of the LIVE database. Mapping parameters as in Fig. 4.

to the solutions based on RFs. The nearest pixel solution yields the worst results, since it is most affected by the artifacts due to the aliasing. Moreover, we have analyzed how the quality of the transformed images is affected by the distance from their center (the fovea), by computing the quality indexes for all the pixels within a given radius. Figure 6 shows that the bilinear interpolation (thick red line) and the overlapping RF (thin black line) implementations of the blind-spot model are the best solutions for all the eccentricities, whereas the nearest pixel (dotted line) yields the lower values of the indexes. This result is consistent with the findings shown in [15].

Table 1. The mean and standard deviation values of the three quality indexes for the four solutions of the blind-spot model, for the selected images of the LIVE database

	$R = 70$, $S = 109$, $\rho_0 = 5$			$R = 150$, $S = 237$, $\rho_0 = 5$		
	PSNR	MSSIM	VIFP	PSNR	MSSIM	VIFP
nearest	18.31±2.99	0.45±0.15	0.10±0.03	20.28±3.10	0.58±0.13	0.18±0.05
bilinear	20.34±3.09	0.52±0.16	0.14±0.05	22.48±3.15	0.65±0.13	0.23±0.06
overlapping RF	21.00±3.01	0.52±0.16	0.15±0.04	22.77±3.10	0.64±0.13	0.24±0.05
adjacent RF	20.70±2.93	0.50±0.15	0.13±0.03	22.56±3.08	0.63±0.12	0.23±0.05

4 Conclusions and Future Work

In this paper, we have compared different techniques to implement the log-polar blind-spot model (nearest pixel, bilinear interpolation, overlapping circular RF, and adjacent RF), from an image quality point of view, and for what concerns the associated computational load. The analysis of the image quality, i.e. the preservation of the visual information in the log-polar transformation, has been conducted by using three widely used image quality indexes. The experimental validation shows how the simple and easy-to-implement bilinear interpolation technique yields good results in terms of the quality of the transformed image. Moreover, such a technique is ten times faster than the solutions based on RFs. Thus, we can conclude that the bilinear interpolation solution of the blind-spot model is the best trade-off for all the applications that require a fast and reliable processing of the images, such as those typical of robotic vision. Indeed, the possibility of efficiently exploiting a space-variant representation is of great importance in the development of active vision systems capable of interacting with the environment, since a precise processing of the visual signal is possible in the foveal area, where the errors in the considered visual feature (e.g. binocular disparity and optic flow) are small enough to allow a fine exploration of the object of interest. At the same time, the coarse computation of the feature in the peripheral area provides enough information to detect new saliencies and to direct the focus of attention there. It is worth noting that this cannot be achieved through an uniform downsampling of the images.

In this paper, we have analyzed the different solutions by examining the quality of the retinal image after the inverse mapping. In a future work, we will comparatively assess the reliability of the visual feature extraction directly in the cortical domain, by using the different techniques.

The software libraries used in this paper are available to the Computer Vision community at `www.pspc.dibe.unige.it/Research/logpolar.html`.

Acknowledgements. This work has been partially supported by University of Genoa (Progetto di Ateneo 2010), and by Italian MIUR (PRIN 2008) "Bio-inspired models for the control of robot ocular movements during active vision and 3D exploration".

References

1. Aloimonos, J., Weiss, I., Bandyopadhyay, A.: Active vision. International Journal of Computer Vision 1(4), 333–356 (1988)
2. Bernardino, A., Santos-Victor, J.: Visual behaviours for binocular tracking. Robotics and Autonomous Systems 25(3-4), 137–146 (1998)
3. Schwartz, E.L.: Spatial mapping in the primate sensory projection: Analytic structure and relevance to perception. Biological Cybernetics 25, 181–194 (1977)
4. Zhang, X., Tay, L.P.: A spatial variant approach for vergence control in complex scenes. Image Vision Comput. 29, 64–77 (2011)

5. Bernardino, A., Santos-Victor, J., Sandini, G.: Foveated active tracking with redundant 2D motion parameters. Robotics and Autonomous Systems, 205–221 (2002)
6. Bernardino, A., Santos-Victor, J.: A binocular stereo algorithm for log-polar foveated systems. In: Bülthoff, H.H., Lee, S.-W., Poggio, T.A., Wallraven, C. (eds.) BMCV 2002. LNCS, vol. 2525, pp. 127–136. Springer, Heidelberg (2002)
7. Yeasin, M.: Optical flow in log-mapped image plane: A new approach. IEEE Trans. Pattern Anal. Mach. Intell. 24, 125–131 (2002)
8. Amiri, M., Rabiee, H.R.: A novel rotation/scale invariant template matching algorithm using weighted adaptive lifting scheme transform. Pattern Recognition 43, 2485–2496 (2010)
9. Smeraldi, F., Bigun, J.: Retinal vision applied to facial features detection and face authentication. Pattern Recognition Letters 23, 463–475 (2002)
10. Berton, F., Sandini, G., Metta, G.: Anthropomorphic visual sensors. In: Encyclopedia of Sensors. In: Encyclopedia of Sensors, pp. 1–16. American Scientific Publishers (2006)
11. Traver, V.J., Bernardino, A.: A review of log-polar imaging for visual perception in robotics. Robotics and Autonomous Systems 58(4), 378–398 (2010)
12. Bolduc, M., Levine, M.D.: A real-time foveated sensor with overlapping receptive fields. Real-Time Imaging 3(3), 195–212 (1997)
13. Jurie, F.: A new log-polar mapping for space variant imaging. Application to face detection and tracking. Pattern Recognition 32, 865–875 (1999)
14. Traver, V.J., Pla, F.: Log-polar mapping template design: From task-level requirements to geometry parameters. Image Vision Comput. 26(10), 1354–1370 (2008)
15. Pamplona, D., Bernardino, A.: Smooth foveal vision with Gaussian receptive fields. In: 9th IEEE-RAS Int. Conf. on Humanoid Robots, pp. 223–229 (2009)
16. Wang, Z., Bovik, A.C., Sheikh, H.R., Simoncelli, E.P.: Image quality assessment: From error visibility to structural similarity. IEEE Transactions on Image Processing 13(4), 600–612 (2004)
17. Sheikh, H.R., Bovik, A.C.: Image information and visual quality. IEEE Transactions on Image Processing 15(2), 430–444 (2006)
18. Young, D.S.: Straight lines and circles in the log-polar image. In: British Machine Vision Conference, pp. 426–435 (2000)
19. Eskicioglu, A.M., Fisher, P.S.: Image quality measures and their performance. IEEE Transactions on Communications 43(12), 2959–2965 (1995)
20. Sheikh, H., Wang, Z., Cormack, L., Bovik, A.: Live image quality assessment database release 2, http://live.ece.utexas.edu/research/quality

Toward Accurate Feature Detectors Performance Evaluation

Pavel Smirnov, Piotr Semenov, Alexander Redkin, and Anthony Chun

Intel Labs
{pavel.s.smirnov,piotr.semenov,alexandr.redkin,anthony.l.chun}@intel.com

Abstract. The quality of interest point detectors is crucial for many computer vision applications. One of the frequently used integral methods to compare detectors' performance is repeatability score. In this work, the authors analyze the existing approach for repeatability score calculation and highlight some important weaknesses and drawbacks of this method. Then we propose a set of criteria toward more accurate integral detector performance measure and introduce a modified repeatability score calculation. We also provide illustrative examples to highlight benefits of the proposed method.

1 Introduction

Image feature detection is a key problem in a wide range of computer vision (CV) applications. The performance of CV applications typically depends on the detection robustness for different transformations of images. Common integral "measure" of detector's performance is a very necessary tool for evaluation. One of the most widely used (for example, in [6,3,2,4,5], and other works) integral performance measures is the repeatability score [10].

In this work, we provide a detailed analysis of this method, which reveals some important weaknesses and shortcomings, and then propose a set of modifications to address them.

This paper is organized as follows: in Section 2, we analyze the existing repeatability score calculation algorithm and highlight some drawbacks, which, in our opinion, significantly reduce the accuracy of this method. In Section 3, we propose a modified approach to repeatability score calculation that does not suffer from these drawbacks. Finally, in Section 4 we summarize our results and future work.

2 Repeatability Score Analysis

2.1 Definitions

The repeatability score was originally introduced in [10]: "The repeatability rate is the percentage of the total observed points which are repeated between two images".

Below are definitions from [10] (with slightly modified notation):

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 51–60, 2011.
© Springer-Verlag Berlin Heidelberg 2011

1. A point p_i detected in image I_i is repeated in image I_j if the corresponding point p_j is detected in image I_j. To actually measure the repeatability, a relation between p_i and p_j, has to be established. In general this is impossible, but if the scene is planar this relation is defined by a homography $H_{j,i}$:

$$p_j = H_{j,i} \cdot p_i \tag{1}$$

2. Some points can not be repeated due to image parts which exist only in one of the images. Points d_j and d_i which could potentially be detected in both images are defined by

$$\begin{aligned} \{d_i\} &= \{p_i \mid H_{j,i} \cdot p_i \in I_j\} \\ \{d_j\} &= \{p_j \mid H_{i,j} \cdot p_j \in I_i\} \end{aligned} \tag{2}$$

   where $\{p_i\}$ and $\{p_j\}$ are the points detected in images I_i and I_j.
3. A point is not in general detected exactly at position p_j, but rather in some neighborhood of p_j. The size of this neighborhood is denoted by ϵ and repeatability within this neighborhood is called ϵ-repeatability. The set of point pairs (d_j, d_i) which corresponds within an ϵ-neighborhood is

$$D_{i,j}(\epsilon) = \{(d_j, d_i) \mid \|d_j - H_{j,i} \cdot d_i\| < \epsilon\} \tag{3}$$

4. The percentage of detected points which are repeated is the repeatability rate from image I_i to image I_j.

$$r_{i,j}(\epsilon) = \frac{|D_{i,j}(\epsilon)|}{\min(|\{d_i\}|, |\{d_j\}|)} \tag{4}$$

   From [10] "we can easily verify that"

$$0 \leq r_{i,j}(\epsilon) \leq 1 \tag{5}$$

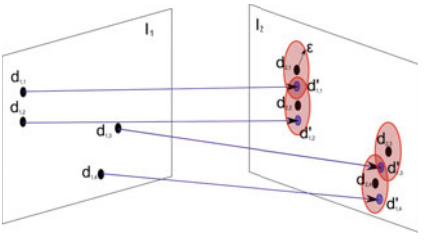
2.2 Issue 1: Implicit Assumption About Biunique Point-to-Point Correspondence

Above (or similar) definitions are in [10,11,8,6,12]. Some publications observe but do not define point-to-point correspondence.

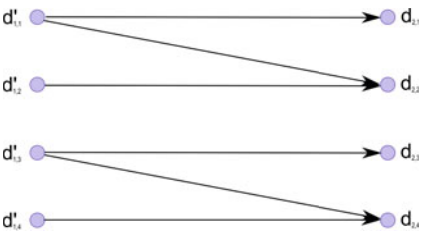
One can see that (5) is true only if cardinality of set $|D_{i,j}(\epsilon)|$ is less of minimum cardinality of the sets: $\min(|\{d_i\}|, |\{d_j\}|)$. So, authors implicitly assume that d_j and d_i should be unique in $D_{i,j}(\epsilon)$ set ($1:1$ or point-to-point correspondence). Otherwise in the $M:N$ case of points correspondence the number of (d_j, d_i) pairs of $D_{i,j}(\epsilon)$ set can be greater then any cardinality of d_j and d_i that can produce "probability" $r_{i,j}(\epsilon) > 1$.

Condition (5) is not enough to select point-to-point subset from all possible $M:N$ some-points-to-some-points $D_{i,j}(\epsilon)$ set, see Fig.1. This set can produce four different $1:1$ subsets with different (from 2 to 4) cardinalities, where the choice of some graph edge can cancel other possible choices (see Fig.2).

It is computationally expensive to support complex algorithms for maximization of the cardinality of the final point-to-point subset. The point-to-point correspondence is unpredictable in the general case.

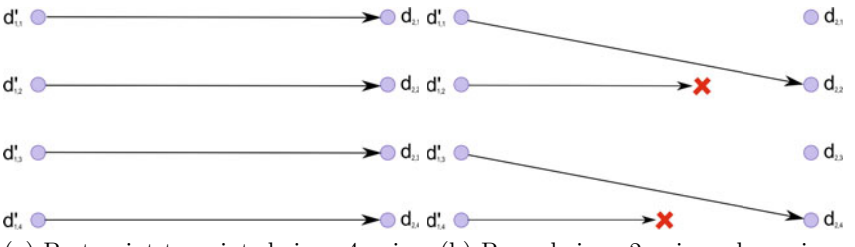


(a) Two sets of points and its possible correspondence $D_{1,2}(\epsilon)$.



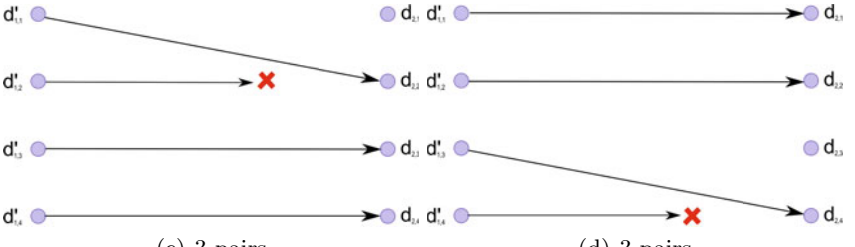
(b) Bipartite graph of all possible correspondent pairs

Fig. 1. Sample of correspondence that is based only on ϵ-selection



(a) Best point-to-point choice - 4 pairs

(b) Poor choice - 2 pairs only - min - max



(c) 3 pairs

(d) 3 pairs

Fig. 2. Point-to-point selection problem

2.3 Issue 2: Noisy Inputs Produce High Scores

The following example demonstrates the issue with using the repeatability score definition as a probability of points' correspondence.

Let I_1 be an image with noise that produce N detected interest points, I_2 is an image with single robust point and homography $H_{2,1}$ is the identity in the mathematical sense ("complex illumination variation" [10,11] only). If some point (robust or noise) detected in I_1 corresponds by (3) to a single robust point of I_2, then repeatability rate by (4) is always 1 regardless number of missed (non-repeatable) points of I_1.

If repeatability "is the percentage of the total observed points which are repeated between two images" then the expected number of such points should

be N, but we can see that cardinality of $D_{i,j}(\epsilon)$ (3) in our case is only 1 (single correspondence).

This example illustrates how noisy images can produce higher repeatability scores and produce inaccurate measurements of detector robustness. This is due to the inherent nature of definition (4) which is based on the minimum of cardinality of compared sets.

Unfortunately the *min*-based definition is wide spread not just for repeatability score for interest points [10,11], but for regions too [6,12].

2.4 Issue 3: Repeatability Score Depends on Scale Factor

The next issue is related to (3), where ϵ is fixed and independent from a homography.

Most of authors consider that two points x_a and x_b correspond to each other if [8]:

1. The error in relative point location is less than $\epsilon = 1.5$ pixels [10,11,6,12]:
$$\|x_b - H \cdot x_a\| < 1.5 \tag{6}$$
where H is the homography between the images.
2. The error in the image surface covered by point neighborhoods is $\epsilon_S < 0.4$ [8]. In the case of scale invariant points the surface error is:
$$\epsilon_S = 1 - s^2 \cdot \frac{\min(\sigma_a^2, \sigma_b^2)}{\max(\sigma_a^2, \sigma_b^2)} \tag{7}$$
where σ_a and σ_b are the selected point scales and s is the actual scale factor recovered from the homography between the images ($s > 1$).

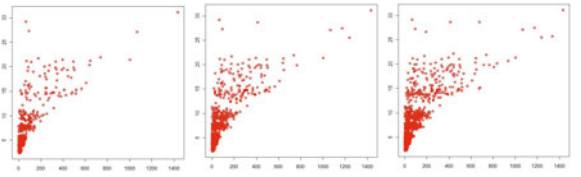
Measurements of repeatability score for scale changes are described in the set of publications [10,11,7,3]. Let the interest point location detection error in first image I_i be δ and the homography scale factor is $k > 1$ the distortion of position $p_j = H_{j,i} \cdot p_i$ in second image I_j is multiplied to k too. The final error is $k \cdot \delta$. If $k \cdot \delta \geq \epsilon$ then condition (3) is not satisfied and the repeatability score is (calculated for fixed ϵ) decreasing with increasing scale factor.

In the common case (3) and (6) are not symmetric. If the upscale factor from the first image to the second image $k_{i,j} > 1$, the downscale factor from the second image to the first image $k_{j,i} < 1$ and the repeatability score is relative to the transform direction.

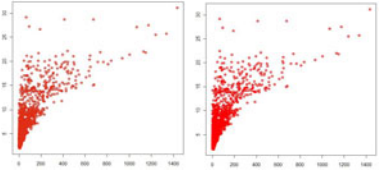
As one can see from the Fig.3 error depends upon the features' scale, so predefined fixed ϵ in the original method affects accuracy (especially for large scales). To address this, the contribution of errors to the final score should be weighed proportionally to the feature scales.

2.5 Alternative Repeatability Score Definitions

In [7] the authors propose a modified definition of repeatability score as the "ratio between the number of point-to-point correspondences that can be established for detected points and the mean number of points detected in two images":



(a) Images 1 → 2 (b) Images 1 → 3 (c) Images 1 → 4



(d) Images 1 → 5 (e) Images 1 → 6

Fig. 3. "Graffiti" dataset [1], SURF detector [3]: Dependency between squared error (horizontal axis) and feature scale (vertical axis)

$$r_{1,2} = \frac{C(I_1, I_2)}{\text{mean}(m_1, m_2)} \tag{8}$$

where $C(I_1, I_2)$ "denotes the number of corresponding couples" and m_1, m_2 "the numbers of detected points in the images".

The "mean" operator in the definition (8) hides the real meaning of the definition. The current definition assumes that in each pair (point-to-point correspondence) there are always 2 points. Let $\{d'_i\}$ and $\{d'_j\}$ are members of corresponding pairs, each with equal (point-to-point) cardinality be

$$C(I_1, I_2) = |\{d'_i\}| = |\{d'_j\}| \tag{9}$$

The total number of corresponding points in both sets is $|\{d'_i\} \cup \{d'_j\}| = |\{d'_i\}| + |\{d'_j\}| = 2C(I_1, I_2)$ The total number of points in overlapped region (2) of both images is $|\{d_i\} \cup \{d_j\}| = |\{d_i\}| + |\{d_j\}|$. The ratio of the total number of corresponding points in both sets to the total number of points in both sets is the repeatability score

$$r_{i,j} = \frac{|\{d'_i\}| + |\{d'_j\}|}{|\{d_i\}| + |\{d_j\}|} \tag{10}$$

With (9)

$$r_{i,j} = \frac{|\{d'_i\}| + |\{d'_j\}|}{|\{d_i\}| + |\{d_j\}|} = \frac{2C(I_1, I_2)}{|\{d_i\}| + |\{d_j\}|} \tag{11}$$

where

$$\frac{2}{|\{d_i\}| + |\{d_j\}|} = \frac{1}{\text{mean}(|\{d_i\}|, |\{d_j\}|)} \tag{12}$$

Criteria (10) defines the repeatability score not as a property of some detected set of image points relative to other set, but as integral property of both sets.

Note, that (10) is suitable for any type of correspondences: 1:1, 1:N, M:N. Estimations of repeatability score in this form of the definition are free from dominance of a robust but small compared set.

2.6 Remaining Issues of Repeatability Score Definition

The alternative definition fixes Issue 2 for high scores when a noisy image compared against a robust image. As one can see, issues 1 and 3 listed above still hold true for the alternative definition in [7]:

1. Repeatability score is not symmetric against direction of homography. Homography has a significant influence on the repeatability score. Distance (or scale difference) measurements are calculated in metrics of second image only and $r_{i,j} \neq r_{j,i}$.
2. Point-to-point selection is dependent upon the order. The point-to-point selection choice (see Fig.1) can produce unstable sets of pairs. For small scale factors of homography ($<<$ 1, downscaling) point position errors can be comparable to the position difference between points that does not allow point-to-points selection on some formal regular basis (minimum of distance, for example). Some points after downscaling can overlap each other.

3 Proposed Modifications toward More Accurate Repeatability Score Calculation

3.1 Complex Shape of the ϵ-Neighborhood Region

If the distance threshold in the second image is fixed, then interest-points which may be repeatable for transformations without significant scale changes can decrease the repeatability score for transformations that significantly increase scale from the first image to the second, or, on the contrary, can show higher repeatability score for transformations which significantly reduce scale. Estimation of the repeatability score is dependent upon the homography direction and are not accurate.

Let the image transformation be the identity (brightness, contrast, noise, etc. non-coordinate distortions are only applied). The distribution of the measured errors in the interest points' coordinates (relative to the "ideal" interest points) is truncated on some level and can be called as δ-neighborhood: acceptable detector error.

The joint ϵ-neighborhood error region of such identity-transformed images is a cross-coorelation function (convolution, if Hermitian) of the δ-neighborhoods (wider, than the δ-neighborhood of each separate image).

The exact method of the above convolution can vary. For example, it can be based on the following error estimation methods:

1. max of errors
2. simple sum of errors

3. error of sum of independent values (square root of the errors sum squared)
4. cross-correlation of errors distribution

The correspondence of points in this model is symmetric. For non-identity transformations the interest point and its δ-neighborhood should be transformed together. In general, the ϵ-neighborhood region is not uniform (and can't be described by ϵ-distance only, as in [8]) and is not the property of the second image only, but is dependent on the transformation, too. The final ϵ-neighborhood in the second image coordinates is a convolution of the transformed first image interest point δ-neighborhood region with the second image (uniform) δ-neighborhood region.

For example, the affine transform can produce an elliptical error distribution shape. More complex transforms can create non-uniform interest point coordinate-dependent error distributions.

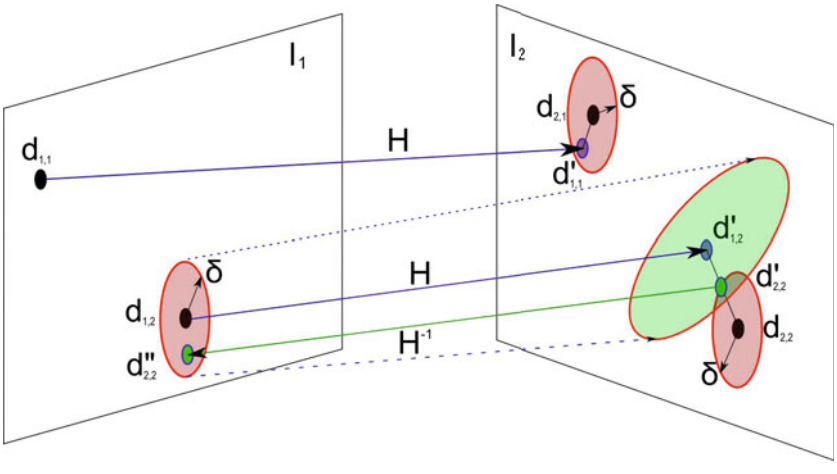


Fig. 4. Sample of points correspondence for complex type of homography

Instead of ϵ-neighborhood criteria (3) for corresponding points selection, the following algorithm 1 can be used, for example (see Fig.4): Let $d'_{1,1} = H \cdot d_{1,1}$ (first case) and $d'_{1,2} = H \cdot d_{1,2}$ (second case).

The methods above are suitable only for models with equal significance of position errors. As demonstrated in Fig.3 errors' significance depends on scale. Correspondence measure for points $p_1 = (x_1, y_1, \sigma_1)$ and $p'_2 = (x'_2, y'_2, \sigma'_2)$ (where $p'_2 = (x'_2, y'_2, \sigma'_2)$ is point of the second image $p_2 = (x_2, y_2, \sigma_2)$ transformed into scale-space coordinates of the first image) can be defined as cross-correlation of Gaussians:

$$\frac{2}{\frac{\sigma_1}{\sigma'_2} + \frac{\sigma'_2}{\sigma_1}} \exp\left(-\frac{1}{2}\frac{(x_1 - x'_2)^2 + (y_1 - y'_2)^2}{\sigma_1^2 + \sigma'^2_2}\right) \tag{13}$$

Algorithm 1. Points correspondence testing

Require: points $d_1 \in I_1$, $d_2 \in I_2$
Ensure: d_1 and d_2 corresponding status
1: $d_1' = H \cdot d_1$
2: **if** $\|d_1' - d_2\| < \delta$ **then**
3: d_1 corresponds to d_2
4: **else**
5: find point d_2' between points d_2 and d_1' with offset δ from the point d_2:

$$d_2' = d_2 + \delta \frac{d_1' - d_2}{\|d_1' - d_2\|}$$

6: apply inverse homography H^{-1} to it:

$$d_2'' = H^{-1} \cdot d_2'$$

7: **if** $\|d_2'' - d_1\| < \delta$ **then**
8: points d_1 and d_2 correspond
9: **else**
10: points d_1 and d_2 do not correspond
11: **end if**
12: **end if**

The points correspond when the correspondence measure (13) is above a threshold. For example, threshold for equation (7) is near 0.968. Using this threshold one can also find scale in (6): 4.18 pixels.

For simple cases (uniform transforms) ϵ-neighborhood criteria (3) is allowed with $\epsilon = \delta(1 + k)$, where k is scale factor of the homography. Appendix "A.2 Gaussian Identities" in [9] can help to deduce (13) for more complex cases of homography.

3.2 New "Correspondence" Assumptions and Two Types of Repeatability Score

Our approach to mitigating issues 1–3 is to use a definition closer to "the percentage of detected points which are repeated" [10] instead of "the ratio between the number of point-to-point correspondences and the minimum number of points detected in the images" [8]. or "ratio between the number of point-to-point correspondences that can be established for detected points and the mean number of points detected in two images" [11].

We propose to define two different types of repeatability scores:

1. Integral repeatability score we define as "the ratio between the number of points in both images that have corresponding points in the other image and the total number of points in the both images" (see (10)).

2. Single-side repeatability score we define as "the ratio between the number of points in the first image that have corresponding points in the second image and the total number of points in the first image" (in terms of (10)):

$$ssr_{i,j} = \frac{|\{d'_i\}|}{|\{d_i\}|} \tag{14}$$

where $\{d_i\}$ is a set of all points that satisfy (2) and $\{d'_i\}$ is a subset of a set $\{d_i\}$ where corresponding ϵ-neighborhood region in image I_j is not empty.

Sets of points being compared should satisfy (2) (points which could potentially be detected in both images). The correspondence criteria uses the complex shape of the ϵ-neighborhood region defined above. Both definitions do not assume point-to-point correspondences, but rather existence or absence of points in the ϵ-neighborhood. The repeatability score result using the new definition is always in the range $[0, 1]$.

The first definition is similar to the definition (10) (except the absence of the strict point-to-point correspondence requirement).

The repeatability score in integral form is a measure of the mutual repeatability of point sets in both images and can be used as an image similarity measure.

The single-side repeatability score is a measure of "relative" points robustness (points in the first image to the points in the second image). It can also be used as a selection criteria for detector tuning between robustness and sensitivity, for example.

4 Conclusion

The repeatability score in its original definitions has some major weaknesses, which could produce unreliable estimations of the feature detectors' performance. In this work, we proposed a modified definition of this measure, which addresses these drawbacks. The proposed method is also less dependent on the homography parameters.

Further research will be dedicated to the development of the integral measure, which will combine both scale and spatial "distances" and other, detector-specific attributes of interest points.

References

1. Affine covariant regions datasets, `http://www.robots.ox.ac.uk/~vgg/data/data-aff.html`, visual Geometry Group (University of Oxford)
2. Abdel-Hakim, A.E., Farag, A.A.: CSIFT: a SIFT descriptor with color invariant characteristics. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 1978–1983 (2006)
3. Bay, H., Tuytelaars, T., Van Gool, L.: SURF: Speeded up robust features. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3951, pp. 404–417. Springer, Heidelberg (2006)

4. Brown, M., Lowe, D.G.: Invariant features from interest point groups. In: British Machine Vision Conference, pp. 656–665 (2002)
5. Burghouts, G.J., Geusebroek, J.-M.: Performance evaluation of local colour invariants. Computer Vision and Image Understanding 113(1), 48–62 (2009)
6. Fraundorfer, F., Bischof, H.: A novel performance evaluation method of local detectors on non-planar scenes. In: 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, p. 33 (2005)
7. Mikolajczyk, K., Schmid, C.: Indexing based on scale invariant interest points. In: Proceedings Eighth IEEE International Conference on Computer Vision, ICCV 2001, Vancouver, BC, Canada, pp. 525–531 (2001)
8. Mikolajczyk, K., Schmid, C.: Scale & affine invariant interest point detectors. International Journal of Computer Vision 60(1), 63–86 (2004)
9. Rasmussen, C.E., Williams, C.K.I.: Appendix A Mathematical Background. In: Gaussian Processes for Machine Learning. MIT Press, Cambridge (2006)
10. Schmid, C., Mohr, R., Bauckhage, C.: Comparing and evaluating interest points. In: Proceedings of the Sixth International Conference on Computer Vision, Bombay, India, pp. 230–235 (January 1998)
11. Schmid, C., Mohr, R., Bauckhage, C.: Evaluation of interest point detectors. International Journal of Computer Vision 37(2), 151–172 (2000)
12. Trujillo, L., Olague, G.: Automated design of image operators that detect interest points. Evolutionary Computation 16(4) (2008)

Evaluation of Local Descriptors for Action Recognition in Videos

Piotr Bilinski and Francois Bremond

INRIA Sophia Antipolis - PULSAR group
2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex, France
firstname.surname@inria.fr

Abstract. Recently, local descriptors have drawn a lot of attention as a representation method for action recognition. They are able to capture appearance and motion. They are robust to viewpoint and scale changes. They are easy to implement and quick to calculate. Moreover, they have shown to obtain good performance for action classification in videos. Over the last years, many different local spatio-temporal descriptors have been proposed. They are usually tested on different datasets and using different experimental methods. Moreover, experiments are done making assumptions that do not allow to fully evaluate descriptors. In this paper, we present a full evaluation of local spatio-temporal descriptors for action recognition in videos. Four widely used in state-of-the-art approaches descriptors and four video datasets were chosen. HOG, HOF, HOG-HOF and HOG3D were tested under a framework based on the bag-of-words model and Support Vector Machines.

1 Introduction

In last years, many researchers have been working on developing effective descriptors to recognize objects, scenes and human actions. Many suggested descriptors have proven to establish very good performance for action classification in videos. They are able to capture appearance and motion. They are robust to viewpoint and scale changes. Moreover, they are easy to implement and quick to calculate. For example [15] proposed Scale-Invariant Feature Transform (SIFT) descriptor, [1] proposed Speeded Up Robust Features (SURF), [3] proposed Histogram of Oriented Gradients (HOG) descriptor. [8] proposed PCA-SIFT, [4] proposed Cuboid descriptor, [13] proposed Histogram of Oriented Gradients (HOG) and Histogram of Optical Flow (HOF) descriptors computed on spatio-temporal grids. [9] proposed a Spatio-Temporal Descriptor based on 3D Gradients (HOG3D). Although much has been done, it is not clear which descriptors are better than the others. They are usually evaluated on different datasets and using different experimental methods. Moreover, existing comparisons usually involve smaller or bigger restrictions. For example, [21] to limit the complexity choose a subset of $100,000$ selected training features what is around 21% of all HOG-HOF descriptors and around 40% of all HOG3D descriptors computed for

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 61–70, 2011.
© Springer-Verlag Berlin Heidelberg 2011

the KTH Dataset. Moreover, the authors make all the experiments using only one codebook size (4000). Also in [19] the authors make an assumption about the codebook size (one codebook size) and evaluate descriptors on one dataset.

In this paper, we present an evaluation of local spatio-temporal features for action recognition in videos. Four widely used in the state-of-the-art approaches descriptors were chosen: HOG, HOF, HOG-HOF and HOG3D. All these descriptors are tested on the same datasets with the same split of training and testing data, and using the same identical classification method. Our evaluation framework is based on the bag-of-words approach, an approach that is very often used together with local features. Computed descriptors are quantized into visual words and videos are represented as histograms of occurrences of visual words. For action classification, non-linear Support Vector Machines (SVM) together with leave-one-out cross-validation technique are used. Our experiments are performed on several public datasets containing both low and high resolution videos recorded using static and moving camera (KTH Dataset, Weizmann Action Dataset, ADL Dataset and Keck Dataset). In contrast to other evaluations, we test all the computed descriptors, we perform evaluation on several differing in difficulty datasets and perform evaluation on several codebook sizes. We demonstrate that accuracy of evaluated descriptors depends on the codebook size and a dataset.

The paper is organized as follows. In section 2, we briefly present the main idea of our evaluation framework. Section 3, presents our experiments and obtained results. Finally, in section 4, we present our conclusion.

2 Evaluation Framework

Our evaluation framework is as follows. In the first step, for each video, local space-time detector is applied. For each obtained point, local space-time descriptor is computed (Section 2.1). In the second step, the bag-of-words model is used to represent actions (Section 2.2). For each video, four different codebooks and four different video representations are computed. Finally, to evaluated descriptors, the leave-one-person-out cross-validation technique and non-linear multi-class Support Vector Machine are applied (Section 2.3 and 2.4). To speed-up the evaluation process, clusters of computers are used.

2.1 Space-Time Local Features

Local spatio-temporal features are extracted for each video. As a local feature detector, the Harris3D algorithm is applied. Then, for each detected feature, four types of descriptors are computed (HOG, HOF, HOG-HOF and HOG3D). The detector and descriptors were selected based on their use in the literature and availability of the original implementation[1,2]. For each algorithm, the default values of parameters were used.

[1] http://www.irisa.fr/vista/Equipe/People/Laptev/download.html
[2] http://lear.inrialpes.fr/people/klaeser/software_3d_video_descriptor/

Harris3D [11] detector - it is proposed by Laptev and Linderberg extension of the Harris corner detector [6]. The authors propose to extend the notion of spatial interest and detect local structures in space-time where the image values have significant local variations in both space and time. The authors use independent spatial and temporal scale values, a separable Gaussian smoothing function, and space-time gradients.

Histogram of Oriented Gradients (HOG) [13] - it is a 72-bins descriptor describing the local appearance. The authors propose to define a grid $n_x \times n_y \times n_t$ (default settings: $3 \times 3 \times 2$) in the surrounding space-time area and compute for each cell of the grid 4-bins histogram of oriented gradients.

Histogram of Optical Flow (HOF) [13] - it is a 90-bins descriptor describing the local motion. The authors propose to define a grid $n_x \times n_y \times n_t$ (default settings: $3 \times 3 \times 2$) around the encompassing space-time area and compute for each cell of the grid 5-bins histogram of optical flow.

HOG-HOF descriptor - it is a 162-bin descriptor combining both Histogram of Oriented Gradients and Histogram of Oriented Flow descriptors.

Spatio-Temporal Descriptor based on 3D Gradients (HOG3D) - it is a 300-bins descriptor proposed by Klaser et al. [9]. It is based on orientation histograms of 3D gradients. The authors propose to define a grid $n_x \times n_y \times n_t$ (default settings: $2 \times 2 \times 5$) in the surrounding space-time area and compute for each cell of the grid 3D gradients orientations.

2.2 Bag-of-Words Model

To represent videos using local features we apply common bag-of-words model. All computed descriptors for all Harris3D detected points are used in the quantization process. First of all, the k-means clustering algorithm with the Euclidean distance is used to create a codebook. Then, each video is represented as a histogram of occurrences of the codebook elements. In our experiments we use four different sizes of codebooks (1000, 2000, 3000 and 4000).

2.3 Classification

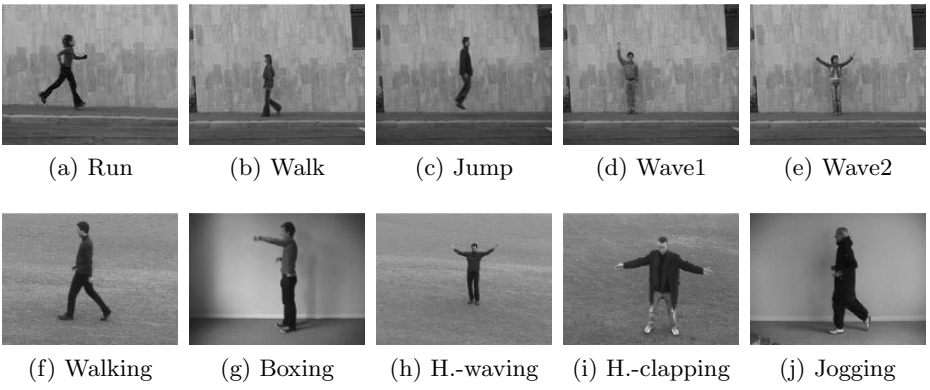
In order to perform classification, we use a multi-class non-linear Support Vector Machines using radial basis function defined by:

$$K(H_a, H_b) = exp(-\gamma D(H_a, H_b)) \tag{1}$$

where both $H_a = \{h_{a1}, ..., h_{an}\}$ and $H_b = \{h_{b1}, ..., h_{bn}\}$ are n-bins histograms. Function D is a χ^2 distance function defined by:

$$D(H_a, H_b) = \sum_{i=1}^{n} \frac{(h_{ai} - h_{bi})^2}{h_{ai} + h_{bi}} \tag{2}$$

Such defined kernel requires two parameters: (a) trade-off between training error and margin, and (b) parameter gamma in the rbf kernel. To evaluate selected descriptors, we test: (a) all values 2^x where x is in range -5 to 16 (22 different



(a) Run (b) Walk (c) Jump (d) Wave1 (e) Wave2

(f) Walking (g) Boxing (h) H.-waving (i) H.-clapping (j) Jogging

Fig. 1. A few sample frames from video sequences from Weizmann (the first row) and KTH (the second row) datasets

values) for the trade-off between training error and margin, and (b) all values 2^y where y is in range 3 to -15 (19 different values) for the parameter gamma in the rbf kernel. As a Support Vector Machine, the SVM multi-class [20] is used (multi-class variant of SVM light [7]). To speed-up the evaluation process, clusters of computers are used.

2.4 Evaluation

To evaluate selected descriptors, we use leave-one-person-out cross-validation technique (unless specified for a dataset), where videos of one actor are used as the validation data and videos from the remaining actors as the training data. This is repeated in such a way that videos from one person are used exactly once as the validation data. In our experiments we use all the descriptors calculated for all detected points to comprehensively try out the effectiveness of used local feature descriptors.

3 Experiments

Our experiments are performed on four different datasets: Weizmann Action Recognition Dataset (Section 3.1), KTH Dataset (Section 3.2), ADL Dataset (Section 3.3) and Keck Dataset (Section 3.4). A few sample frames from these video datasets can be found in Figure 1 and Figure 2. These datasets contain various types of videos: low and high resolution videos, recorded using static and moving camera, and containing one and many people. Information about these databases are summarized in Table 1.

3.1 Weizmann Action Recognition Dataset

The Weizmann Action Recognition Dataset [2,5][3] is a low-resolution (180×144 pixel resolution, 50 fps) dataset of natural human actions. The dataset contains

[3] http://www.wisdom.weizmann.ac.il/~vision/SpaceTimeActions.html

Table 1. Statistics for the Weizmann, KTH, ADL and Keck datasets: number of videos, frames, detected points, HOG-HOF (HOG, HOF) descriptors, HOG3D descriptors, frames per video, detected points per video, HOG-HOF descriptors per video, HOG3D descriptors per video, points per frame, HOG-HOF descriptors per frame, HOG3D descriptors per frame, ratio of number of HOG-HOF descriptors to number of points, and ratio of number of HOG3D descriptors to number of points

	Weizmann	KTH	ADL	Keck
resolution	180×144	160×120	640×360	640×480
#videos	93	599	150	98
#frames	6,108	289,715	72,729	25,457
#Harris3D points	13,259	473,908	718,440	227,310
#HOG-HOF descriptors	13,259	473,908	718,440	227,310
#HOG3D descriptors	9,116	252,014	690,907	207,655
#frames/#videos	65.68	483.66	484.86	259.77
#points/#videos	142.57	791.17	4789.60	2319.49
#HOG-HOF/#videos	142.57	791.17	4789.60	2319.49
#HOG3D/#videos	98.02	505.04	4606.05	2118.93
#points/#frames	2.17	1.64	9.88	8.93
#HOG-HOF/#frames	2.17	1.64	9.88	8.93
#HOG3D/#frames	1.49	0.87	9.50	8.16
#HOG-HOF/#points	1.00	1.00	1.00	1.00
#HOG3D/#points	0.69	0.53	0.96	0.91

93 video sequences showing 9 different people. The dataset contains 10 actions. The full list of actions is: run, walk, skip, jumping-jack (shortly jack), jump-forward-on-two-legs (shortly jump), jump-in-place-on-two-legs (shortly pjump), gallop-sideways (shortly side), wave-two-hands (shortly wave2), wave-one-hand (shortly wave1), and bend. Statistics about this dataset are available in table 1. Evaluation is done using leave-one-person-out cross-validation technique.

Results are presented in table 2. As we can observe, all the descriptors obtain the same accuracy for codebook 2000 and 4000. In this case, the codebook of size 2000 is preferred (faster codebook computation and faster SVM classification). The HOG descriptor performs the best for codebook 2000, the HOF descriptor for codebook 3000, HOG-HOF for codebook 2000 and HOG3D descriptor for codebook 3000. According to the results, the HOG-HOF is the best descriptor for the Weizmann dataset and the HOG descriptor is the worst. Ranking is: HOG-HOF $>$ HOF $=$ HOG3D $>$ HOG. The HOF descriptor obtains the same classification accuracy as HOG3D descriptor but HOF descriptor is smaller in size (90-bins descriptor instead of 300-bins). It takes less time to compute codebook and perform classification for the HOF descriptor. Kläser [10], employing random sampling on training features for codebook generation (codebook size 4000), obtained 75.3% accuracy for the HOG descriptor, 88.8% for the HOF descriptor, 85.6% for the HOG-HOF and 90.7% for the HOG3D descriptor. This shows that the codebook selection method has significant importance to the effectiveness of the BOW method (we obtained up to 10.72% better results).

(a) LookNum. (b) DrinkWater (c) Ans.Phone (d) Write (e) EatSnack

(f) Gesture1 (g) Gesture2 (h) Gesture3 (i) Gesture4 (j) Gesture5

Fig. 2. A few sample frames from video sequences from ADL (the first row) and Keck (the second row) datasets

Table 2. Action recognition accuracy for the Weizmann dataset

	HOG	HOF	HOG-HOF	HOG3D
codebook size 1000	83.87%	88.17%	91.40%	89.25%
codebook size 2000	**86.02%**	90.32%	**92.47%**	90.32%
codebook size 3000	86.02%	**91.40%**	91.40%	**91.40%**
codebook size 4000	86.02%	90.32%	92.47%	90.32%

3.2 KTH Dataset

The KTH dataset [18][4] contains six types of human actions (walking, jogging, running, boxing, hand waving and hand clapping) performed several times by 25 subjects in four different scenarios: outdoors, outdoors with scale variation, outdoors with different clothes and indoors. The dataset contains 599 videos. All videos were taken over homogeneous backgrounds with a static camera with 25 fps. The sequences were down-sampled by the authors to the spatial resolution of 160×120 pixels. For this dataset, as it is recommended by the authors, we divide the dataset into testing part (person 2, 3, 5, 6, 7, 8, 9, 10 and 22) and training part (other video sequences). Statistics about this dataset are available in the table 1.

Results are presented in table 3. Both HOG and HOF descriptors perform the best for codebook 1000 and both HOG-HOF and HOG3D descriptors for codebook 3000. According to the results, the HOF descriptor is superior descriptor for the KTH dataset and again the HOG descriptor is inferior quality. Ranking is: HOF > HOG-HOF > HOG3D > HOG. Wang et al. [21], choosing a subset of $100,000$ selected training features and using codebook size of 4000, obtained 80.9% accuracy for the HOG descriptor, 92.1% for the HOF descriptor, 91.8% for the HOG-HOF and 89% for the HOG3D descriptor. We obtain up to 4.52% better results on this dataset. Selecting only a subset of descriptors can cause loss of some important information.

[4] http://www.nada.kth.se/cvap/actions/

Table 3. Action recognition accuracy for the KTH dataset

	HOG	HOF	HOG-HOF	HOG3D
codebook size 1000	**83.33%**	**95.37%**	93.06%	91.66%
codebook size 2000	83.33%	94.44%	93.98%	92.13%
codebook size 3000	83.33%	94.91%	**94.44%**	**93.52%**
codebook size 4000	82.41%	94.91%	93.98%	93.06%

Table 4. Action recognition accuracy for the ADL dataset

	HOG	HOF	HOG-HOF	HOG3D
codebook size 1000	85.33%	**90.00%**	**94.67%**	**92.00%**
codebook size 2000	**88.67%**	90.00%	92.67%	91.33%
codebook size 3000	83.33%	89.33%	94.00%	90.67%
codebook size 4000	86.67%	89.33%	94.00%	85.00%

3.3 ADL Dataset

The University of Rochester Activities of Daily Living (ADL) dataset [16][5] is a high-resolution (1280×720 pixel resolution, 30 fps) video dataset of activities of daily living. The dataset contains 150 video sequences showing five different people. The dataset contains ten activities. The full list of activities is: answering a phone, dialling a phone, looking up a phone number in a telephone directory, writing a phone number on a whiteboard, drinking a glass of water, eating snack chips, peeling a banana, eating a banana, chopping a banana, and eating food with silverware. These activities were selected to be difficult to separate on the basis of single source of information (e.g. eating banana and eating snack or answering a phone and dialling a phone). These activities were each performed three times by five people differing in shapes, sizes, genders, and ethnicity. The videos were down-sampled to the spatial resolution of 640×360 pixels. Statistics about this dataset are available in table 1. Evaluation is done using leave-one-person-out cross-validation technique.

Results are presented in table 4. As we can observe, apart from the HOG descriptor, all the other descriptors perform the best for codebook 1000. The HOG descriptor performs the best for codebook 2000. According to the results, the HOG-HOF is the best descriptor for the ADL dataset and the HOG descriptor is again the worst. Ranking is: HOG-HOF > HOG3D > HOF > HOG.

3.4 Keck Dataset

The Keck gesture dataset [14][6] consists of 14 different gesture classes, which are a subset of military signals. The full list of activities is: Turn left, Turn

[5] http://www.cs.rochester.edu/~rmessing/uradl/

[6] http://www.umiacs.umd.edu/~zhuolin/Keckgesturedataset.html

Table 5. Action recognition accuracy for the Keck dataset

	HOG	HOF	HOG-HOF	HOG3D
codebook size 1000	42.86%	30.36%	37.50%	50.00%
codebook size 2000	39.29%	33.93%	**46.43%**	50.00%
codebook size 3000	**44.64%**	37.50%	39.29%	**53.57%**
codebook size 4000	41.07%	**42.86%**	44.64%	44.64%

right, Attention left, Attention right, Attention both, Stop left, Stop right, Stop both, Flap, Start, Go back, Close distance, Speed up, Come near. The dataset is collected using a color camera with 640×480 resolution. Each gesture is performed by 3 people. In each sequence, the same gesture is repeated 3 times by each person. Hence there are $3 \times 3 \times 14 = 126$ video sequences for training which are captured using a fixed camera with the person viewed against a simple, static background. There are $4 \times 3 \times 14 = 168$ video sequences for testing which are captured from a moving camera and in the presence of background clutter and other moving objects. Statistics about this dataset are available in table 1.

Results are presented in table 5. The HOG descriptor performs the best for codebook 3000, the HOF descriptor for codebook 4000, the HOG-HOF for codebook 2000 and the HOG3D for codebook 3000. The HOG3D is the best descriptor for the Keck dataset and the HOF descriptor is the worst. Ranking is: HOG3D > HOG-HOF > HOG > HOF.

According to the obtained results, we observe that accuracy of descriptors depends on the codebook size (12.5% difference on the Keck dataset for the HOF descriptor, 7% difference on the ADL dataset for the HOG3D descriptor), codebook selection method (up to 10.72% better results comparing to [10] on the Weizmann dataset) and dataset (HOF descriptor obtains 95.37% on the KTH dataset but only 42.86% on the Keck dataset). Also, we observe that smaller codebook sizes (1000, 2000, 3000) are found to lead to consistently good performance across the different datasets. Due to random initialization of k-means used for codebook generation, we observe no linear relationship accuracy of codebook size.

Our experiments show that the HOG-HOF, combination of gradient and optical flow based descriptors, seems to be a good descriptor. For the Weizmann and ADL datasets, the HOG-HOF descriptor performs best and takes the second place for the KTH and Keck datasets. The HOG descriptor usually perform the worst. The accuracy of the HOF and HOG3D descriptors depends on a dataset. Also, we observe that regardless of the dataset, the HOG-HOF and HOG3D descriptors always work better than the HOG descriptor.

4 Conclusions

In this paper, we present a full evaluation of local spatio-temporal descriptors for action recognition in videos. Four widely used in state-of-the-art approaches descriptors (HOG, HOF, HOG-HOF and HOG3D) were chosen and evaluated

under the framework based on the bag-of-words approach, non-linear Support Vector Machine and leave-one-out cross-validation technique. Our experiments are performed on four public datasets (KTH Action Dataset, Weizmann Action Dataset, ADL Dataset and Keck Dataset) containing low and high resolution videos recorded by static and moving cameras. In contrast to other existing evaluations, we test all the computed descriptors, perform evaluation on several differing in difficulty datasets and perform evaluation on several codebook sizes.

Acknowledgements. This work was supported by the Région Provence-Alpes-Côte d'Azur and partly by the Sweet-Home, Video-Id, ViCoMo, Vanaheim, and Support projects. However, the views and opinions expressed herein do not necessarily reflect those of the financing institutions.

References

1. Bay, H., Ess, A., Tuytelaars, T., Gool, L.V.: SURF: Speeded Up Robust Features. In: Computer Vision and Image Understanding (2008)
2. Blank, M., Gorelick, L., Shechtman, E., Irani, M., Basri, R.: Actions as Space-Time Shapes. In: International Conference on Computer Vision (2005)
3. Dalal, N., Triggs, B.: Histograms of Oriented Gradients for Human Detection. In: IEEE Conference on Computer Vision and Pattern Recognition (2005)
4. Dollar, P., Rabaud, V., Cottrell, G., Belongie, S.: Behavior recognition via sparse spatio-temporal features. In: Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, in conjunction with ICCV 2005 (2005)
5. Gorelick, L., Blank, M., Shechtman, E., Irani, M., Basri, R.: Actions as Space-Time Shapes. Transactions on Pattern Analysis and Machine Intelligence (2007)
6. Harris, C., Stephens, M.: A Combined Corner and Edge Detector. In: Alvey Vision Conference (1988)
7. Joachims, T.: Making Large-Scale SVM Learning Practical. In: Advances in Kernel Methods - Support Vector Learning (1999)
8. Ke, Y., Sukthankar, R.: PCA-SIFT: A More Distinctive Representation for Local Image Descriptors. In: IEEE Conference on Computer Vision and Pattern Recognition (2004)
9. Kläser, A., Marszałek, M., Schmid, C.: A Spatio-Temporal Descriptor Based on 3D-Gradients. In: British Machine Vision Conference (2008)
10. Kläser, A.: Learning human actions in video. In: PhD thesis, Université de Grenoble (2010)
11. Laptev, I., Lindeberg, T.: Space-Time Interest Points. In: International Conference on Computer Vision (2003)
12. Laptev, I.: On Space-Time Interest Points. International Journal of Computer Vision (2005)
13. Laptev, I., Marszałek, M., Schmid, C., Rozenfeld, B.: Learning realistic human actions from movies. In: IEEE Conference on Computer Vision and Pattern Recognition (2008)
14. Lin, Z., Jiang, Z., Davis, L.S.: Recognizing Actions by Shape-Motion Prototype Trees. In: International Conference on Computer Vision (2009)
15. Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints. In: International Journal of Computer Vision (2004)

16. Messing, R., Pal, C., Kautz, H.: Activity recognition using the velocity histories of tracked keypoints. In: International Conference on Computer Vision (2009)
17. Rosten, E., Drummond, T.: Machine learning for high-speed corner detection. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3951, pp. 430–443. Springer, Heidelberg (2006)
18. Schuldt, C., Laptev, I., Caputo, B.: Recognizing Human Actions: A Local SVM Approach. In: International Conference on Pattern Recognition (2004)
19. Stöttinger, J., Goras, B.T., Pönitz, T., Sebe, N., Hanbury, A., Gevers, T.: Systematic Evaluation of Spatio-temporal Features on Comparative Video Challenges. In: International Workshop on Video Event Categorization, Tagging and Retrieval, in conjunction with ACCV 2010 (2010)
20. Tsochantaridis, I., Joachims, T., Hofmann, T., Altun, Y.: Large Margin Methods for Structured and Interdependent Output Variables. Journal of Machine Learning Research (2005)
21. Wang, H., Ullah, M.M., Kläser, A., Laptev, I., Schmid, C.: Evaluation of local spatio-temporal features for action recognition. In: British Machine Vision Conference (2009)
22. Willems, G., Tuytelaars, T., Van Gool, L.: An efficient dense and scale-invariant spatio-temporal interest point detector. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part II. LNCS, vol. 5303, pp. 650–663. Springer, Heidelberg (2008)

On the Spatial Extents of SIFT Descriptors for Visual Concept Detection

Markus Mühling, Ralph Ewerth, and Bernd Freisleben

Department of Mathematics & Computer Science, University of Marburg
Hans-Meerwein-Str. 3, D-35032 Marburg, Germany
{muehling,ewerth,freisleb}@informatik.uni-marburg.de

Abstract. State-of-the-art systems for visual concept detection typically rely on the Bag-of-Visual-Words representation. While several aspects of this representation have been investigated, such as keypoint sampling strategy, vocabulary size, projection method, weighting scheme or the integration of color, the impact of the spatial extents of local SIFT descriptors has not been studied in previous work. In this paper, the effect of different spatial extents in a state-of-the-art system for visual concept detection is investigated. Based on the observation that SIFT descriptors with different spatial extents yield large performance differences, we propose a concept detection system that combines feature representations for different spatial extents using multiple kernel learning. It is shown experimentally on a large set of 101 concepts from the Mediamill Challenge and on the PASCAL Visual Object Classes Challenge that these feature representations are complementary: Superior performance can be achieved on both test sets using the proposed system.

Keywords: Visual Concept Detection, Video Retrieval, SIFT, Bag-of-Words, Magnification Factor, Spatial Bin Size.

1 Introduction

Visual concept detection, also known as high-level feature extraction, plays a key role in semantic video retrieval, navigation and browsing. Query-by-content based on low-level features is insufficient to search successfully in large-scale multimedia databases [10]. Thus, several approaches in the field of image and video retrieval focus on high-level features serving as intermediate descriptions to bridge the "semantic gap" between data representation and human interpretation. Hauptmann et al. [3] has stated that less than 5000 concepts, detected with a minimum accuracy of 10% mean average precision, are sufficient to provide search results comparable to text retrieval in the World Wide Web. Due to the large visual variations in the appearance of semantic concepts, current approaches mainly focus on local keypoint features, with SIFT (scale-invariant feature transform [9]) as the most successful descriptor. These local features are usually clustered to build a visual vocabulary, where the cluster centers are regarded as "visual words". Similar to the representation of documents in the

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 71–80, 2011.
© Springer-Verlag Berlin Heidelberg 2011

field of text retrieval, an image or shot can then be represented as a Bag-of-Visual-Words (BoW) by mapping the local descriptors to the visual vocabulary. Current semantic concept detection systems rely on this BoW representation [11]. Although several variations of keypoint sampling strategies, vocabulary construction techniques, local descriptor projection methods, and machine learning algorithms have been evaluated, the impact of the spatial extents of local SIFT descriptors has not been studied for semantic concept detection in previous work.

In this paper, we investigate the impact of the spatial extents of local SIFT descriptors using a state-of-the-art visual concept detection system. We have observed that for particular semantic concepts different spatial extents cause large performance differences. Based on these observations, we propose to combine feature representations for different spatial extents using multiple kernel learning (MKL). Experimental results on the Mediamill and on the PASCAL Visual Object Classes (VOC) Challenge show that the concept detection performance can be significantly boosted by combining feature representations of different spatial sizes using MKL. In particular, it is more effective than using a spatial pyramid representation. Futhermore, the results indicate that the magnification factor of SIFT descriptors, which defines the spatial bin size depending on the keypoint scale, should be much larger for semantic concept detection than the usually used default value.

The paper is organized as follows. Section 2 discusses related work. Section 3 describes the concept detection system. Experimental results are presented in Section 4. Section 5 concludes the paper and outlines areas for future research.

2 Related Work

In recent years, researchers have shifted their attention to generic concept detection systems, since the development of specialized detectors for hundreds or thousands of concepts seems to be infeasible. Using BoW approaches, continuous progress has been reported in recent years. The top 5 official runs at the TRECVID 2010 semantic indexing task rely on BoW representations [11]. While early BoW approaches have mainly extracted local descriptors at salient points, today it seems that this representation using keypoint detectors like Harris-Laplace or DoG (Difference of Gaussians) is often insufficient to describe natural images. For scene classification, random or dense sampling strategies have outperformed the previously mentioned scale- and rotation-invariant keypoint detectors [1] [12]. Jiang et al. [4] have evaluated various factors of the BoW representation for semantic video retrieval including the choice of keypoint detector, kernel, vocabulary size and weighting scheme. Furthermore, they have proposed a soft-weighting scheme where a keypoint is assigned to multiple visual words, and the importance is determined by the similarity of the keypoint to the vocabulary entry. Another study of Jiang et al. [5] has provided a comprehensive comparison of representation choices of keypoint-based semantic concept detection. Yang et al. [17] have applied techniques used in text categorization, including term weighting, stop word removal or feature selection to generate image representations for scene classification that differ in dimension, selection, and weighting of

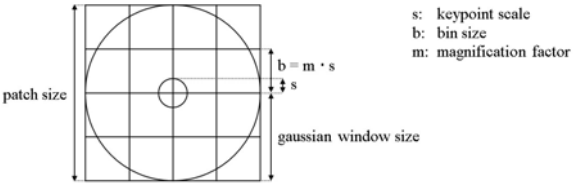


Fig. 1. SIFT descriptor geometry

visual words. Various color features, like rgb-SIFT, opponent-SIFT or hsv-SIFT have been compared by van de Sande et al. [13] for visual concept classification. Lazebnik et al. [8] have suggested spatial pyramid features for scene classification to integrate spatial information. They have concatenated BoW representations for equally sized image subregions of different partitioning levels. The different levels were fused using a weighted combination of kernels per level. It has been shown that the combination of multiple spatial layouts is helpful, whereas an image partitioning of more than 2x2 regions is ineffective [17, 13]. An example of using MKL to combine different BoW representations based on spatio-temporal features is given by Kovashka and Grauman [7] in the field of action recognition. The best semantic indexing system [11] at TRECVID 2010 has used sparse and dense sampling, multiple color SIFT descriptors, spatial pyramids, multi-frame video processing, and kernel-based machine learning.

3 Concept Detection System

In this section, we describe the proposed system for visual concept detection. The concept detection challenge is considered as a supervised learning task. Support vector machines (SVM) that have proven to be powerful for visual concept detection [11] are used for the classification of each concept. We apply two sampling strategies to extract local SIFT descriptors: sparse sampling using the DoG salient point detector and dense sampling. The visual vocabulary is generated using a K-means algorithm, and an image is then described as a histogram indicating the presence of each "visual word". Further implementation choices of the used BoW approach, such as the soft-weighting scheme, the integration of color and spatial information, and the used kernel are described below. The proposed MKL framework to combine feature representations based on different local region sizes is presented in Section 3.5.

3.1 SIFT Descriptor Geometry

The scale-invariant feature transform (SIFT) performs keypoint detection and local feature extraction. A DoG detector is used to detect the keypoints. The appearance of a keypoint is described using a spatial histogram of image gradients, where a Gaussian weighting function is applied to reduce the influence of gradients further away from the keypoint. The SIFT descriptor geometry is

specified by the number and size of the spatial bins and the number of orientation bins. Using 8 orientation bins and 4x4 spatial bins, the local descriptor results in an 128-dimensional vector. To extract SIFT features, the implementation of the Vision Lab Features Library (VLFEAT) [16] is used. It also provides a fast algorithm for the calculation of densely sampled SIFT descriptors of the same scale and orientation. We use a step size of 5 pixels for dense sampling. In the case of scale-invariant keypoints the spatial bin size is determined by the product of the detected keypoint scale and the magnification factor (see Figure 1). The default magnification factor of the SIFT implementation is 3 [9]. Since dense sampled keypoints do not have detected keypoint scales, the bin size is specified directly.

3.2 Soft-Weighting Scheme

Instead of mapping a keypoint only to its nearest neighbor, a soft-weighting scheme similar to Jiang et al. [4] is used, where the top K nearest visual words are selected. Using a visual vocabulary of N visual words, the importance of a visual word t in the image is represented by the weights of the resulting histogram bins $w = [w_1, \ldots, w_t, \ldots, w_N]$ with

$$w_t = \sum_{i=1}^{K} \sum_{j=1}^{M_i} sim(j,t), \tag{1}$$

where M_i is the number of keypoints whose i-th nearest neighbor is the word t.

3.3 Color and Spatial Information

Color information is integrated using rgb-SIFT. Therefore, the SIFT descriptors are computed independently for the three channels of the RGB color model. The final keypoint descriptor is the concatenation of the individual descriptors. Due to the normalizations during the SIFT feature extraction, rgb-SIFT is equal to the transformed color SIFT descriptor, and thus invariant against light intensity and color changes [13].

To capture the spatial image layout, we use a spatial pyramid of 1x1 and 2x2 equally sized subregions. The HoWs (histogram of words) are generated independently for each subregion and concatenated in a final feature vector. The weighting of the HoWs is realized as specified by Lazebnik et al. [8].

3.4 Kernel Choice

The kernel choice is a critical decision for the performance of a SVM. Since histogram representations are used in our approach, we apply the χ^2 kernel that is based on the corresponding histogram distance:

$$k_{\chi^2}(x,y) = e^{-\gamma \chi^2(x,y)} \quad with \quad \chi^2(x,y) = \sum_i \frac{(x_i - y_i)^2}{x_i + y_i}. \tag{2}$$

Jiang et al. [5] have used the χ^2 kernel successfully for BoW features in the context of semantic concept detection. In their study, the χ^2 kernel has outperformed the traditional linear and radial basis function kernels.

3.5 Multiple Kernel Learning

In order to combine the capabilities of feature respresentations based on different spatial extents, MKL is applied to find an optimal kernel combination

$$k = \sum_{i=1}^{n} \beta_i k_i \quad with\ \beta_i \geq 0 \tag{3}$$

where each kernel k_i takes a different feature representation into account. The sparsity of the kernel weights can be controlled by the L_p-norm. We use the l_2-norm that leads to a uniform distribution of kernel weights. Throughout our experiments, we use the MKL framework provided by the Shogun library [15] in combination with the support vector machine implementation of Joachims [6], called SVM^{light}.

4 Experimental Results

In this section, the performance impact of the spatial extents of SIFT descriptors and the combination of different spatial extents using MKL is investigated in the field of visual concept detection. For this purpose, two benchmarks are used, the Mediamill Challenge [14] and the PASCAL Visual Object Classes (VOC) Challenge [2]. The Mediamill Challenge offers a dataset based on the TRECVID 2005 [11] training set with an extensive set of 101 annotated concepts, including objects, scenes, events and personalities. It consists of 86 hours of news videos containing 43,907 completely annotated video shots. These shots are divided into a training set of 30,993 shots and a test set of 12,914 shots. For every shot, a single representative keyframe image is provided. In our experiments the positive and negative training instances per concept are each restricted to 5000 samples to speed up the training process. The PASCAL VOC Challenge

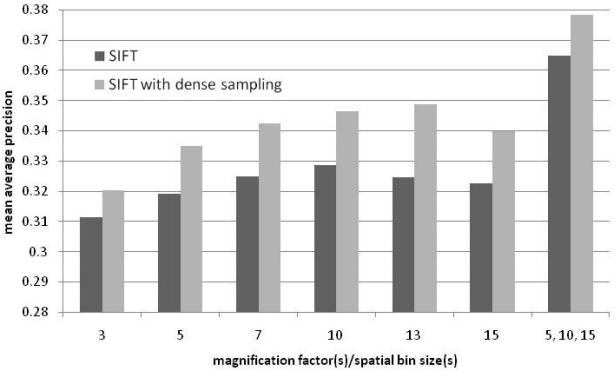


Fig. 2. Evaluation of different spatial sizes on the Mediamill Challenge using a 1000-dimensional vocabulary (averaged over 101 concepts)

provides a test set for image classification with 20 annotated object classes, e.g. "bird", "cat", "cow", "aeroplane", "bicycle", "boat", "bottle", "chair", "dining table" and "person". In total, this dataset consists of 9,963 "flickr" images, approximately equally splitted into training and test set.

4.1 Evaluation Criteria

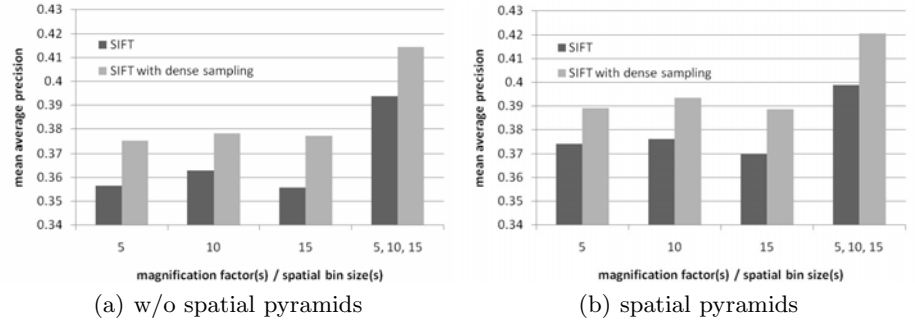
To evaluate the concept retrieval results, the quality measure of average precision (AP) is used. For each concept, the implemented system returns a list of ranked shots, which is used to calculate the average precision as follows:

$$AP(\rho) = \frac{1}{R} \sum_{k=1}^{N} \frac{\left|R \cap \rho^k\right|}{k} \psi(i_k) \tag{4}$$

where $\rho^k = i_1, i_2, \ldots, i_k$ is the ranked shot list up to rank k, N is the length of the ranked shot list, R ist the total number of relevant shots and $\left|R \cap \rho^k\right|$ is the number of relevant shots in the top k of ρ. The function $\psi(i_k) = 1$ if $i_k \in R$ and 0 otherwise. To evaluate the overall performance, the mean average precision score is calculated by taking the mean value of the average precisions for the individual concepts. Furthermore, the official partial randomization test in the TRECVID evaluation [11] is used to determine whether our system is significantly better than a reference system, or if the difference is only due to chance.

4.2 Results

Our experiments are based on visual features analysis. In all experiments, the rgb-SIFT descriptor is used since color SIFT descriptors achieve superior performance for concept detection [13]. We have conducted several experiments on the two benchmark test sets to investigate the impact of spatial bin sizes in combination with different sampling strategies (sparse and dense sampling), different vocabulary sizes (1000 and 4000 visual words) and spatial pyramids.



(a) w/o spatial pyramids (b) spatial pyramids

Fig. 3. Evaluation of different spatial sizes on the Mediamill Challenge using a vocabulary of 4000 visual words (averaged over 101 concepts)

Table 1. Average precision values of 7 selected concepts from the Mediamill Challenge for different spatial bin sizes using a vocabulary size of 4000

In [%]	sparse sampling			dense sampling		
	5	10	15	5	10	15
bicycle	0.5	1.5	**5.5**	2.4	17.6	**65.4**
beach	7.5	**15.6**	14.6	7.7	**17.1**	16.0
desert	**18.8**	17.3	15.7	**22.1**	20.5	18.0
boat	10.4	12.6	**13.1**	16.4	18.4	**20.0**
marching	**46.4**	42.9	39.9	**46.6**	38.2	33.9
tennis	**73.9**	69.8	68.7	**77.9**	69.7	62.0
court	**38.0**	35.0	36.9	34.1	38.6	**41.6**

The experimental results on the Mediamill Challenge are presented in Figures 2–3. In a first experiment, the impact of different spatial bin sizes in combination with different sampling strategies and a vocabulary size of 1000 visual words was investigated. Using a magnification factor of 10, an improvement of 5.5% was achieved compared to the default factor of 3 (see Figure 2). In case of dense sampling, the best performance was achieved using a spatial bin size of 13. We performed several runs to measure the impact of the non-deterministic K-means algorithm on the results. Using 10 iterations, the mean AP and the standard deviation for a magnification factor/spatial bin size of 10 amounts to $32.91\% \pm 0.06$ for scale-invariant keypoints and $34.6\% \pm 0.08$ in the case of dense sampling. The experiment was repeated for the magnification factors/spatial bin sizes 5, 10 and 15 with an increased vocabulary of 4000 visual words and additionally in combination with a spatial pyramid representation (see Figure 3). The spatial pyramids were constructed using a spatial grid of 1x1 and 2x2 regions. In both experiments based on salient points, the best performance was achieved using a magnification factor of 10, and the best spatial bin size for dense sampled SIFT descriptors was 10, too. Some semantic concepts yielded large performance differences for different magnification factors and spatial bin sizes, respectively. Table 1 shows these differences for selected concepts.

Furthermore, we combined the feature representations of different magnification factors/spatial bin sizes using MKL. Significant performance improvements were achieved in all experiments for both sparse and dense sampling. Using a vocabulary of 4000 visual words, the relative performance improvement was up to 10.7% in the case of sparse sampling (see Figure 3(a)) and with spatial pyramids up to 8.2% in the case of dense sampling (see Figure 3(b)). Finally, all spatial pyramid representations based on sparse and dense sampling were combined using MKL, which achieved a mean AP of 43.2%.

On the VOC Challenge different spatial bin sizes are analyzed in combination with sparse and dense sampling. In this experiments, spatial pyramids and a vocabulary size of 4000 visual words were used (see Figure 4). The best performance based on sparse sampling was achieved using a magnification factor of 10, like on the Mediamill Challenge. When dense sampling was used, the best performance was achieved for a spatial bin size of 5. In both cases, the combination of

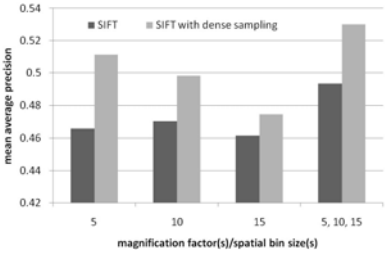


Fig. 4. Evaluation of different spatial sizes on the PASCAL VOC Challenge using spatial pyramids and a 4000-dimensional vocabulary (averaged over 20 object classes).

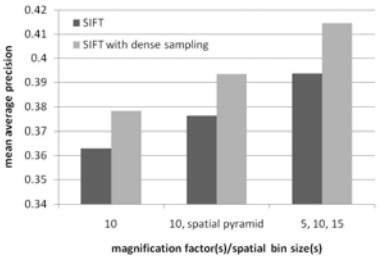


Fig. 5. Comparison of spatial pyramids vs combining different magnification factors/spatial bin sizes on the Mediamill Challenge using a 4000-dimensional vocabulary (averaged over 101 concepts).

different spatial sizes yielded significant relative performance improvements of up to 6.4% and 10.5%, respectively. The combination of all feature representations based on sparse and dense sampling achieved a mean AP of 54.1%.

4.3 Discussion

The results of our experiments show that the spatial bin size and thus the patch size of the SIFT descriptor should not be too small. The magnification factor that determines the spatial bin size depending on the keypoint scale should be chosen considerably larger than the default value of 3. In all experiments on the VOC as well as on the Mediamill Challenge, a magnification factor of 10 achieved the best detection performance. Due to the large visual variations within concept classes, larger patch sizes seem to result in a more generalizable representation. While small patch sizes only describe the near neighborhood of a keypoint, larger patch sizes describe rather coarse image structures. In the case of dense sampling, the impact of the bin size varies depending on the data set. While the concept detection results on the Mediamill Challenge also suggest larger bin sizes, the best performance on the object classification test set was already achieved using a bin size of 5. It seems that larger bin sizes are better suited for detecting scenes than for detecting objects. In general, it can be noticed that the best spatial bin size depends on the used dataset and most notably on the detected concept class.

In all experiments, the combination of different spatial bin sizes/magnification factors using MKL significantly improved the concept detection performance. These results show that the feature representations based on different spatial bin sizes are complementary.

Figure 5 depicts the performance improvement of the spatial pyramid representation versus the combination of different magnification factors/spatial bin sizes. The combination of different spatial bin sizes using MKL is more effective than the use of spatial pyramids. Using a 4000-dimensional vocabulary and a magnification factor respectively spatial bin size of 10, the performance improvement of the spatial pyramid representation was only up to 3.8%. In contrast, the

relative performance improvement of combining different spatial bin sizes was 7.8% in the case of sparse sampling and 8.7% in the case of dense sampling. While the storage complexity of the spatial pyramid representation adds up to 5 4000-dimensional histograms, the combination of 3 different spatial bin sizes yields only 3 histograms per shot.

The combination of different spatial bin sizes, different sampling strategies and spatial pyramids achieved state-of-the-art performances on the Mediamill as well as on the VOC Challenge, 43.2% and 54.1% mean AP, respectively. Considering further frames per shot on the Mediamill Challenge instead of only one keyframe, we have even obtained a mean AP of 44.6%. This is an improvement of over 100% compared to the baseline provided by the Mediamill Challenge. To the best of our knowledge, the best reported result for the same color features on this challenge is approximately 42% [13].

5 Conclusions

In this paper, we have investigated the impact of the spatial extents of SIFT descriptors for visual concept detection. It turned out that the magnification factor that determines the spatial bin size depending on the keypoint scale should be much larger than the normally used default value. Based on the observation that SIFT descriptors with different spatial extents yield large performance differences, we have proposed to combine feature representations based on different magnification factors or different spatial bin sizes, respectively, using MKL. Experimental results on the Mediamill as well as on the VOC Challenge have demonstrated that these feature representations complement each other: The concept detection performance could be significantly boosted by combining different spatial sizes of local descriptors using MKL – this was even more effective than using spatial pyramid representations. An area of future work is to automatically find an optimal combination of these sizes.

Acknowledgements. This work is supported by the German Ministry of Education and Research (BMBF, D-Grid) and by the German Research Foundation (DFG, PAK 509).

References

1. Bosch, A., Zisserman, A., Muñoz, X.: Scene Classification Using a Hybrid Generative/Discriminative Approach. IEEE Transactions on Pattern Analysis and Machine Intelligence 30(4), 712–727 (2008)
2. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: In: The PASCAL Visual Object Classes Challenge 2007, VOC 2007 (2007), `http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html`
3. Hauptmann, A., Yan, R., Lin, W.-H.: How Many High-Level Concepts Will Fill the Semantic Gap in News Video Retrieval?. In: International Conference on Image and Video Retrieval, pp. 627–634. ACM, New York (2007)

4. Jiang, Y.-G., Ngo, C.-W., Yang, J.: Towards Optimal Bag-of-Features for Object Categorization and Semantic Video Retrieval. In: International Conference on Image and Video Retrieval, pp. 494–501. ACM, New York (2007)
5. Jiang, Y.-G., Yang, J., Ngo, C.-W., Hauptmann, A.G.: Representations of Keypoint-Based Semantic Concept Detection: A Comprehensive Study. IEEE Transactions on Multimedia 12, 42–53 (2010)
6. Joachims, T.: Text Categorization With Support Vector Machines: Learning With Many Relevant Features. In: Nédellec, C., Rouveirol, C. (eds.) ECML 1998. LNCS, vol. 1398, pp. 137–142. Springer, Heidelberg (1998)
7. Kovashka, A., Grauman, K.: Learning a Hierarchy of Discriminative Space-time Neighborhood Features for Human Action Recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2046–2053 (2010)
8. Lazebnik, S., Schmid, C., Ponce, J.: Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 2169–2178. IEEE Computer Society, USA (2006)
9. Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision 60(2), 91–110 (2004)
10. Naphade, M.R., Smith, J.R.: On the Detection of Semantic Concepts at TRECVID. In: International Conference on Multimedia, pp. 660–667. ACM, USA (2004)
11. National Institute of Standards and Technology (NIST): TREC Video Retrieval Evaluation (TRECVID), `http://www-nlpir.nist.gov/projects/trecvid/`
12. Nowak, E., Jurie, F., Triggs, B.: Sampling Strategies for Bag-of-Features Image Classification. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3954, pp. 490–503. Springer, Heidelberg (2006)
13. van de Sande, K.E., Gevers, T., Snoek, C.G.: A Comparison of Color Features for Visual Concept Classification. In: International Conference on Content-Based Image and Video Retrieval, pp. 141–150. ACM, USA (2008)
14. Snoek, C.G.M., Worring, M., van Gemert, J.C., Geusebroek, J.M., Smeulders, A.W.M.: The Challenge Problem for Automated Detection of 101 Semantic Concepts in Multimedia. In: ACM International Conference on Multimedia, pp. 421–430. ACM, USA (2006)
15. Sonnenburg, S., Rätsch, G., Henschel, S., Widmer, C., Behr, J., Zien, A., Bona, F., Binder, A., Gehl, C., Franc, V.: The SHOGUN Machine Learning Toolbox. Journal of Machine Learning Research 99, 1799–1802 (2010)
16. Vedaldi, A., Fulkerson, B.: VLFeat: An Open and Portable Library of Computer Vision Algorithms (2008), `http://www.vlfeat.org/`
17. Yang, J., Jiang, Y.G., Hauptmann, A.G., Ngo, C.: Evaluating Bag-of-Visual-Words Representations in Scene Classification. In: International Workshop on Multimedia Information Retrieval, pp. 197–206. ACM, USA (2007)

An Experimental Framework for Evaluating PTZ Tracking Algorithms

Pietro Salvagnini[1], Marco Cristani[1,2], Alessio Del Bue[1], and Vittorio Murino[1,2]

[1] Istituto Italiano di Tecnologia (IIT), Genova, Italy
[2] Computer Science Department, University of Verona, Verona, Italy
{pietro.salvagnini,marco.cristani,alessio.delbue,vittorio.murino}@iit.it

Abstract. PTZ (Pan-Tilt-Zoom) cameras are powerful devices in video surveillance applications, because they offer both wide area coverage and highly detailed images in a single device. Tracking with a PTZ camera is a closed loop procedure that involves computer vision algorithms and control strategies, both crucial in developing an effective working system. In this work, we propose a novel experimental framework that allows to evaluate image tracking algorithms in controlled and repeatable scenarios, combining the PTZ camera with a calibrated projector screen on which we can play different tracking situations. We applied such setup to compare two different tracking algorithms, a kernel-based (mean-shift) tracking and a particle filter, opportunely tuned to fit with a PTZ camera. As shown in the experiments, our system allows to finely investigate pros and cons of each algorithm.

1 Introduction

This paper proposes a platform to evaluate different single-target tracking algorithms for PTZ cameras. Our aim is toward *repeatability*, which is complex in such case because PTZ cameras "see" a different scenario given the choice of pan, tilt and zoom parameters, and such parameters are differently set by the diverse tracking algorithms taken into account. This means that there cannot be a unique video benchmark which allows a genuine global testing. The proposed system provides the same scenario to the PTZ camera as many times as desired, in order to test different tracking algorithms. The core idea consists in projecting a video containing the target on a screen in front of the camera. In this way, we are aware at each instant about the position of the target on the projector screen. This setup makes possible to compare the localization error and other metrics of the target during tracking. The setup, shown in figure 1, is composed by 3 different steps: (1) camera calibration, (2)implementation of the PTZ tracking algorithm, (3) projection of the video with the target on the wall and comparison of the different tracking results with the ground-truth (GT).

The paper is organized as follow. In section 2 the state-of-the-art is presented. In section 3 the whole system will be presented, describing the 3 parts introduced above. A quantitative evaluation of the effectiveness of our framework so as the comparison between two particular tracking algorithms are provided in section

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 81–90, 2011.
© Springer-Verlag Berlin Heidelberg 2011

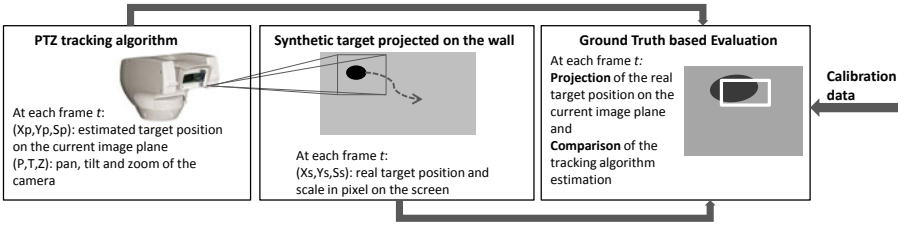


Fig. 1. The whole system

4, whereas in section 5 final considerations about the importance of our contribution and a possible future work are discussed.

2 State of the Art

PTZ cameras can be exploited and studied from different points of view. In our work, we are interested in the geometrical modeling for the PTZ camera control and target projection, as related to visual object tracking. Hence, this section presents the most relevant state of the art regarding four different aspects, PTZ camera modeling, visual object tracking in general and using a PTZ camera, and evaluation of tracking algorithms.

PTZ Geometry. The problem of calibrating a PTZ camera has been addressed in many papers with different methods and levels of approximation. One of the most important work on self-calibration of rotating and zooming camera is the paper of Agapito et al. [1]. It considers in particular how the changes of the zoom and the settings of the focus affect the intrinsic parameters. Other works are related to estimation of a geometrical model for a rotating camera, linking the rotation angles to the camera position in the 3D world. In [7], pan and tilt rotations are modeled as occurring around arbitrary camera axes in space, and the relative position between the axis is estimated.

Visual Object Tracking. An overview on different techniques can be found in [15]. The Bayesian recipe is one of the most widely used framework for tracking, that considers both an *a priori* information on the target (dynamical model), and the information from the current image acquired from the camera (observation model). The choice of the dynamical and the observation models characterizes each different algorithm together with the approximation of the evolution of the probability density function that describes the target state. Nowadays, particle filters are the most employed techniques; here we considered the classical filtering approach of Condensation [10]. A different philosophy sees the tracking as a mode seeking procedure, here represented by the Mean Shift tracker [6], and in particular by the *CamShift* approach [3]. The target model is an histogram and the area of the image that exhibits the most similar histogram is searched at each iteration. This algorithm proposes an extremely efficient technique to minimize the Bhattacharyya distance between histograms. In the last years both these

algorithms have been extended and improved, like in [4] for particle filter or [5] for mean-shift.

PTZ Cameras in Video Surveillance Systems. Typically, in video surveillance settings, a master-slave architecture is adopted using a wide zoom fixed camera (master) and a PTZ camera (slave) that is moved to highlight the relevant subjects of interest in the scene, as in [9]. When using multi-camera architectures, calibration between cameras is a key element and usually requires considerable effort. For this reason, methods that only require weak calibration or implements automatic calibration algorithms (e.g. [2]) are the most popular ones. Among them, two recent works are [13] and [14]. In the former, the scenario consists of a single PTZ camera that tracks a moving target which lies on the floor, and the focus of the work is on the control part of the process: the choice of the camera position at each step is formulated as an optimization problem. In the latter, the camera tracks the upper-body of a person that walks in a room with a fuzzy algorithm. It also compares the results obtained with other tracking algorithms, but such a comparison is performed off-line, using the frames obtained from their PTZ tracking algorithm. As a result, only the visual tracking algorithms are evaluated and not the performance of the system that also accounts for the camera motion.

Evaluation of Tracking Algorithms. The evaluation of tracking algorithms is often related to a specific application, for example surveillance in [8] or low frame rate areal imagery [12] or for a specific category of algorithms, e.g. template-based in [11]. In this work we will adopt similar metrics for the evaluation, but apply them to different, unexplored PTZ scenario.

3 Methodology

In this section, we present the different components of the whole system: the calibration of the PTZ camera, the implementation of the two tracking algorithms, and finally the performance evaluation testbed.

3.1 PTZ Camera Calibration

We adapt a standard pinhole camera model to a specific PTZ camera, shown in Fig. 1, used for our evaluation.

Intrinsic parameters. First, we calibrate the intrinsic or internal parameters, according to the pinhole model with one coefficient for the radial distortion.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_d \\ y_d \\ 1 \end{bmatrix} \quad \begin{cases} x_d = \dfrac{x_r}{z_r}(1 + kr^2) \\ \\ y_d = \dfrac{y_r}{z_r}(1 + kr^2) \end{cases} \quad r^2 = \frac{x_r^2 + y_r^2}{z_r^2} \tag{1}$$

where (x_r, y_r, z_r) are the coordinates in the optical center reference system, (x_d, y_d) are the coordinates after the distortion due to the camera lens and

(u, v) are the pixel coordinates on the image plane. The intrinsic parameters (f_x, f_y, c_x, c_y, k) are estimated for each zoom level between 1x and 20x, with a step of 1x.

Rotation Axis Model. The PTZ camera can rotate around two axes that are not aligned with the camera reference system. The rotation axes do not intersect in any points and do not pass through the optical center, so the correct misalignments should be computed to avoid approximations. Let (ϕ_C^i, θ_C^i) indicate the camera pan and tilt angles as measured by the motor encoder and (x_r^i, y_r^i, z_r^i) the coordinates of a point in the optical center reference system in that pose. When the camera is rotating from the initial pose $\phi_C^0 = 0$ and $\theta_C^0 = 0$ to a new pose (ϕ_C^1, θ_C^1), the transformation between the two reference systems can be described by the composition of two rotations, each of them around a translated axis:

$$\begin{bmatrix} x_r^0 \\ y_r^0 \\ z_r^0 \\ 1 \end{bmatrix} = R_0 T_0 \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \quad \begin{bmatrix} x_r^1 \\ y_r^1 \\ z_r^1 \\ 1 \end{bmatrix} = T_\theta^{-1} R_x(\theta_C^1) T_\theta T_\phi^{-1} R_y(\phi_C^1) T_\phi \begin{bmatrix} x_r^0 \\ y_r^0 \\ z_r^0 \\ 1 \end{bmatrix} \tag{2}$$

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c(\phi) & -s(\phi) & 0 \\ 0 & s(\phi) & c(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, R_y(\theta) = \begin{bmatrix} c(\phi) & 0 & -s(\phi) & 0 \\ 0 & 1 & 0 & 0 \\ s(\phi) & 0 & c(\phi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, T_{\phi,\theta} = \begin{bmatrix} 1 & 0 & 0 & t_x^{\phi,\theta} \\ 0 & 1 & 0 & t_y^{\phi,\theta} \\ 0 & 0 & 1 & t_z^{\phi,\theta} \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

where $s(\cdot)$ and $c(\cdot)$ stand for sine and cosine trigonometric functions. R_0 and T_0 represent the roto-translation from the screen to the camera given the initial position (ϕ_C^0, θ_C^0). Finally we indicate with (x_r^0, y_r^0, z_r^0) in the optical center reference system when in the initial position. These parameters are estimated at the beginning of each experiment, because they depend on the relative position between the camera and the screen. This is achieved by projecting a checkerboard of known size on the screen. On the contrary, the translation vectors $[t_x^\theta, t_y^\theta, t_z^\theta]$ and $[t_x^\phi, t_y^\phi, t_z^\phi]$ are fixed and can be estimated by minimizing the projection over a set of checkerboard images. Please, note that they depend on the zoom values, actually, the focal length increases as the zoom increases and the position of the optical center with respect to the rotation axis also varies. For this reason the calibration of such parameters must be performed for different zoom values, as for the intrinsic parameters. Note also that (2) could be used to go from the coordinates (x_r^2, y_r^2, z_r^2) in a given position (ϕ_C^2, θ_C^2) to coordinates in the initial configuration (ϕ_C^0, θ_C^0) by simply inverting the matrix $T_\theta^{-1} R_x(\theta) T_\theta T_\phi^{-1} R_y(\phi) T_\phi$. Finally, using the transformation from (x_r^2, y_r^2, z_r^2) to (x_r^0, y_r^0, z_r^0) with the one from (x_r^0, y_r^0, z_r^0) to (x_r^1, y_r^1, z_r^1) it is possible to express a general transformation from two camera poses (ϕ_C^2, θ_C^2) and (ϕ_C^1, θ_C^1).

3.2 Tracking Algorithms for PTZ Camera

Two different tracking algorithms have been implemented for the PTZ camera. They are based on two well-known algorithms: the Camshift proposed in [3],

and the Particle Filter of [10]. They represent baselines of two different tracking philosophies here embedded in a PTZ scenario; this required to handle the position displacement and size variation of the target on the image plane due to the camera movements, as well as the control of the camera movements. In Algorithm 1, we present a general scheme for tracking using a PTZ camera, and later we will provide the details about the two different implementations. One of the crucial point of this scheme is that from frame to frame, the target is tracked using spheric coordinates, that is azimuth ϕ_T^t and elevation θ_T^t with respect to the reference coordinate system in the initial camera pose $\mathbf{C}_0 = (\phi_C^0, \theta_C^0, \mathcal{Z}_C^0)$. This is achieved in two steps: first, the estimated bounding box (BB) vertices are transformed in the initial camera pose reference system according to (2), then the spheric coordinates are computed knowing the zoom and focal length:

$$\begin{cases} x_w = f_x(\mathcal{Z}) \tan(\phi_T) \\ y_w = f_y(\mathcal{Z}) \dfrac{\tan(\theta_T)}{\cos(\phi_T)} \end{cases} \quad \begin{cases} \phi_T = \tan^{-1}\left(\dfrac{x_w}{f_x(\mathcal{Z})}\right) \\ \theta_T = \tan^{-1}\left(\dfrac{y_w \cos(\phi_T)}{f_y(\mathcal{Z})}\right) \end{cases} \tag{3}$$

In our algorithm, unlike similar previous works, at each frame, we assign to the camera a new speed and not a new position, in this way, we can produce a

Algorithm 1. A generic tracking algorithm for a PTZ camera

1: **Initialization** $t = 0$

- Move the camera to the starting point $\mathbf{C}_0 = (\phi_C^0, \theta_C^0, \mathcal{Z}_C^0)$
- Acquire the first frame from the camera
- Manually select the target Bounding Box:(c_x, c_y, l_x, l_y)
- Calculate the target model: *(depends on the adopted algorithm)*
- Transform the target position into the spheric coordinates $(c_x, c_y, l_x, l_y) \rightarrow (\phi_T^0, \theta_T^0, 1)$, the third coordinate representing the target scale, with respect to the initial zoom value.

2: **for** $t = 1$ to T **do**

3:
- Receive the current frame I_t from the camera and the camera configuration: $C^t = (\phi_C^t, \theta_C^t, \mathcal{Z}_C^t)$ from which it was captured
- Transform the previous target estimation from polar coordinate to the current view $(\phi_T^{t-1}, \theta_T^{t-1}, \mathcal{Z}_T^{t-1}) \rightarrow (\hat{c}_x^t, \hat{c}_y^t, \hat{l}_x^t, \hat{l}_y^t)$;
- Perform the tracking *(depends on the adopted algorithm)* starting from the initial guess $(\hat{c}_x^t, \hat{c}_y^t, \hat{l}_x^t, \hat{l}_y^t)$ and obtaining the new estimation $(c_x^t, c_y^t, l_x^t, l_y^t)$;
- Transform the new target estimation into polar coordinate considering the current camera position $(c_x^t, c_y^t, l_x^t, l_y^t) \rightarrow (\phi_T^t, \theta_T^t, \mathcal{Z}_T^t)$;
- Set the new camera speed and the zoom values, *(depends on the adopted control strategy)*, for both :

$$v_\phi^t = f_\phi(\phi_C^t, \hat{\phi_T^t}), \quad v_\theta^t = f_\theta(\theta_C^t, \hat{\theta_T^t}), \quad \mathcal{Z}_C^t = f_{\mathcal{Z}}(\cdot)$$

4: **end for**

smoother trajectory for the camera and the resulting video is more easily usable by a human observer.

CamShift. The target model is a 16-bin hue histogram and the target is represented as a rectangle whose sides are parallel to the image plane axis. CamShift (CS) algorithm estimates the position and scale of the rectangle at each frame. The camera control, basically proportional to the error, is set as follows:

$$v_\phi^t = \lambda_\phi(k_\phi \frac{\hat{\phi}_T^t - \phi_C^t}{T_s}) + (1-\lambda_\phi)v_\phi^{t-1}, \quad v_\theta^t = \lambda_\theta(k_\theta \frac{\hat{\theta}_T^t - \theta_C^t}{T_s}) + (1-\lambda_\theta)v_\theta^{t-1}$$
$$\mathcal{Z}_C^t = \lambda_\mathcal{Z}\mathcal{Z}_{opt}^t + (1-\lambda_\mathcal{Z})\mathcal{Z}_C^{t-1} \tag{4}$$

where $\mathcal{Z}_{opt}^t$ is computed according to the estimated target size. Given the target spheric coordinates we can measure its horizontal and vertical angular extension, $\Delta\phi$ and $\Delta\theta$, and set the zoom adequately:

$$\mathcal{Z}_{opt}^t = \min\left(\frac{\tan(\Delta\phi_0/2)}{\tan(k_\mathcal{Z}\Delta\phi/2)}, \frac{\tan(\Delta\theta_0/2)}{\tan(k_\mathcal{Z}\Delta\theta/2)}\right) \tag{5}$$

where $\Delta\phi_0$ and $\Delta\theta_0$ are the fields of view at zoom 1x and $k_\mathcal{Z}$ expresses the desired ratio between the camera field of view and the object angular extension.

Particle Filter. We implemented a Particle Filter (PF) tracker that uses the same observation model as CS, the histogram on the hue values. The state x^j of each particle j has 4 dimensions, 2 for the position and 2 for the lengths of the rectangle. At each iteration t, the particle are sampled from a Gaussian distribution $\mathcal{N}(x_{t-1}^j, \Sigma)$.

To avoid the ambiguity on the target scale in case of a uniform target we also consider an external frame around the target BB and combine two histogram distances. Let h_T the histogram of the target, h_{int}^j the histogram of the region inside the candidate, and h_{ext}^j the histogram of the region external to the j candidate. The best candidate should have a small distance for the internal histogram $d(h_{int}^j, h_T)$ and a large distance for the external histogram $d(h_{ext}^j, h_T)$, where $d(h_1, h_2)$ is the Bhattacharyya distance. The weights w_i of the particles should be proportional to the likelihood $p(h_{int}^j, h_{ext}^j|x^j)$, so we could factorize and exploit the log-likelihood formulation:

$$w_i \propto p(h_{int}^j, h_{ext}^j|x^j) \propto e^{l(h_{int}^j|x^j)}e^{l(h_{ext}^j|x^j)} = e^{-(d(h_{int}^j,h_T))^2}e^{-(1-d(h_{ext}^j,h_T))^2}$$

At each iteration t, the set of N samples and their weights $\{x_t^j, w_t^j\}$ are used to set the camera control commands. The speeds are set again with (4), where $(\hat{\phi}_T^t, \hat{\theta}_T^t)$ are obtained from the particle with the highest weight (MAP criterion). Differently from (5), the $\mathcal{Z}_{opt}^t$ is chosen considering all the particles in order to keep all of them in the field of view of the camera. As shown in Sect. 4, this choice is important since it will enhance the tracker robustness.

3.3 Performance Evaluation

The camera is placed in front of a projector screen. Before starting a video, the extrinsic parameters R_0 and T_0 are estimated for the camera in the initial configuration $C^0 = (\phi_C^0, \theta_C^0, \mathcal{Z}_C^0)$ as explained above. At each iteration of the tracking algorithm, the following 3 values are saved for the next comparison with the GT: (1) the estimated target position on the current image plane $(c_x^t, c_y^t; l_x^t, l_y^t)$, (2) the current pose of the camera $C^t = (\phi_C^t \theta_C^t, \mathcal{Z}_C^t)$, (3) an absolute timestamp T_r^t. After that, in an off-line stage, the four vertices of the bounding box at time t are projected on the current image plane according to the camera pose, using (1) and (2), and compared with the tracker estimation at the same time. Obviously, this requires a quite precise synchronization between the ground-truth data and the actual tracking data, when they are stored during the tests. We indicate with T the number of frames in the sequence, collected by the tracker, and with T_c the number of frames before the target is lost (i.e., when it is no more recovered before of the end of the sequence). Given the target GT and the tracker estimation we use five criteria to evaluate the performances:

- the mean ratio between the estimated area $\|\mathcal{A}_{est}\|$ and the GT area $\|\mathcal{A}_{GT}\|$ over the valid frames: $r_{\mathcal{A}}^T$;
- the mean distance between the GT and the estimated centers (normalized on the target diagonal) over the valid frames: d_{ct};
- the rule $\frac{\|\mathcal{A}_{GT} \cap \mathcal{A}_{est}\|}{\|\mathcal{A}_{GT} \cup \mathcal{A}_{est}\|} \geq \frac{1}{2}$ to establish if the target is tracked properly, r_c is the percentage of correctly tracked frames over the valid frames;
- the mean ratio between the target area $\|\mathcal{A}_{GT}\|$ and the image area $\|\mathcal{A}_i\|$: $r_{\mathcal{A}}^i$;
- the mean distance between the GT target and the image center: d_{ci}.

The first three parameters evaluate the accuracy of the algorithms in tracking the target, while the last two evaluate the ability of the system to keep it in the center of the field of view and at the desired dimension on the screen.

4 Experiments

The system described above has been implemented in C++, using OpenCV functions for most of the vision algorithms. It works in real-time on a laptop, Intel Core 2 Duo CPU 2.8 GHz, 3.48 GB RAM. The projector is a commercial one, with resolution 1280×1024. The PTZ camera is an Ulisse Compact by Videotec, an analog camera, PAL format, whose pan, tilt and zoom are controlled through a serial port. The intrinsic parameters have been computed as in Sect. 3 and interpolated to get the intermediate values. The parameters used in the experiments are the following: $[t_x^\phi, t_y^\phi, t_z^\phi] = [50, 0, 180 - 21\mathcal{Z}]$, $[t_x^\theta, t_y^\theta, t_z^\theta] = [0, 60, -40 - 21\mathcal{Z}]$, for the camera model, $k_\phi = 0.3$, $k_\theta = 0.3$, $k_\mathcal{Z} = 5$, $\lambda_\phi = 0.7$, $\lambda_\theta = 0.7$, $\lambda_\mathcal{Z} = 0.4$ for the control part. The CS parameters are set to the default values $V_{min} = 10$, $S_{min} = 30$, $V_{max} = 256$, and the sampling time is set to $T_s = 0.1$. For the PF we used 400 particles, with variance $\Sigma = diag(25, 25, 2, 2)$, a 16 bin hue histogram

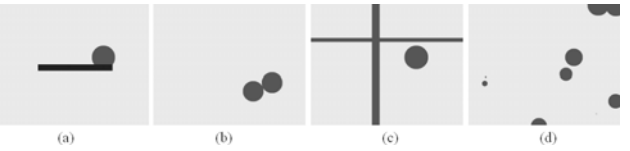


Fig. 2. The synthetic videos used in the comparison. (a): a black rectangle occludes the target; (b): the targets splits into two identical target, then one disappears;(c): a red-shape is in the background; (d) some dots similar to the targets appear and disappear in the background.

Table 1. Comparison between the CS and the PF trackers on a set of videos in which a synthetic target is projected in different scenarios

video	tracker	T	*end*	T_c	r_c	$r^T_{\mathcal{A}}$	d_{ct}	$r^i_{\mathcal{A}}$	d_{ci}	fps
basic	CS	165	yes	165	100.00	0.815	0.070	0.070	36.361	2.86
	PF	158	yes	158	82.91	1.306	0.106	0.017	20.239	3.77
occlusion	CS	998	yes	998	65.31	0.822	0.151	0.080	107.776	11.20
	PF	530	yes	530	66.26	0.826	0.183	0.020	40.918	5.90
splitting	CS	171	no	76	20.48	0.841	0.175	0.045	119.826	9.60
	PF	313	yes	313	28.74	1.191	0.560	0.012	81.189	3.90
red back	CS	153	no	86	56.29	0.844	0.043	0.051	74.479	10.07
	PF	316	no	204	14.57	3.369	1.841	0.004	40.654	3.81
lighting	CS	863	yes	863	81.88	0.897	0.122	0.057	25.546	12.35
	PF	348	yes	348	69.50	1.371	0.161	0.017	28.969	4.62

and $T_s = 0.2$. In Table 1, we report some examples on the effectiveness of the experimental evaluation setup and the comparison between the two algorithms. First, we applied the system to the simples case: a red ball moving in a cyan background. In this case, the CS algorithm works perfectly and this allows us to verify the precision of the evaluation testbed, the percentage of correctly tracked frames r_c is 100%, the area ratio $r^T_{\mathcal{A}}$ is almost 1, and the normalized distance between the centers d_{ct} is very small, as we expect in this successfully tracked sequence.

Then, we created some more challenging scenarios, shown in Fig. 2. In all these cases the red target follows a circular trajectory and its dimension varies periodically, with a global period of 40 secs. The more complex experiments allow a deep comparison between the two algorithms. The main difference is that PF can successfully track the splitting sequence, while CS breaks after some frames. As shown in Fig. 3, this is due to the control strategy for the zoom, that aims to keep all the particles in the field of view. On the contrary, CS is forced to choose only one hypothesis, and if it is the wrong one it fails. Nevertheless, this choice allows CS to provide a higher magnification of the target as listed in the column $r^T_{\mathcal{A}}$ of Table 1. Moreover, CS is typically more precise in term of percentage of correctly tracked frames, mainly because PF best candidate

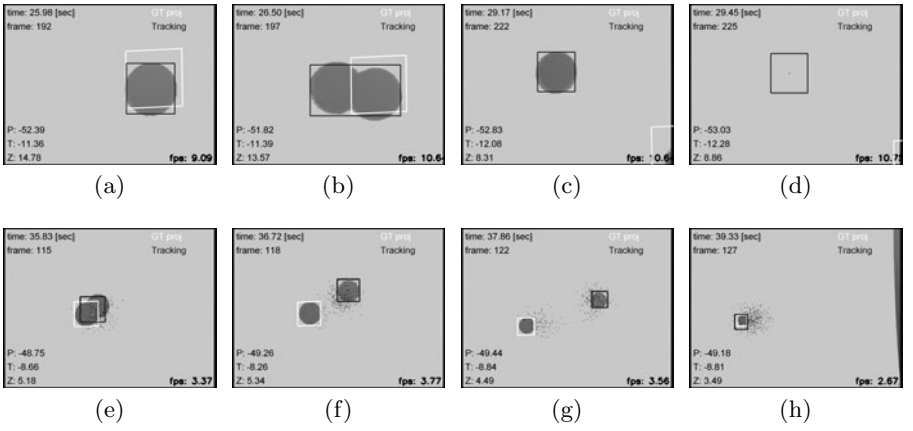


Fig. 3. Comparing CamShift (CS) and Particle Filter (PF) on the *splitting* sequence. The black BB is the target estimation from the tracking algorithm and the white one is the projection of the GT on the current image plane. Top row: some frames from the CS sequence; bottom row: some frames from the PF sequence. The CS algorithm (top row) works properly with a single target (a), but then when multiple targets are present (b) and the two candidates move in different direction, it chooses to follow a single one (c). If the wrong one is chosen, the tracking fails as soon as it disappears (d). On the other hand (bottom row), PF can recover from the error because both target candidates are automatically kept in the field of view of the camera by zooming out (f), (g); then, when the wrong one disappears the correct one is tracked again (h).

does not always fit perfectly the target, or the dynamic model do not succeed to follow quick changes of the target size. Finally, as the target in the PF tracking smaller because of a lower zoom, it is closer to the center of the screen as shown in column d_{ci} of the table.

5 Conclusions

In this work, we have proposed and implemented a novel method to test different PTZ algorithms. We adapted two classical tracking algorithms to the PTZ framework and evaluated them using the same experimental conditions. The obtained results show the effectiveness of the system and highlight the different behaviors of the two algorithms. Future work will focus on the tuning of the precision of the evaluation system and the comparison of the algorithms in more complex scenarios.

Acknowledgments. The authors would like to thank Michele Stoppa for helpful discussions and technical support.

References

1. Agapito, L., Hartley, R., Hayman, E.: Linear self-calibration of a rotating and zooming camera. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1 (1999)
2. Bimbo, A.D., Dini, F., Grifoni, A., Pernici, F.: Uncalibrated framework for on-line camera cooperation to acquire human head imagery in wide areas. In: AVSS 2008: Proceedings of the 2008 IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance, pp. 252–258. IEEE Computer Society, Washington, DC, USA (2008)
3. Bradski, G.R.: Computer vision face tracking for use in a perceptual user interface. Intel Technology Journal 2(2) (1998)
4. Cai, Y., de Freitas, N., Little, J.: Robust visual tracking for multiple targets. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3954, pp. 107–118. Springer, Heidelberg (2006)
5. Collins, R., Liu, Y., Leordeanu, M.: Online selection of discriminative tracking features. IEEE Transactions on Pattern Analysis and Machine Intelligence 27, 1631–1643 (2005)
6. Comaniciu, D., Ramesh, V., Meer, P., Member, S., Member, S.: Kernel-based object tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 25, 564–577 (2003)
7. Davis, J., Chen, X.: Calibrating pan-tilt cameras in wide-area surveillance networks. In: Proceedings Ninth IEEE International Conference on Computer Vision, vol. 1, pp. 144–149 (October 2003)
8. Ellis, A., Shahrokni, A., Ferryman, J.M.: Pets2009 and winter-pets 2009 results: A combined evaluation. In: Twelfth IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS-Winter), pp. 1–8 (December 2009)
9. Greiffenhagen, M., Comaniciu, D., Niemann, H., Ramesh, V.: Design, analysis, and engineering of video monitoring systems: An approach and a case study. PIEEE 89(10), 1498–1517 (2001)
10. Isard, M., Blake, A.: Condensation: Conditional density propagation for visual tracking. International Journal of Computer Vision 29, 5–28 (1998), doi:10.1023/A:1008078328650
11. Lieberknecht, S., Benhimane, S., Meier, P., Navab, N.: A dataset and evaluation methodology for template-based tracking algorithms. In: 8th IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2009, pp. 145–151 (October 2009)
12. Ling, H., Wu, Y., Blasch, E., Chen, G., Lang, H., Bai, L.: Evaluation of visual tracking in extremely low frame rate wide area motion imagery. In: 14th Conference on Information Fusion (FUSION, 2011). IEEE, Los Alamitos (2011)
13. Raimondo, D.M., Gasparella, S., Sturzenegger, D., Lygeros, J., Morari, M.: A tracking algorithm for ptz cameras. In: 2nd IFAC Workshop on Distributed Estimation and Control in Networked Systems, NecSys 2010 (September 2010)
14. Varcheie, P., Bilodeau, G.-A.: People tracking using a network-based ptz camera. Machine Vision and Applications 22, 1–20 (2010), doi:10.1007/s00138-010-0300-1
15. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM Computing Surveys 38(4), 13:1–13 (2006)

Unsupervised Activity Extraction on Long-Term Video Recordings Employing Soft Computing Relations

Luis Patino[1], Murray Evans[2], James Ferryman[2], François Bremond[1], and Monique Thonnat[1]

[1] INRIA Sophia Antipolis - Méditerranée
2004 route des Lucioles - BP 93 - 06902 Sophia Antipolis, France
[2] School of Systems Engineering, University of Reading
RG6 6AY United Kingdom

Abstract. In this work we present a novel approach for activity extraction and knowledge discovery from video employing fuzzy relations. Spatial and temporal properties from detected mobile objects are modeled with fuzzy relations. These can then be aggregated employing typical soft-computing algebra. A clustering algorithm based on the transitive closure calculation of the fuzzy relations allows finding spatio-temporal patterns of activity. We present results obtained on videos corresponding to different sequences of apron monitoring in the Toulouse airport in France.

Keywords: Video data mining, vision understanding, discovery in multimedia database, soft computing.

1 Introduction

Scene understanding corresponds to the real time process of perceiving, analysing and elaborating an interpretation of a 3D dynamic scene observed through a network of sensors (including cameras and microphones). This process consists mainly in matching signal information coming from sensors observing the scene with a large variety of models which humans are using to understand the scene. Although activity models can be built by experts of the domain, this might be a hard and time-consuming task depending on the application and the spectrum of activities that may be observed. The challenge thus consists of discovering, in an unsupervised manner, the significant activities observed from a video sequence. Knowledge discovery systems (KDS) aim at helping the human operator on this aspect. KDS systems have become a central part on many domains where data is stored in a database, but little research has been only done in the field of video data-mining. It must be said the task is particularly challenging because of the difficulty in identifying the interesting patterns of activity in the video due to noise, incomplete or uncertain information inherently present in the data. Soft computing methodologies are particularly suitable for these tasks because

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 91–100, 2011.
© Springer-Verlag Berlin Heidelberg 2011

they provide the capability to process uncertain or vague information, as well as a more natural framework to cope with linguistic terms and produce natural language-like interpretable results. Fuzzy sets are the corner stone of soft computing together with other techniques such as neural networks and genetic algorithms. The relation between different existing fuzzy sets can be graded with the use of fuzzy relations [15]. Various fuzzy-based soft computing systems have been developed for different applied fields of data mining; but only a few systems employ soft computing techniques to partially characterize video activity patterns [3,6]; In this paper we present a fully unsupervised system exploiting the use of fuzzy relations for the discovery of activities from video. First we model spatial and temporal properties from detected mobile objects employing fuzzy relations. We employ typical soft-computing algebra to aggregate these relations. A clustering algorithm based on the transitive closure calculation of the fuzzy relations allows finding spatio-temporal patterns of activity with different granularities. We have applied the proposed technique to different video sequences of apron monitoring in the Toulouse airport in France.

The reminder of this paper is as follows. In the next section, we give a short overview of the related work. We give a general overview of the system architecture and present our global approach in section 3 . The object detection and tracking process is given in section 4, then the data preprocessing steps previous to activity extraction are explained (section 5). We give the activity clustering methodology in section 6. Section 7 gives the main results and evaluation. Finally, Section 8 draws the main conclusions and describes our future work.

2 Related Work

Extraction of the activity contained in the video by applying data-mining techniques represents a field that has only started to be addressed. Although the general problem of unsupervised learning has been broadly studied in the last couple of decades, there are only a few systems which apply them in the domain of behaviour analysis. A few systems employ soft computing techniques to characterize video activity patterns [3,6] but the methodology to self-discover new activities is still missing. Because of the complexity to tune parameters or to acquire knowledge, most systems limit themselves to object recognition . For behaviour recognition, three main categories of learning techniques have been investigated.

• The first class of techniques learns the parameters of a video understanding program. These techniques have been widely used in case of event recognition methods based on neural networks [5], Bayesian classifiers and HMMs [8,13].
• The second class consists in using unsupervised learning techniques to deduce abnormalities from the occurring events [14].
• The third class of methods focuses on learning behaviour based on trajectory analysis. This class is the most popular learning approach due to its effectiveness in detecting normal/abnormal behaviours; for instance, on abnormal trajectory detection on roads [9,12] or pedestrian trajectory characterisation [1]. Hidden

Markov Models (HMM) have also been employed to detect different states of predefined normal behaviour [2,10]. All these techniques are interesting, but little has been said about the semantic interpretability of the results. Indeed, more than trajectory characterisation, we are interested in extracting meaningful activity, where different trajectory types may be involved. This work comes thus into the frame of behaviour extraction from trajectory analysis however we have in addition a higher semantic level that employs spatial and temporal proximity relations between detected mobiles to characterise the ongoing different activities of the scene. In a similar framework, Dubba et al [4] have researched into transforming tracking data into a relational form where the relations are spatio-temporal relations among objects. This is the most comparable work to us, not only because they also look to identify activities as groups with coherent spatio-temporal information, but because they have worked on the same dataset. However, Dubba et al. employ a supervised approach (based on Inductive Logic Programming) and thus requires large quantities of annotated data. Our proposed approach is on the contrary completely unsupervised.

3 General Overview of the System

Our proposed system is mainly composed of two different processing components. The first one is for the detection and tracking of objects. The second subsystem works off-line and achieves the extraction of activity patterns from the video. This subsystem is composed of two modules: The trajectory speed analysis module, and the activity analysis module. The first is aimed at segmenting the trajectory into tracklets of fairly similar speed. The latter is aimed at extracting complex patterns of activity, which include spatial information (coming from the trajectory analysis) and temporal information related to the interactions of mobiles observed in the scene.

Streams of video are acquired at a speed of 10 frames per second. The on-line (real time) analysis subsystem takes its input directly from the data acquisition component; the video is stored in the DB parallel to the real time processing.

4 Real-Time Processing Object Detection and Tracking

The detection and tracking is performed using multiple cameras with an overlapping field of view, and consists of three stages: Detection in the image plane, tracking in the image plane, fusion and tracking in 3D.

4.1 Detection

Detection is performed by combining change detection and motion detection. The first detector is the Adaptive Gaussian Mixture Model of Zivkovic [16]. This method builds on the standard Gaussian Mixture Model approach but permits an adaptive number of components per pixel. This generally produces good object silhouettes and runs very fast.

To complement the change detector, a motion detector is employed. In this method, the three most recent frames $\{I(t), I(t-1), I(t-2)\}$ are used to determine the motion in the most recent frame $I(t)$. A set of corner features is detected in frame $I(t+1)$ using the method in [11]. These features are then tracked forwards to frame $I(t)$ and backwards to frame $I(t+2)$ using the sparse optical flow method in [7]. This results in two direction vectors for each feature, $[d_{0\to1}, d_{1\to2}]$. Features are clustered based on their motion with a constraint on the maximum distance between any two features. A triangulation of each cluster of features is performed such that the cluster can be rendered to a binary motion mask. The two binary motion masks, from the change detector and the motion detector, are combined through a simple logical AND.

4.2 Image Plane Tracking

Tracking in the image plane is performed using two simple templates and a KLT feature tracker. When the detector returns a detection, it can either be associated to an existing tracked target, or to a new target. When a new target is created, two small images are created. One is a greyscale image of the size of the detection bounding box, while the other is an RGB image of the same size. The greyscale image is the *detection mask template* D_t, and is initialised from the binary motion mask of the current image M_t, while the RGB image is the appearance template A_t and is initialised from the RGB pixel values of the current image I_t. Thus, on initialisation, if the top left corner of the detection bounding box is at image coordinates x, y:

$$D_t(u,v) = \begin{cases} 0 & \text{if } M_t(x+u, y+v) = 0 \\ 255 & \text{otherwise} \end{cases} \tag{1}$$

$$A_t(u,v) = I_t(x+u, y+v) \tag{2}$$

When a detection is associated to a new target, the detection and appearance templates are updated as a running average. Should the detection indicate a change in the width or height of the bounding box, the template images can be easily expanded or cropped as required.

Each tracked target maintains a set of KLT features that are tracked between frames. The overall plane tracking method is generally good enough to reliably maintain a track on large objects such as vehicles, which often stop for extended periods in the scene. It is not intended to track objects through occlusions, but rather to detect the presence of objects, and maintain the presence of static objects.

4.3 Multi-camera Fusion and 3D Tracking

The final stage of tracking is performed in the 3D coordinate system of the scene (though tracking remains 2D on the ground plane). Camera calibration is used to project the bounding boxes of per-camera tracking targets to each other camera view as four epipolar lines from the four corners of the bounding box. This

provides a mechanism for rating the extent to which tracking targets are related between views, by determining the extent to which a bounding box fits between the extremal epipolar lines of a bounding box from another view. Agglomerative clustering is used to determine possible solutions for the correct fusion of targets, and an optimisation process then determines the optimal clustering for a single frame of video. Optimal solutions are retained over a temporal window, and an overall optimal association of per-camera targets to fused targets is determined, and fused tracking targets updated for every new frame.

5 Data Preprocessing

In order to discover meaningful activity clusters, it is of prime importance to have available detailed information allowing to detect the different possible interactions between mobiles. As our system is based on trajectory analysis, the first step to prepare the data for the activity clustering methodology is to extract tracklets of fairly constant speed allowing to characterise the displacements of the mobile or its stationary state.

If the dataset is made up of N objects, the trajectory tr_j for object O_j in this dataset is defined as the set of points $[x_j(t), y_j(t)]$ corresponding to their position points; x and y are time series vectors whose length is not equal for all objects as the time they spend in the scene is variable. The instantaneous speed for that mobile at point $[x_j(t), y_j(t)]$ is then $v\,(t) = \left(\dot{x}\,(t)^2 + \dot{y}\,(t)^2\right)^{\frac{1}{2}}$. The objective is then to detect those points of changing speed allowing to segment the trajectory into tracklets of fairly constant speed so that the trajectory can be summarised as a series of displacements at constant speed or in stationary state.

The mobile object time series speed vector is analysed in the frame of a multiresolution analysis of a time series function $v\,(k)$ with a smoothing function, $\rho_{2^s}\,(k) = \rho\,(2^s k)$, to be dilated at different scales s. In this frame, the approximation A of $v\,(k)$ by ρ is such that $A_{2^{s-1}}v$ is a broader approximation of $A_{2^s}v$. By analyzing the time series v at coarse resolutions, it is possible to smooth out small details and select those points associated with important changes.

The speed change points are then employed to segment the original trajectory tr_j into a series of i tracklets tk. Each tracklet is defined by two key points, these are the beginning and the end of the tracklet, $[x_j^i(1), y_j^i(1)]$ and $[x_j^i(end), y_j^i(end)]$ as they define where the object is coming from and where it is going to and also with approximative constant speed. We build a feature vector from these two points. By globally reindexing all tracklets, let m be the number of total tracklets extracted, we obtain the following tracklet feature vector :

$$tk_m = [x_m(1), y_m(1), x_m(end), y_m(end)] \tag{3}$$

6 Activity Clustering Methodology

We understand activity as the interactions occurring between mobile objects themselves and those between mobiles and the environment. We propose in this

work to model those interactions employing Soft computing techniques. The motivation is that they provide uncertain information processing capability; set a framework to work with symbolic/linguistic terms and thus allows producing natural language-like interpretable results.

6.1 Preliminary Definitions

A fuzzy set is a set of ordered pairs such as $A = \{(x, \mu_A(x)) \mid x \epsilon X\}$ and the belonging of x to A is given by μ_A. Any relation between two sets X and Y is known as a binary relation R:

$$R = \{((x, y), \mu_R(x, y)) \mid (x, y) \epsilon X \times Y\}$$

and the strength of the relation is given by $\mu_R(x, y)$. Let's consider now two different binary relations, *R1* and *R2*, linking three different fuzzy sets X, Y, and Z : R1 = x is relevant to y; R2 = y is relevant to z.

It is then possible to find to which measure x is relevant to z (noted *R=R1oR2*) by employing the extension principle:

$$\mu_{R=R1 \circ R2}(x, z) = \max_y \min[\mu_{R1}(x, y), \mu_{R2}(y, z)]$$

It is interesting to verify whether the resulting relation is symmetric, $R(x, y) = R(y, x)$, reflexive $R(x, x) = 1$, which make of R a compatibility relation and occurs in most cases when establishing a relationship between binary sets. Because R was calculated employing the extension principle, R is also a transitive relation. $R(x, y)$ is a transitive relation if $\exists z \in X, z \in Y / R(x, y) \geqslant \max_z \min[R(x, z), R(z, y)]$. R can be made furthermore closure transitive following the next steps

Step1. $R' = R \cup (R \circ R)$
Step2. If $R' \neq R$, make$R = R'$ and go to step1
Step3. $R = R'$ Stop. R is the transitive closure where

$$R \circ R(x, y) = \max_z \min(R(x, z), R(z, y)) \tag{4}$$

R is now a transitive similarity relation with R indicating the strength of the similarity. If we define a discrimination level α in the closed interval [0,1], an $\alpha - cut$ can be defined such that

$$R^\alpha(x, y) = 1 \Leftrightarrow R(x, y) \geqslant \alpha; R = \bigcup_\alpha \alpha R^\alpha \tag{5}$$

It is thus implicit that $\alpha_1 > \alpha_2 \Leftrightarrow R^{\alpha_1} \subset R^{\alpha_2}$; thus, the R^α form a nested sequence of equivalence relations, or from the classification point of view, R^α induces a partition π^α of $X \times Y$ (or X^2) such that $\alpha_1 > \alpha_2$ implies π^{α_1} is a partition refinement of π^{α_2}.

6.2 Clustering of Video Data

We now set out to establish the appropriate relations between detected mobiles in the video reflecting spatio-temporal similarities in order to obtain activity patterns. With this aim, we define the following relations:

$R1_{ij}$: mobile object *O(i)* meets mobile object *O(j)*. In this case the action 'meets' must be understood spatially and thus gives a degree of spatial closeness between the two mobiles.

$$R1_{ij} = min\left(\left\|tk_i(1), tk_j(1)\right\|, \left\|tk_i(1), tk_j(2)\right\|\right), \\ \left(\left\|tk_i(2), tk_j(1)\right\|, \left\|tk_i(2), tk_j(2)\right\|\right) \tag{6}$$

$R2_{ij}$: mobile object *O(i)* starts equal to mobile object *O(j)*. Here we are attempting to relate mobile objects that share temporal closeness.

$$R2_{ij} = 1 - abs(start\_time(i) - start\_time(j)) \tag{7}$$

$R3_{ij}$: mobile object *O(i)* starts after mobile object *O(j)*. Here we are attempting to relate mobile objects that appear in a sequential manner.

$$R3_{ij} = 1 - abs(start\_time(i) - end\_time(j)) \tag{8}$$

Obtaining the patterns of activity is achieved by aggregating the above spatio-temporal relations with a typical T-norm operator.

$R = R1 \cup R2 \cup R3$ aggregates temporal similarity relations between mobiles. We calculate the transitive closure of this new relation. Analogically to section 6.1 an $\alpha - cut$ can be defined such that $R^{\alpha}(x, y) = 1 \Leftrightarrow R(x, y) \geqslant \alpha$ and R^{α} induces a new partition $\pi^{\alpha} = \{C_1^{\alpha}, ..., C_i^{\alpha}, ..., C_{n^{\alpha}}^{\alpha}\}$; each C_i^{α} represents a discovered spatio-temporal activity pattern.

7 Results and Evaluation

The algorithm can be applied to any given period monitoring the servicing of an aircraft in the airport docking area. In order to evaluate whether the activity extraction algorithm works properly and to asses the correctness of the results, we took five video datasets (each lasting about 1 hour) with available Ground-truth annotation and containing the start and end time for the most relevant activity events of the sequence.

The procedure to find the activity clusters is applied as given in section 6.2. In this work, the final relation R, which verifies the transitive closure, is thresholded for different $\alpha - cut$ values going from 0.05 to 0.95 and with a step value of 0.10. Low $\alpha - cut$ values produce only a few number of clusters of broad resolution as most of the activity is merged spatially and temporally. $\alpha - cut$ values near to one produce activity clusters of higher resolution with more precisely defined activities. At each resolution it is possible to calculate the temporal overlap

Fig. 1. Example of an activity cluster obtained. The left panel presents the tracklets of the mobiles participating in the Frontal Loading activity. Filled circles indicate the beginning of a tracklet. Empty circles indicate the end of a tracklet. The right panel presents the start frame of the activity.

between the extracted activity clusters and the ground-truth clusters. The quantitative result of this comparison is given in table 1. For each video sequence and for each ground-truth event only the best overlap across all $\alpha - cut$ values is reported. Remark that specific activities involving one mobile require precise definition obtained with activity clusters of higher resolution while loading operations involving the interaction of several mobiles are defined with mid-resolution activity clusters ($\alpha - cut$ values of 0.75 or 0.85). For instance, figure 1 presents a frontal loading activity cluster obtained for an $\alpha - cut$ value of 0.75. In general, all events are recognized correctly in all video sequences. When the percentage of overlap decreases or even goes to zero, it is mainly due to low-level object occlusion problems, which do not allow extracting all mobile trajectories and disturbs then the analysis of all possible mobile interactions.

Our results can partially be compared to those obtained with a supervised approach to learn apron activity models with Inductive Logic Programming (Dubba et al. [4]). As previously indicated, Dubba et al. have worked on the same apron monitoring video dataset from the Toulouse airport in France. Dubba et al. have concentrated on supervised learning of four apron activities: Aircraft arrival;

Table 1. Percentage of overlap between discovered activities and the reference events contained in the ground-truth. The symbol * indicates that for all video sequences, the ground-truth event matches a discovered activity obtained for an alpha value of 0.95. NA indicates 'does not apply' (does not appear in the video sequence).

	video sequence				
Reference event	1	2	3	4	5
GPU vehicle arrival*	96%	0%	68%	90%	91%
Handler deposits chocks*	57%	81%	65%	60%	67%
Aircraft arrival*	90%	91%	71%	81%	66%
Jet Bridge positionning	56%	42%	70%	58%	70%
Frontal loading operation 1	NA	NA	NA	25%	32%
Frontal loading operation 2	NA	NA	NA	94%	NA
Frontal loading operation 3	NA	NA	NA	67%	NA
Frontal loading operation 4	NA	NA	NA	73%	NA
Rear loading operation 1	43%	52%	43%	82%	50%
Rear loading operation 2	53%	41%	38%	NA	63%
Rear loading operation 3	93%	NA	NA	NA	25%
Push-back positioning*	0%	69%	0%	89%	96%
Aircraft departure*	89%	94%	63%	81%	75%

Table 2. Results comparison between our results and those presented in Dubba et al. at ECAI 2010 [4]

	Dubba et al.	Our Approach	
Reference Event	TPR	TPR	Mean temporal overlap with GT
Rear Loading / Unloading	80 %	100 %	53 %
Aircraft arrival	100 %	100 %	80 %
Aircraft departure	57 %	100 %	80 %
Jet Bridge positionning	57 %	100 %	59 %
Frontal Loading / Unloading	--	100 %	58 %
GPU vehicle arrival	--	80 %	86 %
Handler deposits chocks	--	100 %	66 %
Push-back positioning	--	60 %	85 %

Aircraft departure; Rear Loading/unloading; Jet Bridge Positionning. Dubba et al. obtained a global True Positive Rate (TPR) of 74%. In our work (from Table 1), we have 80% global True Positive Rate for the recognition of eight apron activities: Aircraft arrival; Aircraft departure; Rear Loading/unloading; Jet Bridge Positionning; Frontal Loading/Unloading; GPU vehicle arrival; Handler deposits chocks; Push-back positioning. Dubba et al. approach works as a hit or miss recognition system and in their paper there is no information on what is the temporal overlap between the recognised activities and the ground-truth activities. In our case such temporal overlap has a global value of 73%. The event-by-event comparison between the two approaches is detailed in table 2.

8 Conclusions

We have presented in this paper, a novel approach to extract activity patterns from video. The technique is unsupervised and is based on the use of fuzzy relations to model Spatial and temporal properties from detected mobile objects. Fuzzy relations are aggregated employing typical soft-computing algebra. A clustering algorithm based on the transitive closure calculation of the fuzzy relations allows finding spatio-temporal patterns of activity. Our current results are encouraging as the final patterns of activity are given with coherent spatial and temporal information, which is understandable for the end-user. When comparing our results with explicit ground-truth given by a domain expert, we were able to identify the events in general with a temporal overlap of at least, or near, 50%. Events with small temporal overlap in some video sequences is because of low-level detection problems. The comparison with a supervised method on the same data indicates that our approach is able to extract the interesting activities signalled in the ground-truth with a higher True Positive Rate. More importantly, our approach is completely unsupervised. In our future work we will try to work on improving our technique to determine the meaningfulness (or abnormality) of single activity patterns. We also plan to work on the semantical description of the activity clusters.

References

1. Anjum, N., Cavallaro, A.: Single camera calibration for trajectory-based behavior analysis. In: AVSS 2007, IEEE Conference on Advanced Video and Signal Based Surveillance, pp. 147–152 (2007)
2. Bashir, F., Khokhar, A., Schonfeld, D.: Object Trajectory-Based Activity Classification and Recognition Using Hidden Markov Models. IEEE Transactions on Image Processing 16, 1912–1919 (2007)
3. Doulamis, A.: A fuzzy video content representation for video summarization and content-based retrieval. Signal Processing 80(6), 1049–1067 (2000)
4. Dubba, K.S.R., Cohn, A.G., Hogg, D.C.: Event model learning from complex videos using ilp. In: Proceeding of ECAI 2010, the 19th European Conference on Artificial Intelligence, pp. 93–98 (2010)
5. Foresti, G., Micheloni, C., Snidaro, L.: Event classification for automatic visual-based surveillance of parking lots. In: Proceedings of the 17th International Conference on Pattern Recognition, ICPR 2004, vol. 3, pp. 314–317. IEEE, Los Alamitos (2004)
6. Lee, S.W., Mase, K.: Activity and Location Recognition Using Wearable Sensors. IEEE Pervasive Computing 1(03), 24–32 (2002)
7. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proc. of 7th International Joint Conference on Artificial Intelligence (IJCAI), pp. 674–679 (1981)
8. Lv, F., Song, X., Wu, B., Singh, V., Nevatia, R.: Left luggage detection using bayesian inference. In: Proceedings of the 9th IEEE International Workshop (2006)
9. Piciarelli, C., Foresti, G., Snidaro, L.: Trajectory clustering and its applications for video surveillance. In: Proceedings of the IEEE Conference on Advanced Video and Signal Based Surveillance, AVSS 2005, vol. 18, pp. 40–45. IEEE, Los Alamitos (2005)
10. Porikli, F.: Learning object trajectory patterns by spectral clustering. In: 2004 IEEE International Conference on Multimedia and Expo (ICME), vol. 2, pp. 1171–1174. IEEE, Los Alamitos (2004)
11. Shi, J., Tomasi, C.: Good features to track. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 593–600 (1994)
12. Stauffer, C., Grimson, W.E.L.: Learning patterns of activity using real-time tracking. IEEE Trans. Pattern Anal. Mach. Intell. 22(8), 747–757 (2000)
13. Wilson, A.D., Bobick, A.F.: Hidden Markov models for modeling and recognizing gesture under variation. International Journal of Pattern Recognition and Artificial Intelligence 15, 123–160 (2001)
14. Xiang, T., Gong, S.: Video behaviour profiling and abnormality detection without manual labelling. In: Tenth IEEE International Conference on Computer Vision, ICCV 2005, vol. 2 (2005)
15. Zadeh, L.: Similarity relations and fuzzy ordering. Information sciences 3, 159–176 (1971)
16. Zivkovic, Z.: Efficient adaptive density estimation per image pixel for the task of background subtraction. Pattern Recognition Letters (2006)

Unsupervised Discovery, Modeling, and Analysis of Long Term Activities

Guido Pusiol, Francois Bremond, and Monique Thonnat

Pulsar, Inria - Sophia Antipolis, France

Abstract. This work proposes a complete framework for human activity discovery, modeling, and recognition using videos. The framework uses trajectory information as input and goes up to video interpretation. The work reduces the gap between low-level vision information and semantic interpretation, by building an intermediate layer composed of Primitive Events. The proposed representation for primitive events aims at capturing meaningful motions (actions) over the scene with the advantage of being learned in an unsupervised manner. We propose the use of Primitive Events as descriptors to discover, model, and recognize activities automatically. The activity discovery is performed using only real tracking data. Semantics are added to the discovered activities (e.g., "Preparing Meal", "Eating") and the recognition of activities is performed with new datasets.

1 Introduction

More than 2 billion people will turn over 65 year old by the year 2050. It is of crucial importance for the research community to help aging adults live independently for longer periods of time. The transition from their homes to new and unknown environments (i.e. an assisted living facility) add stressors that deteriorate theirs mind, memory and body. If we can keep the elders in their own homes over longer periods of time, they are in an environment that they know and trust so they can have a greater confidence leading to better quality of live.

The understanding of daily activities is the key to help solve the problem and is a topic that remains open. In the literature the computational approaches assume usually prior knowledge of the activities and the environment. This knowledge is used explicitly to model the activities in a supervised manner. In video surveillance the systems produce large quantities of data and it becomes almost impossible to continually monitor these data sources manually. It is of crucial importance to build computer systems capable of analyzing human behavior with minimal supervision.

Computer-based video applications need several processing levels, from low-level tasks of image processing to higher levels concerning semantic interpretation of the scene. Nowadays the reduction of the gap between low-level tasks up to video understanding is still a challenge.

This work addresses these problems by presenting a novel framework that links the basic visual information to the discovery and recognition of long term activities (e.g. "Eating") by constructing an intermediate layer of Primitive Events in a completely unsupervised way.

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 101–111, 2011.
© Springer-Verlag Berlin Heidelberg 2011

The intermediate layer aims at capturing the motion of the individual to perform basic tasks, using only minimal information (person position and dynamics). The use of small amounts of information allows the fast analysis of large amount of data. The advantage of using visual information is that it is captured using non-invasive sensors and enables to reduce the complexity of systems that use numerous sensors to enrich the observation data [19].

To automatically model the Primitive Events: (a) the human actions are learned in an unsupervised way; (b) the scene contextual information is learned capturing meaningful scene regions; (c) the primitive events are built by merging the actions and the scene information.

The composition of primitive events is very informative about the description of many activities. Thus, we search for particular sequences within the primitive event layer to discover interesting activities. The discovered activities are used to build generic activity models and the modeled activities are recognized in new unseen video datasets.

This paper is divided as follows: in the third section we explain how actions are learned, in the fourth section how the scene contextual information is obtained, in the fifth section how actions are abstracted to primitive events and how to combine the primitive events to discover and model activities, in the sixth section the activity recognition procedure is explained and in the seventh section we evaluate the approach in home-care applications.

2 Related Work

The advances made in the field of object tracking allow data-mining techniques to be applied to large video data. Recently particular attention has been focused on the object trajectory information over time to understand long term activities. Trajectory-based methods to analyze activity can be divided in two groups, supervised and unsupervised.

Typical supervised methods such as [7,11,5] can build activity models in a very accurate way. The problem is that they require big training datasets labeled manually.

The unsupervised methods include Neural Networks based approaches such as [9,8,13,10]. They can represent complex nonlinear relations between trajectory features in a low-dimensional structure. These networks can be trained sequentially and updated with new examples, but the complexity of the parametrization usually makes the networks grow and become useless after long periods of time.

Clustering approaches such as Hierarchical Methods [1] allow multi-resolution activity modeling by changing the number of clusters, but the clustering quality depends on the way to decide when clusters should be merged or not. Adaptive methods [14], where the number of clusters adapts over time, make on-line modeling possible without the constraint of maintaining a training dataset. In these methods it is difficult to initialize a new cluster preventing outlier inclusion. Other methods [17,2] use dynamic programing based approaches to classify activitivities. These methods are effective when time ordering constraints hold.

Hidden Markov Model (HMM) based approaches such as [15] capture spatio-temporal relations in trajectory paths, allowing high-level analysis of an activity, which is suitable for detecting abnormalities. These methods require prior domain knowledge and their adaptability in time is poor.

Morris and Trivedi [12] learn scene points of interest (POI) and model the activities between POIs with HMMs encoding trajectory points. This approach is suitable to detect abnormal activities and performs well when used in structured scenes (i.e. if the usual trajectory paths are well defined, such as on a highway). But the method requires activities to have time order constraints. Also [6] merges the scene POIs and sensorial information. But the method requires a manual specification of the scene.

Most of the methods described above can be applied only in structured scenes (i.e. highway, traffic junction), and cannot really infer activity semantics. To solve these problems we propose an approach that is suitable to unstructured scenes and which is the first to combine local and global descriptors to recognize long term activities.

3 Actions

To understand activities, we propose first to learn the actions that compose them by cutting a video into meaningful action segments. Each segment aims at capturing a person's action such as "standing up". We mark the beginning and ending of a segment by detecting the person's change of state (motion/static). From a video datafile we obtain a sequence of action segments. At each segment we compute the person's main dynamics by clustering meaningful trajectories. Finally, we build $Action$ descriptors that capture the global and local motion of a person in an action segment.

3.1 Global Position and Speed

We compute the person position at each frame by using a person tracker. The position is given to a linear Kalman-filter $(K1)$. At each new frame the prediction of $K1$ is averaged with the new position observation (obs) obtaining a smoothed trajectory (pos):

$$pos_{frame_i} = Avg(obs_{frame_i}, K1(obs_{frame_{i-1}}))$$

The $speed$ of a person in a new frame, is computed by averaging the prediction of another Kalman-filter $(K2)$ and the real speed observation $(sobs)$ in the new frame:

$$speed_{frame_i} = Avg(sobs_{frame_i}, K2(speed_{frame_{i-1}})$$

3.2 Action Segments and Local Dynamics

An **Action Segment** starts with a person's change of state and ends with the next change of state (motion/static). The changes of state are computed sequentially by thresholding the person's $speed$ at each frame.

Local Dynamics are a set of short trajectories describing the motions in an action segment. To compute these trajectories, the algorithm starts by placing 500 KLT points [18] at the first frame of the action segment and tracks them [3] until the last frame. The resulting set of KLT trajectories is numerous and in long action segments noisy trajectories could appear. To filter the noise out we extract KLT trajectories where their start/end points are not far from the global position trajectory start/end points.

Several KLT trajectories could be describing the same motion; we cluster the KLT trajectories using Mean-Shift algorithm [4] to obtain the main Local Dynamic trajectories. Mean-Shift is performed using the entry/exit points of the KLT trajectories to avoid

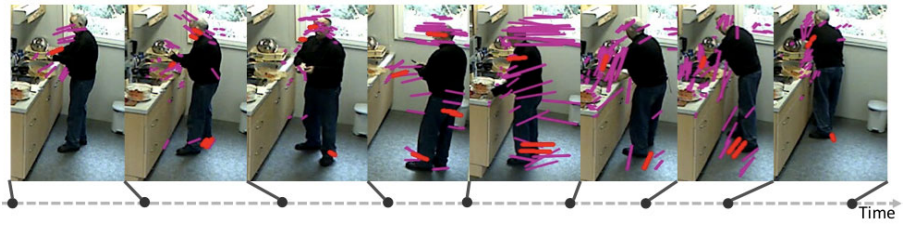


Fig. 1. From left to right a sequence of action segments with the computed KLT trajectories (pink) and Local Dynamics (red) after Mean-Shift clustering

the problem of clustering different trajectory lengths. The advantage of Mean-Shift is that it detects the number of clusters automatically, and filters out small clusters.

In Figure 1 displays a sequence of action segments with the computed Local Dynamics, it can be noticed how the small movements of the person are captured and that the resulting number of Local Dynamics is compact and descriptive.

Other descriptors have been tried (SIFT, SURF), they perform similarly to KLT but with much slower computational speed, while with KLT we process in real time.

3.3 Action Descriptors

An $Action$ is a descriptor that captures global and local information of the trajectories in an action segment:

Globally: $Action_{posStart}$ and $Action_{posEnd}$ are the global person's position at the start/end frames of an action segment.

Locally: $Action_{Length}$ is the average length of the Local Dynamic trajectories and $Action_{Angles}$ is an histogram of the directions of the Local Dynamic trajectories normalized to $\{0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°\}$.

4 Scene Context

In this approach, no information about the scene is known. We learn a scene model composed by scene regions in order to locate actions spatially. The type of regions we are interested in are those where the individual interacts with the scene objects (i.e.,"armchair"). The set of learned regions is called a topology and it is learned by clustering trajectory points.

4.1 Learning a Topology

To build a topology we use the $Action_{posStart}$ and $Action_{posEnd}$ spatial points from a sequence of actions. These points are features describing the locations where the changes of state occur and describe the locations of interaction with the scene. Let $\langle Action_i \rangle$ be a sequence of actions. The set of $InterestPoints$ used is:

$$InterestPoints = \{Action_{i.posStart}\} \cup \{Action_{i.posEnd}\}$$

We perform K-Means clustering over $InterestPoints$. The number of clusters selected represents the level of abstraction of the topology, where lower numbers imply wider regions. Each cluster defines a Scene Region (SR). Finally, we denote $Topology_{levelN} = \{SR_0...SR_N\}$, where each SR_i is labeled with a number for later use.

4.2 Scene Model

A scene model is composed by 3 topologies. They aim at describing coarse, intermediate and specific scene regions. Figure 2 displays 3 topologies composing the model of a scene that we use for experimentation.

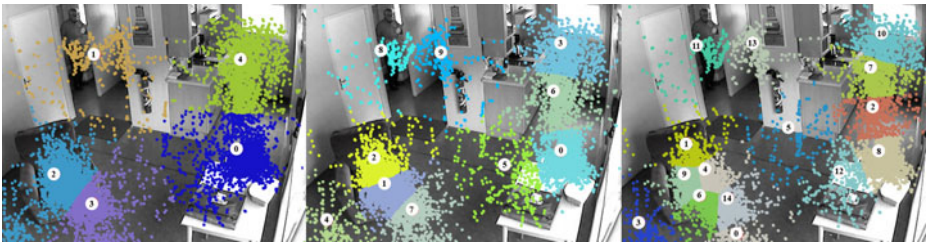


Fig. 2. Computed scene model corresponding to HOME-CARE dataset. From left to right the topologies of level 5, 10 and 15 are displayed. The labeled white dot represent the Scene Region center and the surrounding points the cluster members.

5 Activities

In this section we explain how to combine actions and scene contextual information to discover and model activities. First, we build activity descriptors named Primitive Events that capture an $Action$'s information over the scene. Second, we compute Primitive Event sequences of different levels of abstraction. Third, we combine the sequences to discover activities. And fourth, a discovered activity is modeled to be used by an activity recognition procedure.

5.1 Primitive Events

A Primitive Event (PE) is a descriptor that normalizes the global information of an $Action$ using a scene model. Suppose an $Action$ and a $Topology$ then the PE resulting from $Action$ is defined by its type as:

$$PE = (START \rightarrow END) \ \ (PE\,type).$$

where $START$ and END is the label of the nearest SR (Scene Region) of $Topology$ to $Action_{posStart/posEnd}$ respectively.

$$START = \arg\min_i(dist(Action_{posStart}, SR_i))$$

The $Action$ local descriptors are copied to the PE for later use.

$$(START \rightarrow END)_{Angles/Length} = Action_{Angles/Length}$$

5.2 Primitive Events Sequence

From a sequence of $Actions$, three Primitive Event sequences are computed. One for each $Topology_{Level}$ of a scene model. The motivation of having 3 levels of abstraction of PEs is that with the same set of descriptors, activities of different semantical abstraction levels can be discovered (e.g. "in the kitchen" and "at the kitchen sink").

5.3 Activity Discovery

Independently for each PE sequence described in the previous section, we extract particular subsequences that describe activity. We are interested in two types of subsequences, denoted SPOTTED and DISPLACED.

SPOTTED describes activity occurring within a single topology region (e.g. "Reading in the Armchair"). These are composed by PEs of the same type.

DISPLACED describes activity occurring between two topology regions (e.g. "from Bathroom to Table"). These are composed by a single PE.

Using regular expressions, a $SPOTTED_{A-A}$ is a maximal subsequence of the PEs sequence of the type:

$$(A \rightarrow A)^{+} \tag{1}$$

A $DISPLACED_{A-B}$ is a single PE of the type:

$$(A \rightarrow B), A \neq B \tag{2}$$

The discovered SPOTTED and DISPLACED subsequences are presented to the user as displayed in Fig. 3. The user labels the subsequence that represents an interesting activity at any of the 3 abstraction levels. Adding a label to a subsequence SPOTTED or DISPLACED defines an ACTIVITY SPACE that contains the Primitive Events used to model the activity. An example of how an ACTIVITY SPACE is built is displayed in Fig. 4, where we use the 3 topologies displayed in Fig. 2 to represent a configuration of PE sequences. The example shows how the SPOTTED and DISPLACED subsequences are computed and examples of the ACTIVITY SPACEs defined by labeling as "Preparing Meal" $SPOTTED_{4-4}$ and as "In kitchen table" $SPOTTED_{6-6}$.

5.4 Activity Model

An $Activity$ is modeled by 3 histograms (H_5, H_{10}, H_{15}) and a variable $Activity_{Length}$. Where H_l captures the information of the PEs sequence of $Level_l$ contained in an ACTIVITY SPACE. H_l is an histogram of 2 dimensions. The first coordinate $(global\ feature)$ is the type of a Primitive Event $(S \rightarrow E)$. The second coordinate $(local\ feature)$ is an angle value θ. The count is the accumulation of θ of the primitive events of type $(S \rightarrow E)$ appearing in the PEs sequence of $Level_l$ of the ACTIVITY SPACE.

$$H_l(S \rightarrow E, \theta) = \sum (S \rightarrow E)_i.Angles(\theta) \tag{3}$$

The $Activity_{Length}$ is the average length $(S \rightarrow E)_{Length}$ of the Primitive Events appearing in the ACTIVITY SPACE.

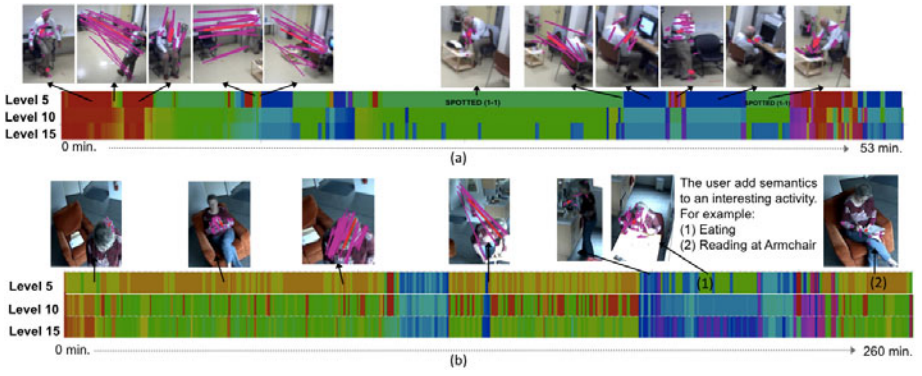


Fig. 3. Activity Discovery of 2 datasets: HOSPITAL (a) and HOME-CARE (b). The scene model used for (b) is displayed in Fig. 2. The colored segments correspond to DISPLACED and SPOTTED subsequences, where the same color is the same subsequence type. For example, SPOTTED(1-1) labeled at the abstraction level 5 (a) corresponds to activity of the person in the chair region. The displayed images are representative actions of the discovered activities.

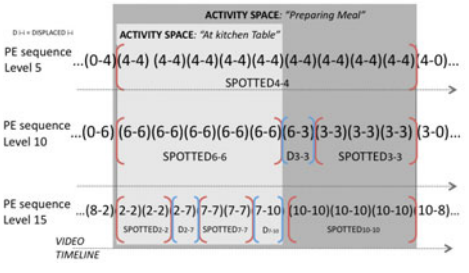


Fig. 4. Example of Activity Discovery sequences. Each layer represents a PE sequence at a level of abstraction. The brackets show the computation of SPOTTED and DISPLACE subsequences, and the ACTIVITY SPACEs are defined by labeling a SPOTTED or DISPLACE subsequence.

6 Activity Recognition

For a new unseen video dataset, we aim at recognizing modeled activities in an unsupervised way. Suppose we have an *Activity* as well as the learned scene model used for modeling *Activity*. We are interested in finding a set of candidate activities that are similar to the modeled one. We explain the steps we use to find candidate activities in a new video:

First, the sequence of actions is computed as described in Section 3.

Second, the Primitive Event sequences are computed, as described in Section 5.2. The difference is that this time we do not compute a new scene model, instead we use the learned *scene_model*. This way, the **PEs of the new video match spatially (PE type) with the PEs used for learning** *Activity*.

Third, the activity discovery process is performed as described in Section 5.3. From the computed set of SPOTTED and DISPLACED subsequences, those that match the subsequence used for labeling $Activity$ are selected. For example, in Figure 2 we label $SPOTTED_{4-4}$ to model "Preparing Meal". For the new video, all $SPOTTED_{4-4}$ appearing at $Level_5$ are selected.

Fourth, the algorithm computes an ACTIVITY SPACE for each SPOTTED or DISPLACED selected in the previous step. From each ACTIVITY SPACE a candidate $Activity'$ is modeled as described in Section 5.4.

Fifth, because of the previous steps, a modeled $Activity$ and a candidate $Activity'$ have a global spatial correspondence. But this does not ensures that both activities are the same (i.e. two different activities may take place at the same spatial location). To measure the similarity we compute $score_{Length}$ and $score_{Histogram}$ and we compare the values to thresholds $T1$ and $T2$. To obtain a binary recognition, an $Activity'$ is the same as $Activity$ if the following statement is $true$:

$$score_{Length} < T1 \wedge score_{Histogram} < T2$$

Activity Similarity: We propose a distance that measures the similarity of all activity descriptors (local and global) by computing 2 scores between the model $Activity$ and the candidate $Activity'$.

The $score_{Length}$ measures the similarity length of the local dynamics:

$$score_{Length} = abs(Activity_{Length} - Activity'_{Length})$$

The $score_{Histogram}$ measures the similarity of the spatial position and local dynamic angles. This score is computed at the different levels of abstraction (capturing the sub-activities similarity) by comparing the 3 histograms of $Activity$ (H_5, H_{10}, H_{15}) with the 3 histograms of $Activity'$. We experimented different similarity measures for multidimensional histograms and finally adopted Earth Movers Distance (EMD):

$$score_{Histogram} = \sum EMD(H_i, H'_i)$$

Thresholds: The recognition thresholds $T1$ and $T2$ are learned using the information of the modeled activities. Let $Model_1 ... Model_i$ be of the same activity, we calculate the mean $score_{Length}$ and $score_{Histogram}$ of all combinations as well as their standard deviation σ_1 and σ_2 . Then $T1$ and $T2$ are defined as:

$$T1 = Average(score_{Length}) + 2 * \sigma_1$$

$$T2 = Average(score_{Histogram}) + 3 * \sigma_2$$

7 Experiments

For experimentation we use videos of 2 different scenes: HOME-CARE and HOSPITAL datasets. Each video contains a single person and are recorded using a monocular video camera (640 x 480 pixels of resolution). HOME-CARE contains 7 elderly people performing non-guided activities in an apartment (in total 24 hours of video). HOSPITAL contains 4 videos of patients performing guided and non guided activities in

a hospital room (3 hours of video). The last dataset is currently being used to study Alzheimer's disease symptoms and the protocol of the guided activities is described by Romdhane et al. [16].

From the discovered activities (i.e. Fig. 3) we label activities shared by most persons. They are selected using DISPLACED and SPOTTED subsequences, where the last ones are the most challenging because of possible activity confusions. For example, "Balance" and "Up/Down" are exercises for measuring the person's stability, both take place same location. The set of labeled activities is displayed in Tables 1, 2.

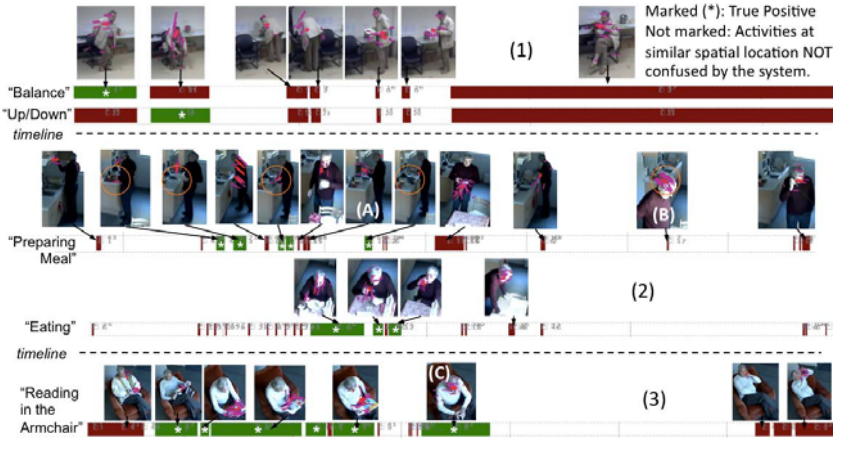


Fig. 5. Marked with (*) are the recognized segments (TP) of the activities: (1) "Balance" and "Up/Down"; (2) "Preparing Meal" and "Eating"; (3) "Reading in the Armchair". The activities are aligned in time. Not marked segments are other -different- activities occurring at the same spatial location not matching with the model. At the top, images representing characteristic actions of the activities. (A) is a False Negative due to lack of motion; (B) is an example of how local motion occurs at the "Preparing Meal" location, but there is no global position matching; (C) is a False positive due to similar motion and global position with the activity model.

Table 1. Recognition results of the selected activities for HOSPITAL dataset

Activity	TP	FP	FN	RT	FT
Balance	3	0	0	100%	1%
Up/Down	3	0	0	100%	4%
Reading at the table	10	1	1	95%	3%
Preparing Coffe	7	1	0	88%	5%
At the Computer	6	1	0	91%	4%
Excercice 1	3	0	0	99%	2%
Excercice 2	3	0	0	99%	1%

Table 2. Recognition results of the selected activities for HOME-CARE dataset

Activity	TP	FP	FN	RT	FT
Eating	31	1	0	97%	7%
Reading in the Armchair	24	4	0	92%	11%
Preparing Meal	52	6	3	83%	6%
Standing at Armchair	11	2	0	95%	5%
Sitting at Eating place	8	0	1	99%	2%
Inside the bathroom	14	2	0	82%	7%
Armchair to Table	32	4	0	96%	1%
Armchair to Kitchen	15	1	0	98%	3%

7.1 Evaluation

The Activity Recognition method depends on the Activity Discovery method, therefore the evaluation of the first one reflects the quality of the discovery procedure.

We evaluate the activity recognition method using cross validation technique. The evaluation is performed recognizing activities in a test video by learning the scene and activity models from the remaining videos. For example, in HOME-CARE, to recognize activities of person G, we compute the scene and activity models using the videos of persons A,B,C,D,E,F. In total 6 experiments are performed (one for each test video).

Performance Measurements: For each dataset an activity ground truth (GT) is manually labeled. The GT describes the intervals of time when an activity begins and ends. The Activity Recognition method returns the intervals of time where an activity is recognized. Each recognized activity instance is compared with the GT and the following measurements are extracted:

True Positive (TP): Number of activity instances correctly recognized.

False Positive (FP): Number of recognized instances not appearing in the GT.

False Negative (FN): Number of instances appearing in the GT not recognized..

Recognition Time (RT): Percentage of time the activity is recognized, over the GT duration of the activity.

False Recognition Time (FT): Percentage of time the activity is recognized while it is not occurring in the GT, over the time the activity is recognized.

Results: Table 1 and Table 2 display the recognition results. In both datasets the method has a very good performance. The FP occurs when the motion of the person while doing different activities is similar and the FN because of the lack of motion. The FT occurs because a person stops an activity without changing of place (i.e. at the end of Eating stays still for a while). To illustrate the complexity of the recognized activities we display some results graphically in Fig. 5.

8 Conclusions

We propose a method to discover and recognize long term activities loosely constrained, in unstructured scenes. The insight of this paper is that it is the first time a complete framework links from the pixel level to complex semantics (“Eating”), using global and local features. Other approaches either use local or global features and the type of activities recognized can be considered as actions (sitting down in a chair).

The contributions are summarized as: An algorithm to learn a scene context (Activity Model); a data structure that combines global and local descriptors (Primitive Events); a method to combine small tasks to discover activities automatically; a method to recognize activities in new datasets. The evaluation results show that it can be used to study activities in home care applications and to perform fast and reliable statistics that can help doctors to diagnose diseases such as Alzheimer. Our future work is going to be the the extension of the approach to perform on-line activity recognition.

References

1. Antonini, G., Thiran, J.: Trajectories clustering in ICA space: an application to automatic counting of pedestrians in video sequences. In: ACIVS 2004. Proc. Intl. Soc. Mag. Reson. Med. IEEE, Los Alamitos (2004)
2. Bobick, A.F., Wilson, A.D.: A state-based approach to the representation and recognition of gesture. IEEE Trans. Pattern Anal. Mach. Intell. 19(12), 1325–1337 (1997)
3. Bouguet, J.Y.: Pyramidal implementation of the Lucas Kanade feature tracker description of the algorithm (2000)
4. Georgescu, B., Shimshoni, I., Meer, P.: Mean shift based clustering in high dimensions: A texture classification example. In: 9th ICCV, pp. 456–463 (2003)
5. Gong, S., Xiang, T.: Recognition of group activities using dynamic probabilistic networks. In: ICC 2003, pp. 742–749 (2003)
6. Hamid, R., Maddi, S., Johnson, A., Bobick, A., Essa, I., Isbell, C.: A novel sequence representation for unsupervised analysis of human activities. A.I. Journal (2009)
7. Haritaoglu, I., Harwood, D., Davis, L.S.: W4: Real-time surveillance of people and their activities. TPAMI 22(8), 809–830 (2000)
8. Hu, W., Xiao, X., Fu, Z., Xie, D.: A system for learning statistical motion patterns. TPAMI 28(9), 1450–1464 (2006)
9. Johnson, N., Hogg, D.: Learning the distribution of object trajectories for event recognition. In: BMVC 1995, Surrey, UK, pp. 583–592 (1995)
10. Khalid, S., Naftel, A.: Classifying spatiotemporal object trajectories using unsupervised learning of basis function coefficients. In: VSSN 2005: Proc. of Intl Workshop on Video Surveillance & Sensor Networks (2005)
11. Laptev, I., Lindeberg, T.: Space-time interest points. In: ICCV, pp. 432–439 (2003)
12. Morris, B.T., Trivedi, M.M.: Learning and classification of trajectories in dynamic scenes: A general framework for live video analysis. In: AVSS 2008 (2008)
13. Owens, J., Hunter, A.: Application of the self-organizing map to trajectory classification. In: VS 2000: Proc. of the Third IEEE Int. Workshop on Visual Surveillance (2000)
14. Piciarelli, C., Foresti, G.L.: On-line trajectory clustering for anomalous events detection. Pattern Recogn. Lett. 27(15), 1835–1842 (2006)
15. Porikli, F.: Learning object trajectory patterns by spectral clustering. In: IEEE International Conference on Multimedia and Expo., ICME 2004, vol. 2, pp. 1171–1174 (2004)
16. Romdhane, R., Mulin, E., Derreumeaux, A., Zouba, N., Piano, J., Lee, L., Leroi, I., Mallea, P., David, R., Thonnat, M., Bremond, F., Robert, P.: Automatic video monitoring system for assessment of alzheimer's disease symptoms. JNHA - The Journal of Nutrition, Health and Aging Ms. No. JNHA-D-11-00004R1 (2011)
17. Calderara, S., Cucchiara, R., Prati, A.: Detection of abnormal behaviors using a mixture of von mises distributions. In: IEEE AVSS (2007)
18. Shi, J., Tomasi, C.: Good features to track. In: IEEE CVPR 1994, p. 593–600 (1994)
19. Zouba, N., Bremond, F., Thonnat, M.: Multisensor fusion for monitoring elderly activities at home. In: AVSS 2009, Genoa, Italy (September 2009)

Ontology-Based Realtime Activity Monitoring Using Beam Search

Wilfried Bohlken, Bernd Neumann, Lothar Hotz, and Patrick Koopmann

FB Informatik, Universität Hamburg, Germany
{bohlken,neumann,koopmann}@informatik.uni-hamburg.de,
hotz@hitec-hh.de

Abstract. In this contribution we present a realtime activity monitoring system, called SCENIOR (SCEne Interpretation with Ontology-based Rules) with several innovative features. Activity concepts are defined in an ontology using OWL, extended by SWRL rules for the temporal structure, and are automatically transformed into a high-level scene interpretation system based on JESS rules. Interpretation goals are transformed into hierarchical hypotheses structures associated with constraints and embedded in a probabilistic scene model. The incremental interpretation process is organised as a Beam Search with multiple parallel interpretation threads. At each step, a context-dependent probabilistic rating is computed for each partial interpretation reflecting the probability of that interpretation to reach completion. Low-rated threads are discarded depending on the beam width. Fully instantiated hypotheses may be used as input for higher-level hypotheses, thus realising a doubly hierarchical recognition process. Missing evidence may be "hallucinated" depending on the context. The system has been evaluated with real-life data of aircraft service activities.

1 Introduction

This paper is about realtime monitoring of object behaviour in aircraft servicing scenes, such as arrival preparation, unloading, tanking and others, based on video streams from several cameras[1]. The focus is on high-level interpretation of object tracks extracted from the video data. The term "high-level interpretation" denotes meaning assignment above the level of individually recognised objects, typically involving temporal and spatial relations between several objects and qualitative behaviour descriptions corresponding to concepts used by humans. For aircraft servicing, interpretation has the goal to recognise the various servicing activities at the apron position of an aircraft, beginning with arrival preparation, passenger disembarking via a passenger bridge, unloading and loading operations involving several kinds of vehicles, refuelling, catering and other activities. Our work can be seen as an alternative to an earlier approach reported in [1], which does not possess the innovative features reported here.

It is well established that high-level vision is essentially an abductive task with interpretations providing an "explanation" for evidence [2-4]. In general, there may be

[1] This work was partially supported by EC Grant 214975, Project Co-Friend.

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 112–121, 2011.
© Springer-Verlag Berlin Heidelberg 2011

several possible explanations even for perfect evidence, and still more if evidence is incomplete or uncertain. Hence any scene interpretation system must deal with multiple solutions. One goal of this paper is to show how a probabilistic preference measure can be combined with an abductive framework to single out the most probable solution from a large set of logically possible alternatives. Different from Markov Logic Networks which have been recently proposed for scene interpretation [5] we combine our logical framework with Bayesian Compositional Hierarchies (BCHs) specifically developed for hierarchical scene models [6].

A second goal is to present an approach where a scene interpretation system is automatically generated from a conceptual knowledge base represented in the standardised ontology language OWL-DL. This facilitates the interaction with reasoners (such as Pellet or RacerPro) and the integration with other knowledge bases.

A third innovative contribution of this paper is a recognition strategy capable of handling highly contaminated evidence and in consequence a large number of alternative interpretations. This is mainly achieved by maintaining up to 100 alternative interpretation threads in a Beam Search [8]. Results show that a preference measure can be used effectively to prune the beam at intermediate stages and to select the best-rating from several final interpretations.

2 Behaviour Modelling

In this section we describe the representation of activity models in a formal ontology. Our main concern is the specification of aggregate models adequate for the activities of the aircraft servicing domain, but also to exemplify generic structures for other domains.

In a nutshell, an aggregate is a conceptual structure consisting of

- a specification of aggregate properties,
- a specification of parts, and
- a specification of constraints between parts.

To illustrate aggregate specifications, consider the aggregate Unloading-Loading-AFT as an example. It consists of three partial activities as shown in Fig. 1, which must meet certain constraints to combine to an unloading or loading activity.

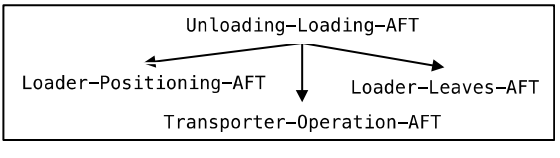


Fig. 1. Part structure of the aggregate Unloading-Loading-AFT

First, temporal constraints must be met: The loader must be placed at the aircraft before any transporter operations can take place, and must leave after completion of these operations. Similarly, spatial constraints must be met, in our domain realised by fixed zones defined for specific servicing activities (e.g. the AFT-Loading-Zone).

Finally, the same physical object occurring in separate parts of an aggregate must be referred to by an identity constraint. Please note that the graphical order of aggregate parts shown in Figs. 1 and 2 does nor imply a temporal order.

As mentioned in the introduction, we have chosen the web ontology language OWL-DL for defining aggregates and related concepts. OWL-DL is a standardised formalism with clear logical foundations and provides the chance for a smooth integration with large-scale knowledge representation and reasoning. Furthermore, the object-centered style of concept definitions in OWL and its support by mature editors such as Protégé[2] promise transparency and scalability. Simple constraints can be represented with SWRL, the Semantic Web Rule Language, albeit not very elegantly.

In OWL-DL, the aggregate Unloading-Loading-AFT is defined as follows:

```
Unloading-Loading-AFT ⊑ Composite-Event ⊓
has-part1 exactly 1 Loader-Positioning-AFT ⊓
has-part2 exactly 1 Transporter-Operation-AFT ⊓
has-part3 exactly 1 Loader-Leaves-AFT
```

The left-hand side implies the right-hand side, corresponding to an abductive reasoning framework. In our definition, the aggregate may name only a single taxonomical parent because of the intended mapping to single-inheritance Java templates. Furthermore, the aggregate must have exactly one part for each hasPartRole. While the DL syntax would allow number restrictions for optional or multiple parts, we found it useful to have different aggregate names for different part configurations and a distinct hasPartRole for each part to simplify the definition of conceptual constraints.

Our aircraft servicing domain is described by 41 aggregates forming a compositional hierarchy. The leaves are primitive aggregates with no parts, such as Loader-Leaves-AFT. They are expected to be instantiated by evidence from low-level image analysis. In addition to the compositional hierarchy, all objects, including aggregates, are embedded in a taxonomical hierarchy which is automatically maintained by OWL-DL. Thus, all activities can be related to a general activity concept and inherit roles such as has-agent, has-start-time, and has-finish-time.

Fig. 2 gives an overview of the main components of aircraft servicing activity concepts. Besides the logical structure, we provide a hierarchical probabilistic model as a preference measure for rating alternative interpretations [6]. In our domain, the model is confined to the temporal properties of activities, i.e. durations and temporal relations between activities, which are represented as Gaussian distributions with the range -2σ .. 2σ corresponding to crisp temporal constraints. Using this model, the probabilities of partial interpretations can be determined and used to control the Beam Search. Unfortunately, OWL-DL and its approved extensions do not offer an efficient way for representing probabilities, so the probabilistic model is kept in a separate database.

Our approach to activity representation can be summarised as follows:

- The main conceptual units are aggregates specifying the decomposition of activities into subactivities and constraints between the components.

[2] http://protege.stanford.edu/

- The representation language for the logical structure is the standardised language OWL-DL which offers integration with high-level knowledge bases and reasoning services, e.g. consistency checking.
- A hierarchical probabilistic model is provided as a preference measure for temporal aggregate properties.

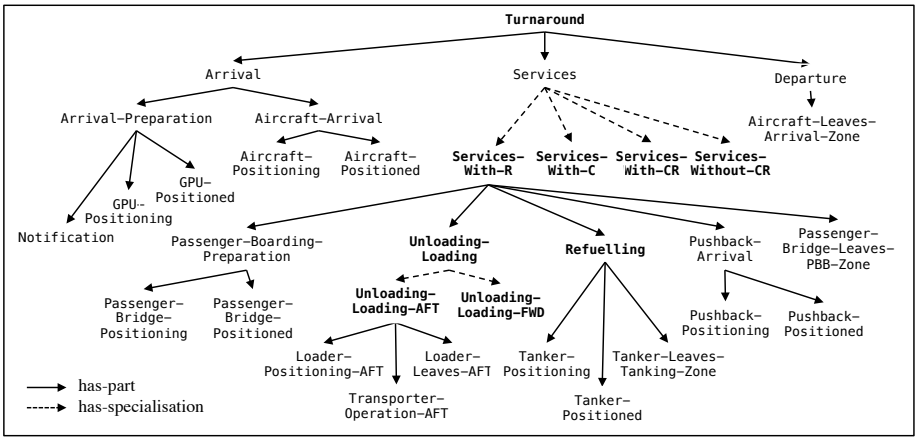


Fig. 2. Activity concepts for aircraft servicing

3 Initialising the Scene Interpretation System from the Ontology

In this section we describe the scene interpretation system SCENIOR, beginning with an overview. In Subsection 3.2 we describe the generation of rules for rule-based scene interpretation and the generation of hypotheses templates as interpretation goals. The interpretation process itself is described in Subsection 3.3.

3.1 System Overview

Fig. 3 shows the architecture of the interpretation system SCENIOR. In the initialisation phase of the system, the conceptual knowledge base, represented in OWL-DL and SWRL, is converted into a JESS conceptual knowledge base, with rules for both bottom-up and top-down processing. Furthermore, hypotheses graphs are created corresponding to submodels of the compositional hierarchy, providing intermediate goals for the interpretation process. The temporal constraints defined with SWRL rules are translated into temporal constraint nets (TCNs) which maintain constraint consistency as the scene evolves. The interpretation process is organised as a Beam Search to accommodate alternative interpretations. A probabilistic scene model, realised as a Bayesian Compositional Hierarchy (BCH), provides a preference measure. For the sake of compactness, TCN and BCH are not described in detail in this paper, see [9] and [6].

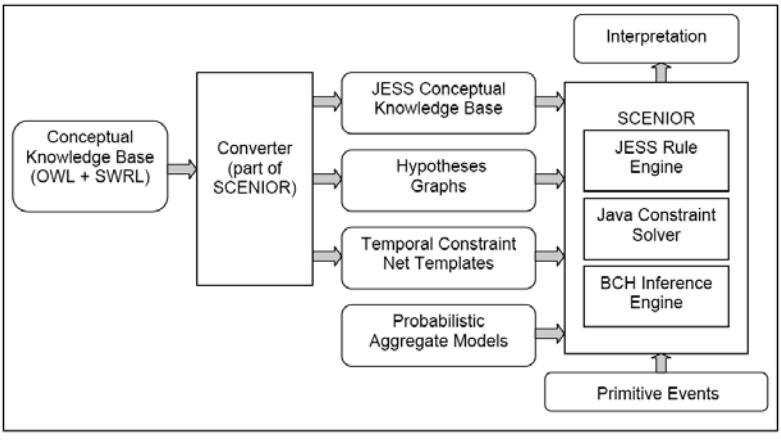


Fig. 3. Main components of the scene interpretation system SCENIOR

3.2 Rule Generation from the Ontology

As shown in [7], scene interpretation can be viewed as a search problem in the space of possible interpretations defined by taxonomical and compositional relations and controlled by constraints. Four kinds of interpretation steps are required to navigate in interpretation space and construct interpretations:

- Aggregate instantiation (moving up a compositional hierarchy)
- Aggregate expansion (moving down a compositional hierarchy)
- Instance specialisation (moving down a taxonomical hierarchy)
- Instance merging (unifying instances obtained separately)

In our framework we create rules for the first three steps, together with some supporting rules. The step "instance merging" is dispensable with the use of hypotheses graphs and parallel search.

Submodels and Hypotheses Graphs. Usually, many models have to be considered in a scene interpretation task. To cope with model variants and to avoid redundancies, we define *submodels* which may be part of several alternative models and are treated as interpretation subgoals (e.g. `Refuelling`). After instantiation, they can be used as "higher-level evidence" for other aggregates (e.g. various kinds of services).

Submodels (marked as `context-free` in the conceptual knowledge base) give rise to hypotheses graphs. Formally, they represent the partonomical structure of a submodel and the equality constraints described with SWRL rules. Their main function is to provide coherent expectations about possible activities. During interpretation hypotheses graphs can be used to "hallucinate" missing evidence and thus continue a promising interpretation thread.

Rules. During the initialisation process, the following interpretation rules are created fully automatically from the ontology:

- *Evidence-assignment rules* assign evidence provided by lower-level processing to a leaf of a hypotheses graph. The premise of the rule addresses a template created for each aggregate (referred to as template-x below).
- *Aggregate-instantiation rules* instantiate a hypothesised aggregate (status hypothesised) if all its parts are instantiated or hallucinated. This is a bottom-up step in the compositional hierarchy and the backbone for the scene interpretation process.
- *Specialisation rules* refine an instance to a more specialised instance. This can happen if more information becomes available as the scene evolves (for example, Vehicle-Inside-Zone may be specialised to Tanker-Inside-Zone).
- *Aggregate-expansion rules* instantiate part of an aggregate if the aggregate itself is instantiated or hallucinated. A separate rule is created for every part of the aggregate. This is a top-down step in the compositional hierarchy. The rule will be invoked if a fact has not been asserted bottom-up but by other means, e.g. by common-sense reasoning (so far this is only rudimentary realised by the hallucination mechanism).

A simplified generic patterns for the evidence-assignement rule is given below, the other rules are defined in a similar way.

```
(defrule aggregate-x-ea-rule
        ?e-id <- (template-x (name ?e)(status evidence))
        ?h-id <- (template-x (name ?h)(status ?status_1))
                (test (or (eq ?status_1 hypothesised)
                          (eq ?status_1 hallucinated)))
        ;;check temporal constraints
=>
        (modify ?e-id (status assigned))
        (modify ?h-id (status instantiated))
        ;;update temporal constraint net)
```

3.3 Interpretation Process

In the initialisation phase of the system, a separate thread is created for each submodel. Each thread has its own independent JESS engine, initialised with all rules and the hypotheses graph corresponding to this submodel.

Now the system is ready to start the interpretation process. It receives primitive events as input and feeds these as working memory elements to every alive rule engine (in the beginning, these are the initialised interpretation threads). Then the rules are applied, eventually leading to instantiated aggregates. These may in turn provide input for higher-level aggregates. If there is more than one activation for an evidence-assignment rule within one thread (i.e. if multiple evidence assignments are possible), this thread is cloned into several threads, one for each possible assignment. A newly created thread is an exact copy of the original thread. This way, a search tree is established which examines all interpretation possibilities in parallel.

So far, we have not yet discussed how to deal with noise, which can either occur in terms of activities not modelled in the ontology, or due to errors of low-level processing. Various kinds of vehicles not taking part in a service or performing some unknown task enter and leave the servicing area throughout a turnaround. Also,

low-level processing in our application is difficult and not at all perfect, hence strange events not corresponding to any real-world activities are delivered as input to SCENIOR. Since there is no way to distinguish correct evidence from noise, as long as both satisfy the constraints, SCENIOR follows both interpretations in parallel, expanding the search tree at each step.

SCENIOR can process in real-time up to ca. 100 threads in parallel on an ordinary PC. Our experiments with airport activities showed that this maximal number of interpretation threads is normally reached while recognising a complete turnaround (see Section 4). At this point, the rating provided by the BCH comes into play and all lowest-rated threads in excess of the maximal beam width are discarded.

Finally, upon termination of the input data stream, all complete turnaround interpretations are ranked using the BCH, and the highest-ranking interpretation is delivered as the result.

4 Experimental Results and Evaluation

In this section we show results of SCENIOR obtained for concrete turnaround scenes at Blagnac Airport in Toulouse. We first illustrate the effects of context-dependent ratings. We then provide a performance evaluation of SCENIOR for 20 turnarounds. The results are explained by the noise statistics of the data which show that the correct interpretation will not always receive the highest rating.

4.1 Illustration of Probabilistic Rating

We now describe the initial phase of a concrete scene interpretation task to demonstrate the effect of the ranking provided by the BCH in a Beam Search. The input data have been obtained from one of the 60 turnarounds by low-level processing of project partners in France and England.

To rate interpretations in this experiment, the probability density of clutter has been set to 0.01 which is less than the typical probability of a regular piece of evidence for a turnaround. Note that the probability density is taken to measure the "probability" of an event. A small constant factor Δt for a time span, over which a density must be integrated, is omitted for clarity. Since the ratings are naturally decreasing with each step and may reach very small numbers, the natural logarithm of a probability is taken, resulting in negative ratings. The primitive events used here belong to an ontology version different from the one presented in Section 2.

In the scene interpreted in this experiment, an Airplane-Enters-ERA event has been generated erroneously by low-level processing for a tanker crossing the ERA (Entrance Restricted Area) shortly before the arrival of the airplane. Fig. 4 left shows the corresponding video frame taken by one of the eight cameras with the crossing tanker in the far background. Two threads are generated, Thread A interpreting this evidence as part of an Arrival, the Thread B as clutter. Later on, the true aircraft arrives (Fig. 4 right), generating an Airplane-Enters-ERA event in the Thread B and a clutter event in a new third thread.

The ratings for the partial interpretations of both alternatives are shown in Table 1. Interpretation A is the erroneous and Interpretation B is the correct one. Initially, the arrival of the GPU sets a context where a vehicle is expected to enter the ERA, hence the crossing tanker is a candidate. But as soon as the true airplane enters, an alternative arises and is favoured because the probabilistic model expects an Airplane-Enters-ERA event 8 minutes after GPU-Enters-GPU-Zone, and the airplane's arrival is closer to that estimate than the tanker's. Note that clutter events not assigned to either of the two interpretations are not shown in the table.

Fig. 4. Snapshots of the ERA (Entrance Restricted Area) after completing Arrival-Preparation. The GPU (Ground Power Unit) is in place. The tanker crossing the ERA in the background (left) causes an erroneous interpretation thread (see text).

Table 1. Initial ratings of the two alternative interpretations

e1 = mobile-inside-zone-86
e2 = mobile-stopped-90
e3 = mobile-inside-zone-131
e4 = mobile-inside-zone-155
est = estimated event

Evidence	Time	Interpretation A	Ranking A	Interpretation B	Ranking B
e1	17:10:31	GPU-Enters-GPU-Zone	0	GPU-Enters-GPU-Zone	0
e2	17:10:32	GPU-Stopped-In...	-2,16	GPU-Stopped-In...	-2,16
e3	17:13:31	Airplane-Enters-ERA	-5,32	Clutter	-2,16
e4	17:20:35	Clutter	-5,32	Airplane-Enters-ERA	-5,09
est	≥17:13:35	Airplane-Stopped...	-6,24		
est	≥17:13:35	Stop-Beacon	-7,71		
est	≥17:20:35			Airplane-Stopped...	-6,01
est	≥17:28:35			Stop-Beacon	-7,48

The table also includes the estimated times of the expected next events Airplane-Stopped-Inside-ERA and Stop-Beacon together with the expected ratings for the competing interpretations. Note that estimated time windows may begin earlier than the actual time, allowing for hallucinated events in the past. Considering that Stop-Beacon will occur after the true aircraft arrival and not at the time expected in Interpretation A, the rating of this interpretation will surely be much lower than the estimated value, further increasing the distance between the right and the wrong interpretation.

The performance of SCENIOR was evaluated for 20 annotated turnarounds, with primitive events provided by low-level image analysis of the project partners. The ontology and the probabilistic model were derived from 32 other turnarounds. Because of the noisy input data, it was necessary to interpret each evidence both as belonging to a turnaround (given that the constraints were satisfied) and as clutter. 17 of the 20 turnarounds resulted in complete interpretations. This was facilitated by Special Vision Tasks with controlled cameras for three crucial events and by "hallucinations" for missing evidence in certain contexts. The three problematic sequences were highly irregular and did not match the conceptual model (e.g. GPU arrival after aircraft arrival). SCENIOR showed a reliable system performance with up to 100 parallel threads (limited by a preset beam width) for partial alternative interpretations, as shown in Fig. 5. It can be seen that alltogether more than 1000 partial interpretations have been initialised, many caused by the context-free submodels which posed interpretation goals throughout the sequence.

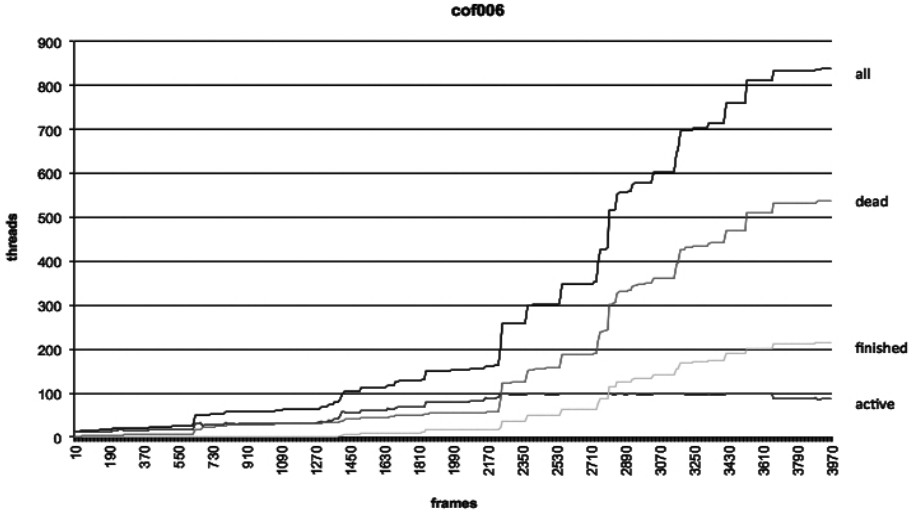


Fig. 5. Number of threads for a typical interpretation process as a function of time (in frames). The maximum nember of active threads has been set to 100.

The recognition rate of subactivities is shown in the table below. It was limited to 75% because of the noisy low-level input data with missing crucial evidence.

Table 2. Correctly recognised subactivities in 20 test sequences

SEQUENCE	1	2	3	4	5	6	8	9	18	25	29	58	59	62	63	66
Arrival	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Passenger-Boarding-Preparation	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Unloading-Loading-AFT	1	1	1	0	1	0	1	1	1	1	0	0	0	0	0	0
Unloading-Loading-FWD				1		1	0	0				1	1		1	1
Refuelling			0	0	0			0		0		0		0		
Pushback-Arrival	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0
Passenger-Bridge-Leaves-PBB-Zone	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Departure	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

To prove the domain-independence of SCENIOR, we also applied the system to activity data of the smart-home environment CASAS[3]. After establishing an ontology for the new domain, SCENIOR recognised all activities without any problems.

5 Conclusions

We have presented the scene interpretation system SCENIOR, designed to work with (i) conceptual knowledge bases expressed in the standardised ontology language OWL-DL, (ii) extended by SWRL rules for constraints, and (iii) supported by a probabilistic scene model for a preference measure. An interpretation strategy employing up to 100 parallel interpretation threads has been realised with JESS rule engines, and successful real-time interpretations have been achieved for noisy aircraft turnaround scenes. The results show that high-level interpretation of activities in low-structured domains and with noisy input data may face formidable ambiguity problems. We believe that the system architecture presented in this contribution has all ingredients to cope with such problems and may prove its worth in diverse applications. A first proof has been obtained in terms of a successful application SCENIOR to the CASAS smart-home environment by simply exchanging the ontology.

References

1. Fusier, F., Valentin, V., Brémond, F., Thonnat, M., Borg, M., Thirde, D., Ferryman, J.: Video Understanding for Complex Activity Recognition. Machine Vision and Applications 18(3), 167–188 (2007)
2. Cohn, A.G., Magee, D., Galata, A., Hogg, D., Hazarika, S.: Towards an architecture for cognitive vision using qualitative spatio-temporal representations and abduction. In: Freksa, C., Brauer, W., Habel, C., Wender, K.F. (eds.) Spatial Cognition III. LNCS (LNAI), vol. 2685, pp. 232–248. Springer, Heidelberg (2003)
3. Shanahan, M.: Perception as abduction: Turning sensor data into meaningful representation. Cognitive Science 29, 103–134 (2005)
4. Moeller, R., Neumann, B.: Ontology-Based Reasoning Techniques for Multimedia Interpretation and Retrieval. In: Kompatsiaris, Y., Hobson, P. (eds.) Semantic Multimedia and Ontologies: Theory and Applications, pp. 55–98. Springer, Heidelberg (2008)
5. Morariu, V.I., Davis, L.S.: Multi-agent event recognition in structured scenarios. In: CVPR 2011, IEEE Conference on Computer Vision and Pattern Recognition (2011)
6. Neumann, B.: Bayesian Compositional Hierarchies - A Probabilistic Structure for Scene Interpretation. TR FBI-HH-B-282/08, Univ. of Hamburg, Dep. of Informatics (2008)
7. Neumann, B., Möller, R.: On Scene Interpretation with Description Logics. In: Christensen, H.I., Nagel, H.-H. (eds.) Cognitive Vision Systems. LNCS, vol. 3948, pp. 247–275. Springer, Heidelberg (2006)
8. Norvig, P.: Paradigms of Artificial Intelligence. Morgan Kaufmann, San Francisco (1992)
9. Bohlken, W., Neumann, B.: Generation of Rules from Ontologies for High-Level Scene Interpretation. In: Governatori, G., Hall, J., Paschke, A. (eds.) RuleML 2009. LNCS, vol. 5858, pp. 93–107. Springer, Heidelberg (2009)

[3] ailab.wsu.edu/casas/

Probabilistic Recognition of Complex Event

Rim Romdhane, Bernard Boulay, Francois Bremond, and Monique Thonnat

INRIA Sophia Antipolis,
France
{Rim.Romdhane,Bernard.Boulay,Francois.Bremond,Monique.Thonnat}@inria.fr

Abstract. This paper describes a complex event recognition approach with probabilistic reasoning for handling uncertainty. The first advantage of the proposed approach is the flexibility of the modeling of composite events with complex temporal constraints. The second advantage is the use of probability theory providing a consistent framework for dealing with uncertain knowledge for the recognition of complex events. The experimental results show that our system can successfully improve the event recognition rate. We conclude by comparing our algorithm with the state of the art and showing how the definition of event models and the probabilistic reasoning can influence the results of the real-time event recognition.

Keywords: Complex event recognition, uncertainty, event description.

1 Introduction

In the literature, many video event recognition systems have been described. However, many challenging problems still remain to obtain a robust recognition because of noise, illumination changes, segmentation issues and occlusions. We propose a constraint-based approach for real-world video interpretation based on probabilistic reasoning for composite event recognition. The main goal is to improve the techniques of video data interpretation taking into account the imprecision and uncertainty of low level data. To reach this goal, we address uncertainty in event modeling and event recognition processes by a combination of logical and probabilistic methods. In summary, the contributions of this paper are: 1. A general framework for video complex event recognition based on a constraint-based approach for video event recognition and a probabilistic reasoning for handling uncertainty. We propose a dynamic linear model for attributes filtering. 2. New event modeling specification: we improve the event description language proposed by [1] and introduce a new probabilistic description based approach to gain in flexibility for event modeling by adding the notion of utility. Utility expresses the importance of sub-events to the recognition of the whole event. The paper is organized as follows: In section 2, we review the related work. In section 3 and 4 we describe the proposed video interpretation framework for complex event recognition. The experiments realized to evaluate the proposed method are shown in section 4. Finally, we present the conclusion in section 6.

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 122–131, 2011.
© Springer-Verlag Berlin Heidelberg 2011

2 Related Work

Many approaches for event representation and recognition have been proposed during the last decade [2,3]. These approaches can be classified into two main categories: probabilistic approaches and symbolic approaches.

The main probabilistic approaches that have been used to recognize video events include Bayesian classifiers [4] and Hidden Markov Models [5,6]. Bayesian classifiers are well adapted to combine observations at one time point, but they have not a specific mechanism to represent the time and temporal constraints between visual observations. For instance, Dynamic Bayesian Networks (DBN) have been used successfully to recognize short temporal actions [7], but the recognition process depends on time segmentation: when the frame-rate or the activity duration changes, the DBN has to be re-trained. Many probabilistic event recognition approaches can handle uncertainty using a probabilistic framework. For instance, in [8] the authors introduce the switching Hidden Semi-Markov Model (S-HSMM) to deal with time duration modeling. This extension attempts to introduce more semantic in the formalism at the cost of tractability.

Symbolic approaches have been largely used to recognize activities. The main trend consists in designing symbolic networks whose nodes or predicates correspond to the boolean recognition of simpler events. Stochastic grammars have been proposed to parse simple actions recognized by vision modules [9]. Logic and Prolog programming have also been used to recognize activities defined as predicates [10]. Constraint Satisfaction Problem (CSP) has been applied to model activities as constraint networks [11]. The symbolic approaches have shown their efficiency in term of complex event recognition. However, these approaches do not handle the uncertainty of the recognition process leading to recognition errors in complex situations. Thus, in this paper, we propose a new constraint based approach for complex event recognition with probabilistic reasoning to improve the recognition performance.

3 Event Description Language

The proposed approach relies on a priori knowledge including the description of the expected objects in the scene, the observed scene, the sensors (*e.g.* fixed video cameras) and the definition of the event models. The expected objects are the physical objects moving in the scene (*e.g.* person, vehicle) which are organized hierarchically (e.g. a car is defined as a sub-type of vehicle). We call domain ontology the description of the expected objects and the set of event models which are predefined by human expert. An event model (fig. 1) is composed of five elements:

-***Physical objects:*** including mobile objects (*e.g.* person, vehicle), contextual objects (equipments, zones).

-***Components:*** the sub-events composing the event.

-***Constraints:*** conditions between the physical objects and/or the components including symbolic, logical, spatial and temporal constraints based on[13].

CompositeEvent (**Up-Go**,
PhysicalObjects ((p: Person), (eq: equipment), (z1: Zone), (z2: Zone), (z3: Zone))
Components ((*c1*: **CompositeState** Person-interacts-with-chair (p, eq, z1) **[1]**),
(*c2*: **PrimitiveState** Person-walking (p, z2) **[0.2]**)
(*c3*: **PrimitiveState** Person-inside-Stop-zone (p, z3) **[1]**)
(*c4*: **PrimitiveState** Person-inside-zoneUsechair (p, z1) **[1]**)
(*c5*: **PrimitiveEvent** Change-posture-stand-to-sit (p) **[1]**)
(*c6*: **PrimitiveEvent** Change-posture-sit-to-stand (p) **[1]**))
Constraints (*c1* **before** *c2* ; *c2* **before** *c3*; *c3* **before** *c4*; *c4* **before** *c5*; *c5* **before** *c6*)
Alarm (Priority "Normal"))

CompositeEvent (**Begin-Guided-test**,
PhysicalObjects ((p1: Person), (p2: Person), (z1: Zone), (z2: Zone), (z3: Zone))
Components ((*c1*: **PrimitiveState** Person-at-Entrance (p1, z1) **[1]**),
(*c2*: **PrimitiveState** Person-at-Entrance (p2, z1) **[1]**)
(*c3*: **PrimitiveEvent** Person-change-to-ZoneUsechair (p1, z2) **[1]**)
(*c4*: **PrimitiveEvent** Person-change-to-ZoneStop (p2, z3) **[1]**))
Constraints (*c1* **meet** *c2*; c3 **meet** *c4)*
Alarm (Priority "Normal"))

Fig. 1. Two Event models: "Up-Go" illustrates a medical exercise for testing the ability of the patient to perform several activities. The model is composed of five steps: (1) the patient is standing at the chair of exercise for a predefined period of time,(2) he/she walks up to a stop zone marked by a red line, (3) returns close to the chair, (4) he/she sits at the chair and (5) gets up. "Begin-Guided-test" describes the beginning of the medical exercise: the nurse and the patient entering together in the room and then going to different places. An utility coefficient was associated to each sub-event.

-***Alarm:*** describes the level of importance of an event.
-***Action:*** describes a specific treatment to be executed when an instance of an event model is recognized:(e.g. launch a specific vision task such as the monitoring of PTZ cameras (zoom on to get better classification of the mobile object) or provide feedback to vision components to enhance the tracking task).

We propose a notion of utility in the definition of the event model by associating a coefficient to each sub-event. Utility which is defined by a human expert expresses the importance of sub-events for the recognition of the whole event. Its range is in the interval]0,1], higher is the utility value higher is the importance of the sub-event in the recognition of the whole event. The value 1 means that the sub-event is required for the recognition. At least one of the sub-events must have a high utility value otherwise the event model will not be considered during the event recognition process.

4 Event Recognition Process

The proposed event recognition algorithm uses as input the tracked mobile objects (extracted by vision algorithms, segmentation, detection, tracking), a priori knowledge of the scene and predefined event models.

The algorithm operates in 2 stages: (i) at each incoming frame, it computes all possible primitive states related to all mobile objects present in the scene, and (ii) it computes all possible events (*i.e.*primitive events, and then composite states and events).

An event model ω is composed of the set of physical objects $\xi(\omega)$, their associated attributes $A(\xi(\omega))$ and the set of sub-events $Se(w)$. The recognition of the event model ω consists of a loop to select a set of physical objects $\xi(\omega)$ then verify the corresponding temporal/spatial/logical constraints $\zeta(\omega)$ until all combinations have been tested. Once a set of physical objects satisfies all constraints we consider that the event is recognized and we generate an event instance p attached to the corresponding event model, the physical object and the recognition time t. The event instance is then stored in the list of recognized events. To prevent from event fragmentation, we consider that if at the previous instant, an event instance p' of the same type (same model, same physical objects) was recognized on a time interval $[t_0,t_1]$ with $|t_1 - t| < \delta$, the two event instances are merged into an instance that is recognized on the time interval $|t_0 - t|$.

During the event recognition process, the system estimates the confidence of primitive states and composite events. The confidence measures describe the quality of the analyzed data based on the temporal coherence of the attribute values.

4.1 Probabilistic Primitive State Recognition

The confidence of primitive state is estimated based on Bayes formula (Eq 1).

$$P(w|\zeta(\omega), Id(\xi(\omega))) = P(\zeta(\omega)|w) \times P(Id(\xi(\omega))|w) \times \frac{P(w)}{P(\zeta(\omega), Id(\xi(\omega)))} \quad (1)$$

We compute then the ratio: $\frac{P(w|\zeta(\omega),Id(\xi(\omega)))}{P(\neg w|\zeta(\neg\omega),Id(\xi(\neg\omega)))}$. with $\neg\omega$ is equal to ω = false. If the ratio value is upper than 1, the primitive state has a high chance to be recognized.

$P(Id(\xi(\omega))|w)$ is the identifier confidence which indicates how well the mobile object $\xi(\omega)$ has been correctly tracked. This probability is obtained by estimating the quality of the tracking process depending on several criteria: the displacement, the appearance and the attribute consistency over the tracking period as described in [18]. The constraint confidence $P(\zeta(\omega)|w)$ is computed depending on the constraint type. There are 2 types of constraint for primitive state: spatial (i.e. a person in a zone) and logical. The confidence of logical constraints (i.e. associate a symbol to a contextual object) is equal to 1 as we consider that the user has a negligible chance to associate a wrong symbol.

The confidence of spatial constraints is obtained by multiplying the confidence of object attributes $P(A(\xi(\omega))|w)$ involved in the constraint with the probability of the constraint to be verified (Eq. 2). For the spatial constraints such as 'inside-zone' or 'close to equipment', we compute the distance dist of the person to the contextual objects (i.e. zone, equipment), more this distance is small more the probability that this constraint is satisfied is close to 1.

$$P(\zeta(\omega))|w) = P(A(\xi(\omega))|w).\frac{1}{\sigma_d\sqrt{2\pi}}exp(-\frac{dist^2}{2\sigma_d^2}) \quad (2)$$

The confidence of mobile object attributes $P(A(\xi(\omega))|w)$ can be retrieved from vision algorithms (detection, tracking, posture recognition...). If this confidence is not directly provided, we compute this confidence using a dynamic linear filter such as Kalman filter algorithm.

- Dynamic Model for Temporal Attributes Filtering
We propose a dynamic linear model for computing and updating the attributes of the mobile objects to deal with tracking errors. This process works in two steps. The first step consists in computing the expected value α_{exp} of an attribute α at the current instant t_c given the estimated value of α and its velocity at the previous time t_p. The second step is to compute the estimated value α_{est} of the attribute based on the previous one. The final value of the attribute $\bar{\alpha}$ is the mean between the expected and the estimated values of the attribute.

$$\bar{\alpha}(t_c) = mean(\alpha_{exp}(t_c), \alpha_{est}(t_c)) \tag{3}$$

$$\alpha_{exp}(t_c) = \bar{\alpha}(t_p) + V_\alpha(t_c)(t_c - t_p); \tag{4}$$

$$V_\alpha(t_c) = \frac{V_{\alpha_c}.R_v + e^{-\lambda(t_c - t_p)}.V_\alpha(t_p)S_{V_\alpha}(t_p)}{S_{V_\alpha}(t_c)}; \tag{5}$$

$$S_{V_\alpha}(t_c) = R_v + e^{-\lambda(t_c - t_p)}.S_{V_\alpha}(t_p) \tag{6}$$

V_{α_c} corresponds to the instantaneous velocity of the attribute α at time instants t_{c-1} and t_c, R_v is the instantaneous realiability of the velocity computed as the mean between the reliability of α at time instants t_{c-1} and t_c. $V_\alpha(t_p)$ is the estimated velocity at the previous time t_p. S_{V_α} is the temporal reliability of velocity. The value $e^{-\lambda(t_c - t_p)}$ corresponds to the cooling function of the previously observed attribute values. It can be interpreted as a forgetting factor for reinforcing the newer infromation.

$$\alpha_{est}(t_c) = \frac{\alpha_c.R_{\alpha c} + e^{-\lambda(t_c - t_p)}.\alpha_{est}(t_p).S_\alpha(t_p)}{S_\alpha(t_c)} \tag{7}$$

$$S_\alpha(t_c) = R_{\alpha c} + e^{-\lambda(t_c - t_p)}.S_\alpha(t_p) \tag{8}$$

Where α_c is the value of the attribute given by vision algorithm and $R_{\alpha c}$ is the reliability of this attribute α_c at time t_c. The reliability estimation of the attribute changes according to its type. For 2D attributes, the reliability is estimated inversely proportional to the distance to the camera accounting that the segmentation errors increase when the object is farther from the camera. For 3D attributes such as 3D position, we create a history H for the attribute values. Based on this temporal history we compute the confidence of the current attribute value using the Gaussian function. The Gaussian parameters (μ, σ) are computed dynamically using the temporal history.

4.2 Hierarchical Uncertainty Propagation

The recognition of a given complex event is triggered only if its last sub-event (called event terminaison) is recognized which avoids an exponential computation.

Thus the algorithm runs in real time since only the events which their terminaison is recognized are processed.

We compute the confidence of the complex event at time t given its probability at previous time t-i and the probability of its sub-events $Se(w)$ (Eq 9). The probability of the event at previous time is weighted by the coefficient $\gamma \in [0, 1]$ which decrease when the last recognized instance of the event is far in time.

$$P^t(w) = P^t(Se(w), w^{t-i}) = P(Se(w)).\gamma P(w^{t-i}). \tag{9}$$

w^{t-i} is the last recognized instance of the event at the previous instant $t - i$.

To improve the temporal constraints verification process, we add the notion of tolerance when processing the temporal intervals comparison. For example to improve the verification of the temporal constraint *'A before B'* we need to find a time t' such that event A has started and ended at time t' and an event B has started after A at time $t'' + \beta$. β is the tolerance coefficient.

After calculating the probability associated to an event, the system can make a recognition decision by accepting events with a probability above a threshold and rejecting others. That is, only the events with high confidence probability are recognized.

5 Experimental Results

We show the effectiveness of using an ontology by applying our algorithm to three different applications: two health care and one airport activity monitoring applications (Fig. 2). Airport application consist in monitoring aircraft and vehicle behaviours whereas health care application consist in monitoring elderly persons observed in an experimental laboratory/hospital room during one hour. The video sequences are challenging in term of illumination changes and shadow. An ontology for airport activity monitoring was built. It is composed of 4 physical object type (person, aircraft, vehicle and zone) and 81 event models: 8 primitive states, 3 primitive events, 24 composite states and 45 composite events. We enhance this ontology for the health care applications by adding new physical objects such as equipment and by modifying some existing primitive events (*e.g.* adding posture attribute to the person). We have reused simple events defined for the airport activity monitoring application and define new event models adapted to the health care. We have tested the event recognition accuracy of our algorithm on health care applications and have compared our results with the approach proposed in [1].

The vision chain algorithms (segmentation, classification, detection and tracking) fails sometimes to provide correct outputs (misclassification, misdetection,...) due to changes of luminosity and noise from video acquisition.

Table 1. Comparison of recognition rate (% R), the false positive (FP) and the false negative (FN) of our algorithm with probabilistic reasoning (probabilistic) and without probabilistic reasoning (deterministic)

Events	♯*videos*	♯*actor*	% R	FP	FN
Deterministic algo					
Up-Go	27	1	59.25	3	11
Begin-Guided-test	9	2	88.9	1	1
Interacts-with-chair	10	1	100	0	0
Stay-at-kitchen	15	1	86	1	2
prepare-meal	8	1	75	1	2
Probabilistic algo					
Up-Go	27	1	92.59	5	2
Begin-Guided-test	9	2	100	1	0
Interacts-with-chair	10	1	100	0	0
Stay-at-kitchen	15	1	93.3	1	1
prepare-meal	8	1	87.5	3	1

Fig. 2. Two health care and one airport activities monitoring applications

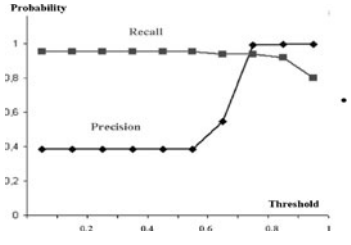


Fig. 3. The performance of primitive states detection was measured depending on the threshold defining the level of likelihood to decide that an event is recognized. With the threshold equal to 0.8, the performance of our system is 0.96 for precision and 0.93 for recall.

We tested the recognitin performance of the primitive state of the proposed system by varying the decision threshold value. The precision and recall rates of the primitive states detection are shown in figure 3. The primitive states are sometimes wrongly recognized due to video noise and vision errors. However, by fixing for all experiments the threshold of detection of primitive states to 0.75 we manage to successfully decrease the false detection of primitive states. By

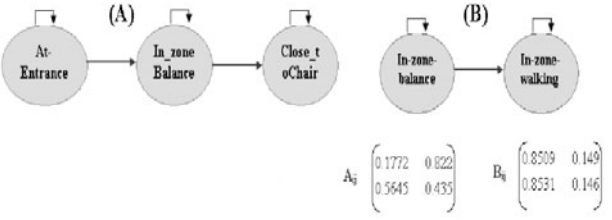


Fig. 4. HMM model for the event (A) 'begin balance exercise': the person enters the room, go to the zone of balance and get close to the chair to begin balance exercise. The event (B) 'change-zone' with the learned transition and observation matrices.

avoiding miss detections of primitive states and using a flexible event description, the proposed system recognizes the complex events with a recognition rate about 92.59 % for the 'Up-Go' event and 100% for the event Begin-Guided-test (Tab. 1). The low rate of false alarms in the case of complex events can be explained by the fact that the event models are very constrained and they are unlikely to be recognized by error.

The comparison (Table 1) shows that the complex event 'Up-Go' in the case of the probabilistic algorithm (92.59 %) is higher than the recognition rate of the deterministic algorithm [1] (59.25%). This can be explained by the fact that the deterministic algorithm fails to recognize the primitive state Person-inside-Stop-zone because the person was not correctly detected. However, the probabilistic algorithm manages to recognize this primitive state and as a consequence the complex event.

Comparison with Probabilistic Method

For comparison with probabilistic method, Bayesian Network models were developed.In our case, the structure of the network is derived from the knowledge about the application domain. For example, logical constraints of sub-events that represent the recognition of a particular event indicate the direct causal link between them. The conditional probabilities were learned using the expectation maximization (EM) algorithm.

In addition, the proposed algorithm was compared with HMMs. We use a left-right HMM for representing the temporal constraints (Fig 4). We model different event such as change zone and change-posture. In the phase of training, we use the sequences of health care database manually classified as belonging in an event. For each event, a HMM is trained. For training, we use the expectation maximization algorithm to estimate the parameters of the HMM model. We use the Forwards-Backwards algorithm for the probability computation.

Table 2 shows the confusion matrices for the proposed algorithm (PA), BNs and HMMs experiments. The proposed algorithm outperform the HMMs and BNS for the event recognition rate. It can be explained by the lack of training data. To have a good recognition rate for the probabilistic approaches like HMMs and BNs we need to have a good parameter estimation. The learning stage need a large and pertinent amount of data.

Table 2. Confusion matrix for the proposed algorithm (PA), the BNs and HMMs.C: Person-sit-at-chair, T: Person-watch-TV, C: Person-interacts-with-Library.

PA			BNs			HMMs		
C	T	L	C	T	L	C	T	L
1	0	0	**.88**	0	.12	**.78**	0	.22
0	**.78**	.22	0	**.78**	.22	0	**.67**	.33
.11	0	**.89**	.33	0	**.67**	.33	0	**.67**
average Pcc=89%			average Pcc=77%			average Pcc=70%		

6 Conclusion

We have proposed a flexible event modeling language and a novel event recognition algorithm to describe and recognize complex video events with probabilistic reasoning to handle the uncertainty. We have proposed a dynamic model for computing and updating the attributes of the mobile objects to deal with tracking errors. We have detailed the estimation of primitive state probability as a Bayesian process and we have computed the confidence of complex event as Markov process taking into account the probability of the event at previous time. A future work consists at deeply studying the uncertainty due to occlusions. Studying more techniques to handle the tracking errors and comparison with those different techniques is also planned. Moreover, a learning stage is still required to learn the algorithm parameters.

Acknowledgment. This work was supported partly by the PACA region, the Sweet-home, and CIU projects. However, the views and opinions expressed herein do not necessarily reflect those of the financing institutions.

References

1. Vu, T., Bremond, F., Thonnat, M.: Automatic Video Interpretation: A Novel Algorithm for Temporal Scenario Recognition. In: The Eighteenth International Joint Conference on Artificial Intelligence, Mexico (2003)
2. Ryoo, M.S., Aggarwal, J.K.: Semantic Representation and Recognition of Continued and Recursive Human Activities. In: International Journal of Computer Vision (2009)
3. Chen, L., Nugent, C.: Ontology-based recognition in intelligent pervasive environments. International Journal of Web Information Systems 5, 410–430 (2009)
4. Oliver, N., Horvitz, E.: A comparison of hMMs and dynamic bayesian networks for recognizing office activities. In: Ardissono, L., Brna, P., Mitrović, A. (eds.) UM 2005. LNCS (LNAI), vol. 3538, pp. 199–209. Springer, Heidelberg (2005)
5. Hoey, J., Bertoldi, P.P., Mihailidis: Assisting persons with dementia during handwashing using a partially observable markov decision process. In: International Conference on Computer Vision Systems, ICVS (2007)
6. Kuettel, D., Breitenstein, M., Van Gool, L., Ferrari, V.: Whats going on? Discovering Spatio-Temporal Dependencies in Dynamic Scenes. In: CVPR (2010)

7. Gong, Xiang, T.: Recognition of group activities using dynamic probabilistic networks. In: The 9th International Conference on Computer Vision (2003)
8. Duong, T.V., Bui, H.H., Phung, D.Q., Venkatesh, S.: Activity recognition and abnormality detection with the switching hidden semi-markov model. In: CVPR (2005)
9. Ivanov, Y., Bobick, A., Mihailidis: Recognition of visual activities interactions by stochastic parsing. IEEE Trans. Patt. Anal. Mach. Intel. 1, 838–845 (2005)
10. Davis, L., Harwood, D., Vidmap, D.: ideo monitoring of activity with prolog. In: AVSS (2005)
11. Reddy, S., Gal, Y., Shieber, S.M.: Recognition of users' activities using constraint satisfaction. In: Houben, G.-J., McCalla, G., Pianesi, F., Zancanaro, M. (eds.) UMAP 2009. LNCS, vol. 5535, pp. 415–421. Springer, Heidelberg (2009)
12. Nevatia, R., Hongeng, S., Bremond, F.: Video-based event recognition:activity representation and probabilistic recognition methods. In: CVIU, vol. 2, pp. 129–162 (2004)
13. Allen, J.F.: Maintaining knowledge about temporal intervals. Communications of the ACM (1983)
14. Getoor, L., Taskar, B.: Introduction to Statistical Relational Learning. MIT Press, Cambridge (2007)
15. Avanzi, A., Bremond, F., Tornieri, C., Thonnat, M.: Design and Assessment of an Intelligent Activity Monitoring Platform. EURASIP (2005)
16. Liao, L., Fox, D., Kautz, H.: Location-based activity recognition using Relational Markov Networks. In: IJCAI (2005)
17. Pentney, W., Popescu, A., Wang, S., Kautz, H., Philipose, M.: Sensor-based understanding of daily life via large-scale use of common sense. In: AAAI 2006 (2006)
18. Chau, D.P., Bremond, F., Thonnat, M.: Robust Mobile Object Tracking Based on Multiple Feature Similarity and Trajectory Filtering. In: VISSAP (2010)

Learning What Matters: Combining Probabilistic Models of 2D and 3D Saliency Cues

Ekaterina Potapova, Michael Zillich, and Markus Vincze*

Automation and Control Institute
Vienna University of Technology
{potapova,zillich,vincze}@acin.tuwien.ac.at

Abstract. In this paper we address the problem of obtaining meaningful saliency measures that tie in coherently with other methods and modalities within larger robotic systems. We learn probabilistic models of various saliency cues from labeled training data and fuse these into probability maps, which while appearing to be qualitatively similar to traditional saliency maps, represent actual probabilities of detecting salient features. We show that these maps are better suited to pick up task-relevant structures in robotic applications. Moreover, having true probabilities rather than arbitrarily scaled saliency measures allows for deeper, semantically meaningful integration with other parts of the overall system.

Keywords: 3D saliency cues, cue integration, probabilistic learning.

1 Introduction

Vision in complex real world scenarios, especially unconstrained segmentation of objects, is a notoriously difficult problem and robotics has realised the importance of attention for robotic systems [23]. Vision in a robot is part of a larger system, which has specific tasks to solve. These tasks allow to derive constraints for the vision system to keep vision problems tractable. These constraints come in the form of attention operators that highlight those parts of the scene most promising for the task at hand.

The range of robotic tasks we consider for this paper includes manipulation, grasping and tracking. We therefore assume objects to appear in various locations and configurations, partly occluded, surrounded by clutter, but typically located on a supporting surface, such as a table or shelf.

What we essentially want is the system to segment objects that can be picked up, or if that is not possible due to clutter or occlusion, we want to at least detect good initial grasp points. These tend to be located somewhere on parts

* The research leading to these results has received funding from the European Community's Seventh Framework Programme under grant agreement IST-FP7-IP-215821 GRASP 2008-2012 and from the Austrian Science Fund (FWF) under project TRP 139-N23, InSitu.

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 132–142, 2011.
© Springer-Verlag Berlin Heidelberg 2011

sticking out from the scene. Pre-grasp manipulation of such parts might free the object from the pile.

Scene segmentation is one of the most researched topics in computer vision, and many different approaches have been proposed [3,4,17], but no generic solution suitable for every task exists. Recent state-of-the-art research in this field suggests the use of seed points to guide the segmentation process [18,14,22]. This leads to the problem of identifying good seed points. Inspired by pre-attentive vision theory recent research has suggested the use of attention points, which can be extracted from saliency maps, using for example a winner-take-all (WTA) algorithm [15].

Many well-known and widely acknowledged models for computation of saliency maps, such as [10,13,12,11,1] use only 2D information about the scene. Itti-Koch-Niebur (IKN) [13] is a generic cue inspired by physiological models, and has proven its efficiency in 2D images. Fig. 1(e) shows the saliency map computed by the IKN cue for the image in Fig. 1(a). Several recent extensions to 3D take advantage of the increased availability of 3D sensing equipment, such as inexpensive laser or time-of-flight sensors and RGB-D cameras [9,16,21,2].

However, classical saliency cues indicate only outliers in the scene, while we require regions with specific task-relevant properties to stand out. One can see this problem as the top-down attention task described in [20,8], while our current goal is to build a bottom-up attention system tuned to identifying particular properties of the visual search space. Finally, given that there is a number of intuitively plausible saliency cues (2D and 3D) there is no model for combining these cues in a principled manner with respect to a given task, without using top-down specific features of required objects or parts of visual space.

We address the above issues with a learning based approach, which can be extended to top-down search tasks in the future. Using the Microsoft Kinect depth sensor we have created an RGB-D image database, consisting of different types of table scenes which are challenging for segmentation, owing to the presence of fully and partially occluded objects, multi-colored objects etc. The database consists of four types of scenes: a) isolated free-standing objects (IFSO), b) occluded objects (OO), c) objects placed in a box (BO) and d) a box containing objects and surrounded by other objects (BOSO). For each type of scenes multiple configurations of objects are presented. In total there are 86 RGB-D images in the database. Task regions were hand-labeled by outlining them with a polygon. In our problem task relevant regions are whole objects. Labeling was done by one person, whose task was to segment objects in the scenes as precisely as possible. For BOSO objects we are interested only in objects situated directly in the box, that is why objects around the box were not labeled at all. Fig. 1(a)-d) show examples of labeled images.

The main novelty of this paper lies in the area of understanding how and what preattentive cues should be combined in a specific robotics task of calculating attention points for segmentation of graspable objects.

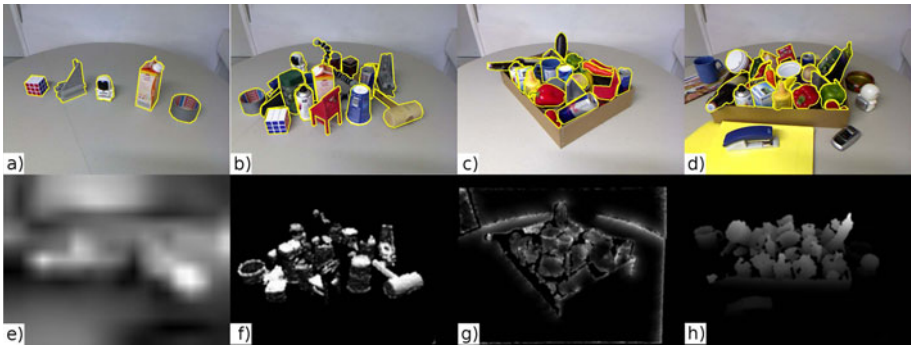


Fig. 1. Four pairs of images and saliency maps (a) and e), b) and f), c) and g), d) and h)). Images a)-d) show examples of images along with labeling for isolated free-standing objects, occluded objects, objects placed in a box and a box containing objects and surrounded by other objects respectively. Images e)-h) show examples of saliency maps based on IKN cue, RSO cue, OE cue and SH cue respectively.

2 Investigated Cues

Inspired by findings from preattentive human vision [6,5,19] we investigated several 3D cues, e.g. based on surface height (SH), relative surface orientation (RSO) and occluded edges (OE) and combined them with cues obtained from 2D information (color, orientation and intensity). As input we have a point cloud $P = \{\mathbf{p}_{ij}\}$ of the table scene, arranged as a rectangular array. I.e. for each image pixel i, j we have a 3D point $\mathbf{p}_{ij}$ together with its RGB color value.

2.1 Surface Height Cue

For the task of picking up objects in a cluttered scene, the simplest way to start grasping is first to pick up all objects that stick out from the clutter. These objects are good candidates for initial grasping attempts, and they should therefore be considered more interesting than the rest. These objects can be pointed out by attention points derived from the surface height preattentive cue, which is based on a height map of the scene. Fig. 1(h) shows the saliency map based on the SH cue for the image in Fig. 1(d).

To calculate height we need to determine a reference, i.e. the supporting plane on which objects rest (e.g. a table). We use RANSAC [7] to determine the plane coefficients $Ax+By+Cz+D = 0$. Note that we can assume from the task context of grasping objects from a table that such a single supporting plance exists. For every point $\mathbf{p}_{ij}$ its distance to the supporting plane $d(i, j)$ is calculated. We set d_{max} to be the distance between the ground plane and the most remote point in the point cloud. Values of the SH cue are calculated according to:

$$SH(i, j) = f(d(i, j)) \tag{1}$$

We furthermore scale height values non-linearly according to $f(x) = ax^2$ to obtain more pronounced salient regions, where a is chosen such that $f(d_{max}) = 1$

2.2 Relative Surface Orientation Cue

The surfaces of objects parallel to the supporting plane often present good candidates for first grasping positions, because they usually indicate top-surfaces of simple objects that can be easily grasped. One of our 3D preattentive cues aims to identify top-surfaces based on surface orientation. We calculate relative orientation between local surface normals and supporting plane normal. Fig. 1,f) shows the saliency map based on the RSO cue for the image in Fig. 1,b).

With $\mathbf{n}$ the normal vector of the supporting plane and $\mathbf{n}_{ij}$ the local surface normal vector determined from a plane fitted to the neighborhood of $\mathbf{p}_{ij}$, values of the RSO cue are calculated according to:

$$RSO(i,j) = |\mathbf{n}_{ij} \cdot \mathbf{n}| \tag{2}$$

2.3 Occluded Edges Cue

The success of the segmentation based on seed points depends a lot on the position of the seed point. The more central the location of the seed point with respect to the object, the higher is the probability that the object will be properly segmented. To this end we designed a cue based on occluded edges. The cue is derived from the depth map of the scene. Fig. 1,g) shows an example of the saliency map based on the OE cue for the image in Fig. 1,c). Using the Canny operator an edge map EM is calculated from the depth map. From every point $p(i_0, j_0)$ that belongs to one of the edges we create a potential field $P(\cdot)$ according to:

$$P(d) = a\frac{1}{d} - b \tag{3}$$

where d is the distance from the current point $p(i,j)$ to the initial edge point $p(i_0, j_0)$ whose influence we are calculating, a is set to 0.5 and b is set to 0.01 in our experiments. The influence is expanded only in directions of decreasing values of the depth map, i.e. the object side of the occluding edge. The value of the point $p(i,j)$ in the OE map is equal to:

$$OE(i,j) = \sum_{\forall(i_0,j_0):EM(i_0,j_0)\geq 0} P(\sqrt{(i-i_0)^2 + (j-j_0)^2}) \tag{4}$$

Finally, OE map is linearly normalized to the range [0,1].

2.4 Cue Combination

We investigated two approaches for cue combination to obtain a final saliency map SM. The first approach is similar to cue combination used in IKN method: the final saliency map SM_S is equal to the sum of individual cues:

$$SM_S(i,j) = w_1 IKN(i,j) + w_2 SH(i,j) + w_3 RSO(i,j) + w_4 OE(i,j), \tag{5}$$

where $\sum w_i = 1$ and we set $w_i = 0.25$.

The second combination method uses multiplication instead of summation, so that we obtain SM_M as multiplication of individual cues:

$$SM_M(i,j) = IKN(i,j)SH(i,j)RSO(i,j)OE(i,j). \tag{6}$$

Fig. 6 e)-h) and Fig. 6 m)-p) show examples of SM_S and SM_M combination types for different types of the scenes.

3 Probabilistic Learning

Combining cues according to Eq. 5 or 6 does not take into account the relative importance of cues. One way to address this is to learn weights for individual cues. Another possibility is to directly learn probabilistic models of cues and then combine these. We used a labeled database to train a probabilistic model of relevance for each saliency cue. For each cue c_i we learned the probability of observing that for given cue a pixel was marked as task relevant salient ($s = true$) - situated inside labeled polygons, or non-salient ($s = false$) - situated outside labeled polygons.

$$\begin{aligned} p(c_i \mid s = true) \\ p(c_i \mid s = false) \end{aligned} \tag{7}$$

We estimated parameters for normal distributions for every type of cue separately and for two types of cue combination: addition and multiplication. Note that our labels essentially mark whole objects, with parts of them being salient (different parts for different cues) and other parts not salient, i.e. we use generic labels, rather than labeling for each cue individually. But this means that estimating the above probabilities directly from the labeled images would essentially learn that inside a region labeled as salient, all sorts of saliency values can appear. But we know that inside labeled regions we are only interested in what makes part of that region salient, not the fact that not all of it is salient. To this end, during estimation of the normal distribution, we weight pixels with saliency I according to $w(I) = I^2$. Note that this measure would not be necessary with marked regions, precisely outlining salient regions for each cue individually. We chose this method however, because we want one set of generic labels, applicable to various different cues, picking up saliency somewhere inside those regions.

Fig. 2 shows estimated normal distributions of saliency values (in the range $[0, 1]$) for the IKN cue constructed for occluded objects scenes and for the RSO cue constructed for a box with objects surrounded by other objects (scenes (a) and b) respectively). We can clearly see that distributions are well separated, allowing distinction of salient from non-salient regions. Note that the choice of a normal distribution is strictly speaking not correct, as values are truncated to the interval $[0, 1]$. Further work will investigate the use of a truncated normal distribution or beta distribution on $[0, 1]$.

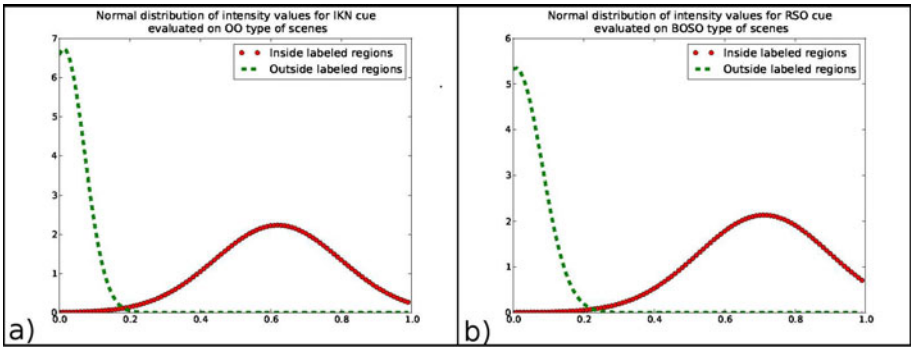


Fig. 2. Normal distribution of saliency values inside and outside labeled regions: a) for IKN cue for occluded objects scenes, b) for RSO cue for a box with objects surrounded by other objects scenes

Following Bayes rule we can then infer the posterior probability of saliency as

$$p(s \mid c_i) = \frac{p(c_i \mid s)\ p(s)}{p(c_i)} = \frac{p(c_i \mid s)\ p(s)}{\sum_{k \in \{t,f\}} p(c_i \mid s = k)} \tag{8}$$

with $p(s)$ being the prior probability of saliency. This could be obtained from top level context information, but is simply assumed 1 here, as we are more interested in the relative differences between cues.

Fig. 3 a)-d) shows the posterior probabilities of salient values for different cues and combinations of cues for different types of scenes. The smaller slope of the IKN as well as OE cues over all types of images indicates that for our type of scenes they are less distinctive than the others. This means that these cues cannot precisely distinguish regions belonging to different objects.

Based on evaluated parameters of the normal distributions, posterior probability images were built for a validation set. The relative sizes of training set and test set were 0.8:0.2.

Fig. 4 shows examples of posterior probability images for different types of cues and cue combinations for the image shown in Fig. 1 d). For an ideal probabilistic image regions of different objects should have the highest saliency values (in our case 1) and be separated from each other. As we can see from Fig. 4 among individual cues RH and RSO cues show the best performance, while combination by multiplication performs better than combination by summation.

As can be seen from the Fig. 4 the IKN cue for such complex scenes assigns high probability values to areas, which do not belong to any object. This is because IKN does not take into consideration 3D spatial positions of the objects, and thus cannot distinguish objects with e.g. similar color. Probability images give us insight into how cues can be combined in terms of top-down attention for a specific task of segmentation for grasping.

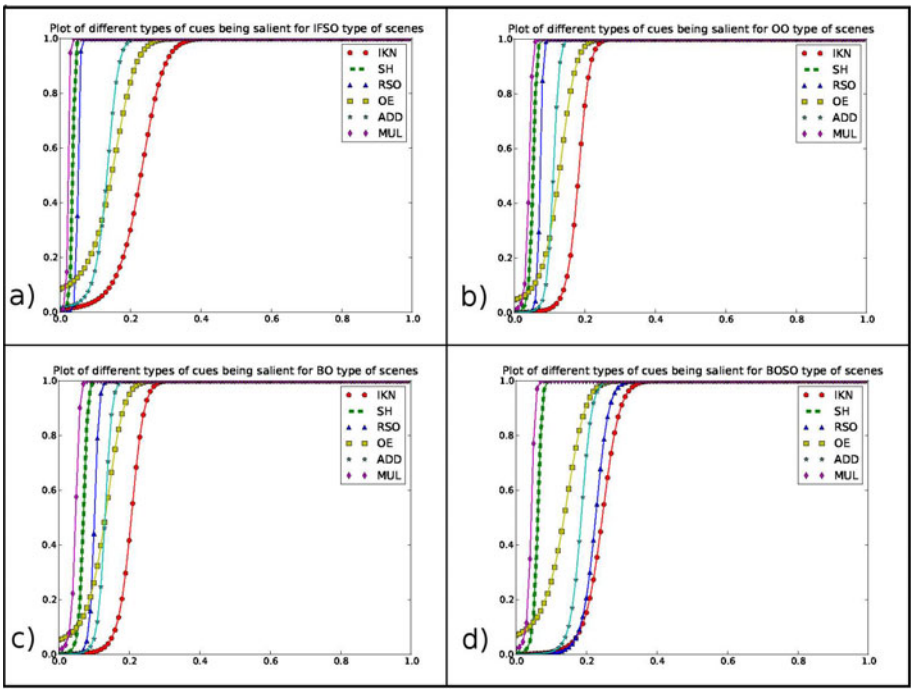


Fig. 3. Probability of salient regions being situated inside labeled regions for different types of scenes: a) isolated free–standing objects b) occluded objects c) objects placed in a box, and d) a box containing objects and surrounded by other objects for different individual cues and cue combinations (for all plots probability via salient value).

4 Evaluation and Results

To evaluate individual cues as well as the cue combinations, we calculated the ratio of first five WTA [15] attention points from the saliency map being situated inside labeled regions of a hold-out set of training images. Averaged results are presented in Fig. 5. Results indicate that especially for complicated scenes with occluded objects 3D saliency cues based on surface height and relative surface orientation perform better than simple 2D cues. Furthermore the cue based on occluded edges did not prove to be a useful cue for our tasks.

Evaluation results go along with distributions obtained from probabilistic learning, while there is still an open question what cue combination is the best for the given task and more experiments on that should be provided.

Fig. 6 shows examples of images with first five attention points indicated in blue color and corresponding saliency maps.

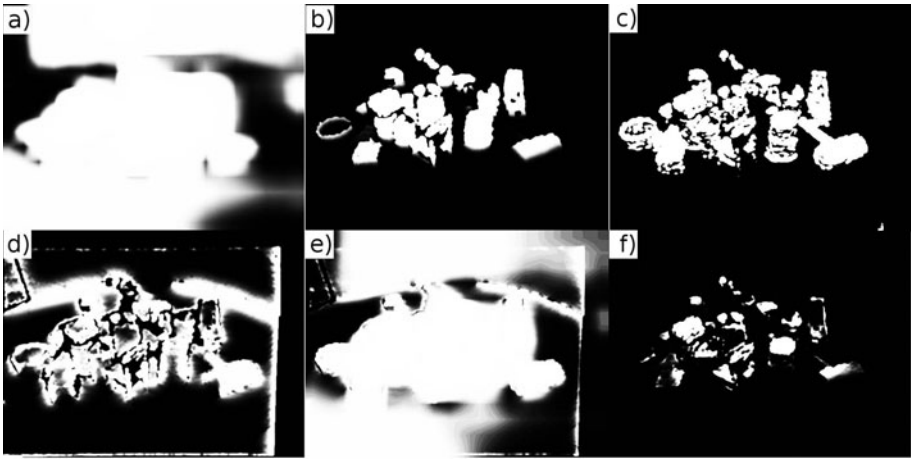


Fig. 4. Posterior probability images for image shown in Fig. 1 d) for a) IKN cue, b) SH cue, c) RSO cue, d) OE cue, e) SM_S cue and f) SM_M cue.

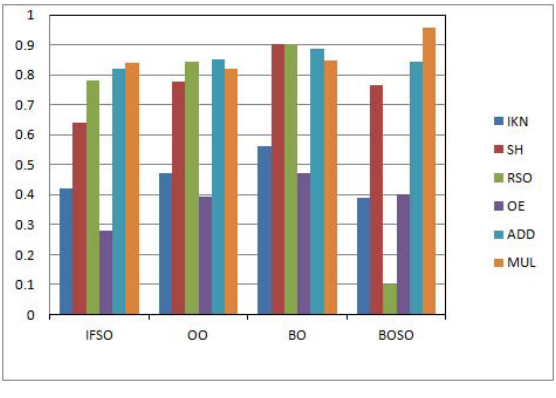


Fig. 5. The ratio of the first five attention points being situated inside different labeled ROIs (IFSO - single standing objects, OO - occluded objects, BO - objects in a box, BOSO - a box with objects which is situated among other objects.

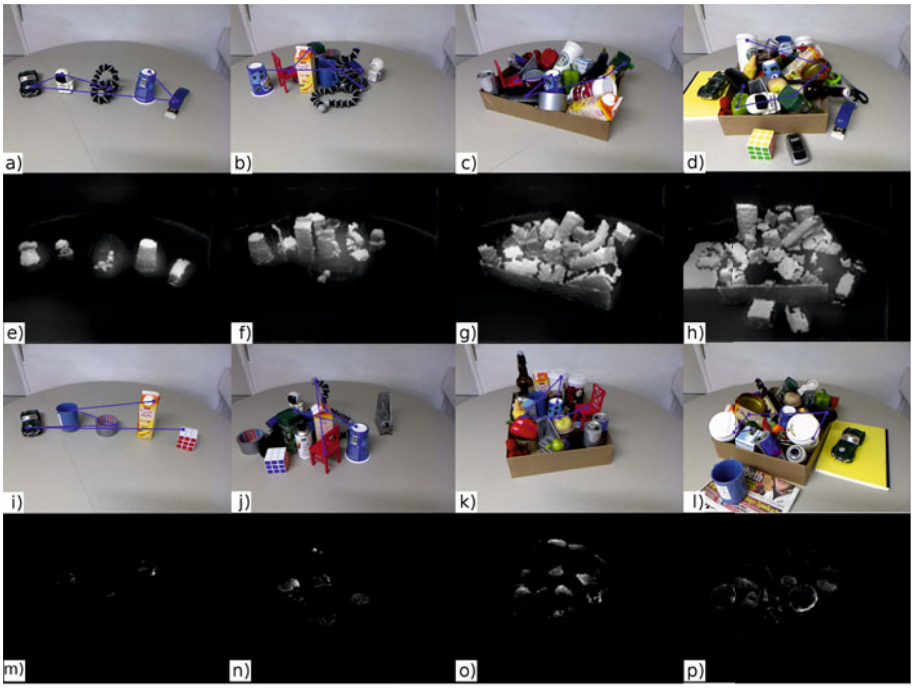


Fig. 6. Results of WTA algorithm for different types of images (left to right: IFSO, OO, BO, BOSO) and corresponding saliency maps: a)-d) present WTA results on SM_S and e)-h) are corresponding saliency maps; i)-l) present WTA results on SM_M and m)-p) are corresponding saliency maps

5 Conclusion and Future Work

In this paper we investigated the use of 3D cues to obtain attention points that can be used as seed points for segmentation of objects for robotic grasping tasks. We implemented three 3D cues to compete against the standard IKN model [13]. Scenes with growing complexity (isolated free-standing objects, occluded objects, objects in a box, and a box containing objects and surrounded by other objects) were evaluated against each cue and two types of cue combination – summation and multiplication. We furthermore estimated probabilistic models over the whole set of images for every type of cue. We could show that height and relative surface orientation cues considerably improve performance in calculating attention points on potential objects for grasping over the standard IKN model [13]. In the most complex cases the combination of both 3D cues gives clearly the best results. This indicates that 3D cues deserve more attention when moving out into the real world with robots.

Our future work will lie in the area of implementing and evaluating more types of 3D preattentive cues and using the results in actual grasping scenarios.

References

1. Achanta, R., Estrada, F., Wils, P., Süsstrunk, S.: Salient region detection and segmentation. In: 6th Int. Conf. on Computer Vision Systems, pp. 66–75 (2008)
2. Akman, O., Jonker, P.: Computing saliency map from spatial information in point cloud data. In: Advanced Concepts for Intelligent Vision Systems, pp. 290–299 (2010)
3. Boykov, Y.Y., Jolly, M.P.: Interactive graph cuts for optimal boundary & region segmentation of objects in n-d images. In: 8th IEEE Int. Conf. on Computer Vision, pp. 105–112 (2001)
4. Comaniciu, D., Meer, P.: Mean shift: a robust approach toward feature space analysis. IEEE Trans. on Pattern Analysis and Machine Intelligence 24(5), 603–619 (2002)
5. Enns, J.T., Rensink, R.A.: Influence of scene-based properties on visual search. Science 247(4943), 721–723 (1990)
6. Enns, J.T., Rensink, R.A.: Sensitivity to three-dimensional orientation in visual search. Psychological Science 1(5), 323–326 (1990)
7. Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Comm. of the ACM 24, 381–395 (1981)
8. Frintrop, S., Backer, G., Rome, E.: Goal-directed search with a top-down modulated computational attention system. In: Kropatsch, W.G., Sablatnig, R., Hanbury, A. (eds.) DAGM 2005. LNCS, vol. 3663, pp. 117–124. Springer, Heidelberg (2005)
9. Frintrop, S., Rome, E., Nüchter, A., Surmann, H.: A bimodal laser-based attention system. Computer Vision and Image Understanding 100, 124–151 (2005)
10. Harel, J., Koch, C., Perona, P.: Graph-based visual saliency. Advances in Neural Information Processing Systems 19, 545–552 (2007)
11. Hou, X., Zhang, L.: Saliency detection: A spectral residual approach. In: IEEE Conf. on Computer Vision and Pattern Recognition, pp. 1–8 (2007)
12. Itti, L., Koch, C.: Computational modelling of visual attention. Nature Reviews Neuroscience 2(3), 194–203 (2001)
13. Itti, L., Koch, C., Niebur, E.: A model of saliency-based visual attention for rapid scene analysis. IEEE Trans. on Pattern Analysis and Machine Intelligence 20(11), 1254–1259 (1998)
14. Ko, B.C., Nam, J.Y.: Object-of-interest image segmentation based on human attention and semantic region clustering. J. Opt. Soc. Am. A 23(10), 2462–2470 (2006)
15. Lee, D.K., Itti, L., Koch, C., Braun, J.: Attention activates winner-take-all competition among visual filters. Nature Neuroscience 2(4), 375–381 (1999)
16. Maki, A., Nordlund, P., Eklundh, J.O.: A computational model of depth-based attention. In: 13th Int. Conf. on Pattern Recognition, pp. 734–739 (1996)
17. Malik, J., Belongie, S., Leung, T., Shi, J.: Contour and texture analysis for image segmentation. Int. Journal of Computer Vision 43(1), 7–27 (2001)
18. Mishra, A., Aloimonos, Y., Fah, C.L.: Active Segmentation with Fixation. In: Twelfth IEEE Int. Conf. on Computer Vision (2009)
19. Nakayama, K., Silverman, G.H.: Serial and parallel processing of visual feature conjunctions. Nature 320, 264–265 (1986)

20. Navalpakkam, V., Itti, L.: An integrated model of top-down and bottom-up attention for optimizing detection speed. In: IEEE Conf. on Computer Vision and Pattern Recognition, pp. 2049–2056 (2006)
21. Ouerhani, N., Huegli, H.: Computing visual attention from scene depth. In: 15th Int. Conf. on Pattern Recognition, pp. 375–378 (2000)
22. Ouerhani, N., Archip, N., Hügli, H., Erard, P.J.: Visual attention guided seed selection for color image segmentation. In: 9th Int. Conf. on Computer Analysis of Images and Patterns, pp. 630–637 (2001)
23. Tsotsos, J.K., Shubina, K.: Attention and Visual Search: Active Robotic Vision Systems that Search. In: 5th Int. Conf. on Computer Vision Systems (2007)

3D Saliency for Abnormal Motion Selection: The Role of the Depth Map

Nicolas Riche, Matei Mancas, Bernard Gosselin, and Thierry Dutoit

University of Mons (UMONS), Faculty of Engineering (FPMs)
20, Place du Parc, 7000 Mons, Belgium
{Matei.Mancas,Nicolas.Riche,Bernard.Gosselin,
Thierry.Dutoit}@umons.ac.be

Abstract. This paper deals with the selection of relevant motion within a scene. The proposed method is based on 3D features extraction and their rarity quantification to compute bottom-up saliency maps. We show that the use of 3D motion features namely the motion direction and velocity is able to achieve much better results than the same algorithm using only 2D information. This is especially true in close scenes with small groups of people or moving objects and frontal view. The proposed algorithm uses motion features but it can be easily generalized to other dynamic or static features. It is implemented on a platform for real-time signal analysis called Max/Msp/Jitter. Social signal processing, video games, gesture processing and, in general, higher level scene understanding can benefit from this method.

Keywords: Saliency, Attention, 3D features, Kinect, Depth map, Gestures.

1 Computational Attention

Computational attention intends to provide algorithms which predict human attention. Attention refers to the process that allows one to focus on some stimuli at the expense of others and it is divided into two complementary influences. Bottom-up attention uses signal characteristics to find the salient objects. Top-down attention uses a priori knowledge to modify the bottom-up saliency. The relative importance of bottom-up and top-down attention depends on the situations [1].

In this paper we focus on bottom-up attention which uses the instantaneous spatial context: it compares a given motion behavior to the rest of the motion within the same frame. Some of the authors providing static attention approaches generalized their models to the time dimension: Dhavale and Itti [2], Tsotsos et al. [3], Parkhurst and Niebur [4], Itti and Baldi [5], Le Meur [6] or Bruce [7]. Motion has a predominant place and the multi-scale temporal contrast of its features is mainly used to highlight important movements. Boiman and Irani [8] provided a model which is able to compare the current movements with others from the video history or a database. Nevertheless, at our best knowledge, none of the motion-based attention models takes into account video depth from a 3D camera while very few use depth for static images [9]. In the next section, the importance of the depth motion extraction is shown in section 2. In section 3, we describe a near real-time motion-based attention model which highlights rare, surprising thus, abnormal motion. In section 4, we show the improvement brought to our model by the use of the depth information, especially in close scenes with frontal view. Finally we discuss and conclude in section 5.

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 143–152, 2011.
© Springer-Verlag Berlin Heidelberg 2011

2 Why 3D Features for Attention?

The 2D motion features extraction from videos can identify the relevant motion within the *(X, Y)* plane. However, they show their limits when movement occurs on the *Z* (depth) axis. We can see an example in Figure 1 where the relevant motion is poorly captured with 2D motion features as the main movement is along the *Z* axis.

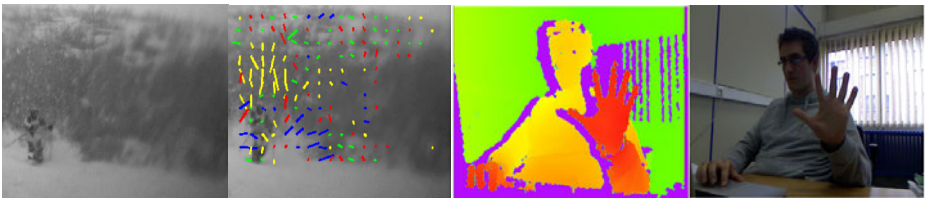


Fig. 1. From left to right: a frame with a skier coming towards the camera (depth – Z axis velocity); 2D motion features (optical flow for X and Y velocity); Depth map (red: close, green: far); RGB corresponding frame

The left image shows the initial video frame while the second image from the left shows the extracted optical flow. The *(X,Y)* motion is properly captured: the snow falling vertically (*Y* axis) above the skier is detected (yellow vertical lines) and the snow moved by the skier on his right on the *X* axis (blue horizontal lines). But the motion of the skier himself is not well described: the image shows several lines of different colors (*X,Y* directions) on the skier while in reality he is coming towards the camera (*Z* axis). This example shows that detection of the motion on the *Z* axis would assign the skier with his real displacement. Obviously, a better feature extraction will also enhance the attention model performance.

The availability of low-cost 3D sensors with active infra-red illumination (as the Microsoft "Kinect" [10]) is an opportunity to easily extract scene depth (Z) information along with classical videos providing *(X,Y)* information. As shown in Figure 1 (third and fourth image from the left), these cameras provide us with RGB video (forth image) and the corresponding depth map (third image). The color map of the third image shows pixels close to the camera in red and pixels far from the camera in green. The pixels in violet are pixels where the information is not available (too close to the camera, too far from it, infra-red shadows, etc.).

The third image from Figure 1 shows that the depth map is homogenous and its quality is well behind the one of classical stereo cameras. This fact is very interesting for the extraction of the movement along the *Z* axis. The implementation for both 3D feature extraction and bottom-up attention computation was carried out on Max MSP [11] using the Jitter and FTM [12] libraries. Max is a platform for real-time signal processing which allows both fast prototyping by using visual programming with libraries supported by a large community and flexibility by the possibility to build additional blocks if needed. Jitter is a library added to Max which provides the possibility to work with matrices, and thus with images and video. FTM is a shared library for Max providing a small and simple real-time object system and a set of optimized services to be used within Max externals. Its capability to handle matrices makes it complementary to Jitter.

3 Attention Model for Motion Selection

The proposed algorithm has three main steps (Figure 2). First, motion features are extracted from the video. Static features could also be extracted, but here only motion-related features were used. A second step is a spatio-temporal filtering of the features at several scales to provide multi-scale statistics. Finally, a third step uses those statistics to quantify at several scales the features' rarity within the video frame.

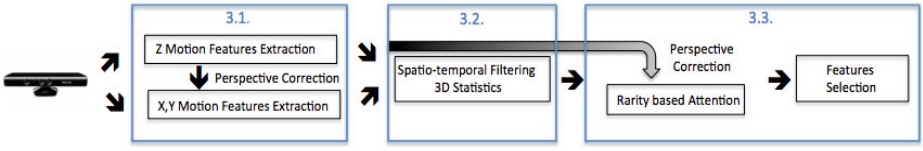


Fig. 2. Block diagram of algorithm used to detect salient motion events using the depth map

3.1 Motion Features Extraction

3.1.1 Part 1: X and Y Extraction

We first work in the plane *(X,Y)*. On the video from the RGB camera, we apply an optical flow algorithm. Optical flow is a measure of the velocity of each pixel between two consecutive frames. (Figure 1, second image). We choose the Farneback approach [13] as it is quite fast and pick Δx and Δy.

3.1.2 Part 2: Z Extraction

We make the difference between two consecutive frames of the depth map to get Δz. Some noise is present on the depth map (violet pixels, Figure 1, third image). The noise of the depth map is eliminated by saturating the shadows and a separation between motion and noise can be achieved by thresholding (Figure 3, first row).

Fig. 3. First row, from left to right: video frame (the hand moves on the *Z* axis); frame differencing of the noisy depth map; frame differencing on the denoised depth map. Second and third row, First column: video frames with the hand going towards the camera (up) and in the opposite direction (down), middle column: feature map of the direction towards the camera, left column: feature map of the direction opposite to the camera.

After noise elimination, the Z axis speed is given by the absolute value of Δz (Figure 3, first row, left image) while the direction is given by the sign of Δz (Figure 3, rows 2 and 3 middle and right).

3.1.3 Part 3: Depth-Based Perspective Correction

Figure 4 (first and second images) shows the perspective problem. The perspective view of a camera will provide wrong apparent sizes of moving objects (people far from the camera seem smaller than people close to the camera) and also wrong apparent speeds of the moving objects (an object moving close to the camera will seem to have a much higher speed than an object moving far from the camera). This perspective view will have negative effects on the speed computation and on the attention model (on the X and Y axes), especially within close camera scenes.

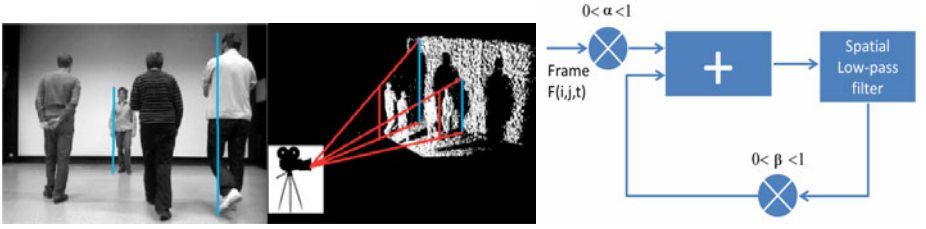


Fig. 4. From left to right: View from the camera: the apparent size of people is different function of their distance with the camera (vertical blue lines); Reconstructed image from the Kinect: people have similar sizes (red vertical lines) but the shadows (apparent sizes) are different (blue vertical lines); Schematization of the 3D low-pass filtering

To remove this effect, we need to compute the distance ("*dist*") of each pixel relative to the camera. This distance will let us know the real objects speed and sizes (Thales theorem). Thanks to the depth map of the Kinect, this depth distance can be directly used to compute the speed and to correct the objects sizes. This is also crucial in attention computation as pixels' rarity depends on objects size. The corrected 3D speed is obtained with the Eq. (1) where Δx and Δy are computed using the optical flow on the RGB video, Δz the frame differencing on the depth map and *dist* comes from the depth map. The features are then discretized into 6 directions (north, south, west, east, front, back) and 5 speeds (very slow, slow, mean, fast, very fast).

$$Speed_{3D} = \sqrt{dist \times \Delta x^2 + dist \times \Delta y^2 + \Delta z^2} \tag{1}$$

3.2 Spatio-Temporal Filtering of the Features

We use low-pass spatio-temporal filters to roughly summarize the statistics of the feature maps (6 directions and 5 speeds). Several scales (both in space and time) of those filters should be applied to the feature maps, but, to keep the algorithm real-time, only two scales were taken into account. To implement a spatio-temporal low-pass filter which will be applied to each of the discretized feature channels (6 directions and 5 speeds), we separated the space and time dimensions. As it can be seen on Figure 4, third image, the frames (F) are first spatially low-pass filtered ($LP_{i,j}$

in Eq. 2). Then, a weighted sum is made on the time dimension by using a feedback and a multiplication factor β<1. This process will tend to provide lower weight to the frames which made the feedback several times (the older ones) because of the β^n in Eq. 2 which will be smaller and smaller when the feedback iteration n will be higher. Our approach only takes into account frames from the past (not from the future).

The neighborhood of the filtering is obtained by changing m (diameter-1 of the spatial kernel $A(h,k)$ (Eq. 3)) and by modifying the β parameter for the temporal part (Eq. 2). If β is closer to 0, the weight applied to the temporal mean will decrease very fast, so the temporal neighborhood will be reduced, while a β closer to 1 will let the temporal dimension be larger. The two filters that we implemented had parameters of m =2 and 8 for the spatial filtering and $\beta = 0.4$ and 0.3 for the temporal filtering.

$$\hat{F}(i,j,t) = \alpha \times \sum_n \beta^n \times LP_{i,j}(F(i,j,t-n))^n \tag{2}$$

where $LP_{i,j}$ is a classical Gaussian spatial low-pass filtering:

$$LP_{i,j}(F(i,j,t-n)) = \sum_{h=-\frac{m}{2}}^{\frac{m}{2}} \sum_{k=-\frac{m}{2}}^{\frac{m}{2}} A(h,k) \times F(i-h, j-k, t-n) \tag{3}$$

3.3 From Feature Detection to Feature Selection

After the filtering of each of the 11 feature maps (6 directions, 5 speeds), the resulting images are separated into 3 bins each. The occurrence probability $P_s(b_i)$ of each bin and for a given scale s is computed as described in Eq. 4:

$$P_s(b_i) = \frac{H(dist \times b_i)}{\sum dist \times \|B\|} \tag{4}$$

where $H(dist \times b_i)$ is the value of the histogram H for the bin b_i (how many times the statistics of a video volume resulting from the 3D low-pass filtering can be found within the frame). The pixels belonging to the bin b_i are previously multiplied by the distance that separates them from the camera: this operation provides a higher weight to pixels which are far from the camera and which belong to objects which have an apparent size smaller than their real size. In that way, the effect of the perspective is cancelled. $||B||$ is the cardinality of the frame (size of the frame in pixels) and $\sum dist$ the sum of distances of the all the pixels. $P_s(b_i)$ is the occurrence probability of the pixels of the bin b_i where the perspective has been cancelled.

Finally, the self-information $I(b_i)$ for the pixels of each bin is computed after taking into account $P_s(bi)$ at the different scales s at which it was computed. This self-information represents the bottom-up attention or saliency for all the pixels of bin b_i (Eq. 5). In order to keep real-time processing, only two scales were used here, so s=2.

$$I(b_i) = -\log\left(\frac{\sum_s P_s(b_i)}{s}\right) \tag{5}$$

Once a saliency map is computed for each of the 6 direction feature maps, they are merged into a *(X,Y,Z)* direction saliency map using the maximum operator but with a coefficient of 2 for the *Z* axis and 1 for the *X* and *Y* axis (Eq. 6). For the speed saliency maps, the speed on the Z axis is incorporated into the 5 saliency maps already existent in 2D (very slow, slow, mean, fast, very fast). Those maps are merged using the same approach as for the direction maps (Eq. 6). The coefficient of 2 is empirical and it is due to the fact that the motion on *(X,Y)* on one hand and the motion on *Z* on the other hand are not extracted using the same approach (optical flow for *(X,Y)* and frame differencing for *Z*).

$$S = Max(2 \times S_Z + S_X + S_Y) \tag{6}$$

The final *(X,Y,Z)* map tells us about the rarity of the statistics of a given video hyper-volume *(X,Y,Z,t)* at two different scales for a given feature. *Rare motion is salient.*

4 Model Validation

4.1 When Using 3D Features?

We represented the speed and direction saliency maps by using a RGB final saliency map. A red dominant means that the speed feature is the most interesting. A cyan dominant means that direction is the most important. A white blob (which is a mix of red and cyan) means that both speed and directions may attract attention. Here we used only 2D motion features on complex real scenes (Figure 5). In the first two images from the first row there is a close scene with a frontal view. The other scenes contain wider and wider views with mostly top views. Surprisingly good results can be found on those wide scenes as shown in Figure 5.

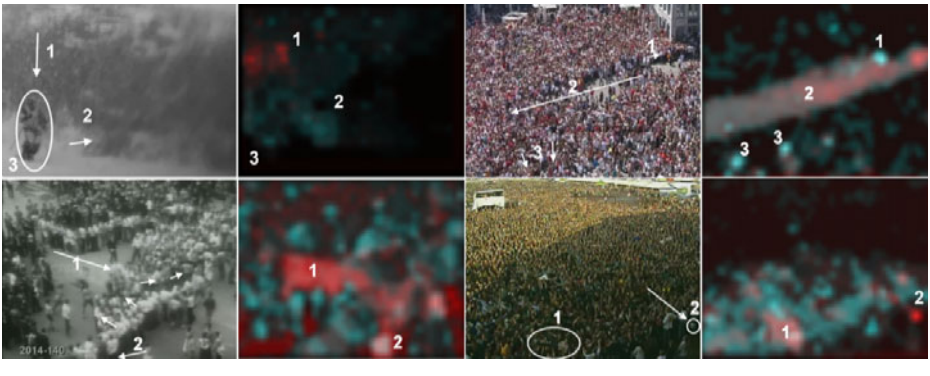


Fig. 5. First and third column: annotated frames, Second and fourth column: color saliency maps. A red dominant means that the speed feature is the most interesting. A cyan dominant means that direction is the important feature.

On the second row, first and second images, we can see that people running towards the others are detected (1) and the person who is faster and with a different

direction (2) is also highlighted. On the first row, third and fourth images, the two people walking against the main central flow (1) are well visible. It is also the case with some people having perpendicular directions (3). Finally in the second row, third and fourth images one person carried by the crowd (1) and a thrown object (2) are also well detected with a higher speed compared with the other moving objects.

Nevertheless, the results are very poor for the first row, first and second image in Figure 5. While the rapidly falling snow (*Y* axis motion) is well detected (1) and the snow pushed by the skier (*X* axis motion) on his right (2) is also detected, the skier himself (3) is not detected at all! The skier is the only moving object on the *Z* axis, thus it is very salient, but as only 2D features are extracted, he is not well detected.

This scene comparison in 2D shows that the more the scene is wide and the camera has a top view, the less important the *Z* axis motion is. Indeed, a top-view will map most of the motion on the *(X,Y)* plane and very small people doing gestures on *Z* (like jumping for example) are almost not detectable in those configurations.

An interesting conclusion is that, while in videosurveillance-like situations (wide field of view, almost top-view) one does not need a precise knowledge about the Z axis, for ambient intelligence and robot-like situations (smaller field of view, frontal view), the knowledge of the Z axis is crucial. This is convenient, as the Kinect sensor horopter is between 25 cm and 6 meters.

4.2 Scenarios Used for Validation

Four people take part to three scenarios. Each one of the scenarios is designed to validate the model along a specific axis (*X*, *Y* and *Z)*. For a first run, for each axis, the purpose is to validate the direction (one of the four people will always be in the opposite direction of the three others) and, during a second run, the goal is to validate the speed (a person will walk faster than the three others). To quantify the model results we define a success rate which is the ratio between the number of frames where the maximum of saliency is located onto the person with a different behavior (in terms of speed or direction) than the others and the total number of valid frames. The valid frames are the frames where the four people are in motion (as only motion features are taken into account). Each of the 6 video runs last around 1 or 2 minutes.

4.3 Validation of the Perspective Correction

To show the contribution of the perspective correction effect, we processed the scenario along the *X* axis (Table 1, left-side) and the one along the *Z* axis (Table 1, right-side) with or without perspective correction.

Table 1. Influence of perspective correction on the *X* axis scenario (left-side) in terms of success rate using the *(X, Y)* saliency maps and on the *Z* axis scenario using the *Z* saliency map

	X-axis Scenario		Z-axis Scenario	
	Sal. Map XY no correct.	Sal. Map XY with correct.	Sal. Map Z no correct.	Sal. Map Z with correct.
Direction	64.6 %	80 %	80.5 %	93 %
Speed	67.1 %	81.3 %	69 %	77 %

In the *Y* axis scenario, there is no perspective effect as all the participants are at the same distance from the camera. Table 1 shows significant improvement for success rate if the perspective correction is applied. Thereafter, we will always use the perspective correction in the following experiments.

4.4 Scenarios Used for Validation

As stated in section 4.2, in a first run of the three scenarios, the goal was to validate the attention-based motion direction selection. In each of the three scenarios, people move at very similar speed but one of the four moves in the opposite direction with respect to the three others. Some results are shown in Figure 6. The white blobs are pointing towards the image areas with a salience higher than 96% of the maximum of the saliency map. Figure 6 shows that the model correctly extracts the man who is walking differently with respect to the main group. Table 2 (left-side) provides the quantitative details of the test on the different sequences:

Table 2. Success rate percentage for salient direction (left-side) and speed (right-side) detection on the three axis using the *(X,Y), (X,Y,Z) and Z* saliency maps

	Direction			Speed		
	Sal. Map XY	Sal. Map XYZ	Sal. Map Z	Sal. Map XY	Sal. Map XYZ	Sal. Map Z
X-axis	80 %	80 %	47 %	81.3 %	77 %	42 %
Y-axis	94 %	90.5 %	51 %	86.2 %	84.4 %	33.3 %
Z-axis	54.3 %	83.3 %	93 %	44 %	71 %	77 %

Fig. 6. Direction scenarios along 3 axes. The white blobs locate the person which has different direction on the X axis (first row), on the Y axis (second row) and on the Z axis (third row).

Table 2 (left-side) provides the success rates for the three axes in selecting the salient person (the one having different behavior compared to the others). The figures are given for the 2D saliency map *(X,Y)*, the 3D saliency map *(X,Y,Z)* and the saliency map of the *Z* axis alone. A first remark is that the *(X,Y)* saliency map performs very poorly on the *Z*-axis (54.3%). A second remark is that the *Z* saliency map performs very well on the *Z* axis (93%) while it performs very poorly on the *X* (47%) and *Y* (51%) axes. Finally, a third remark is about the fusion system (Eq. 6). While the fusion of the *Z* axis saliency with the *(X,Y)* axes saliency seems to work well on the *X*-axis scenario (both have 80%), for the *Y*-axis scenario the *(X,Y)* saliency map has a 94% success rate while the *(X,Y,Z)* saliency map has only a 90.5% success rate. This shows that the use of the information from the *Z* axis on a scenario concerning mainly the *Y* axis slightly decreased the system performance. Concerning the *Z*-axis scenario, the conclusion is the same as for the *Y* axis: the *Z* saliency map provides very good results (93%) while the addition of *X* and *Y* information in a *(X,Y,Z)* saliency map decreases the result to 83.3%. This third remark shows an issue in the empirical fusion strategy proposed in Eq. 6: the *(X,Y)* saliency map works better on the X and Y scenarios than the *(X,Y,Z)* saliency map, while the *Z* saliency map alone works better on the *Z* scenario than the *(X,Y,Z)* saliency map.

4.5 Motion Speed Validation

To validate the speed, we use the same principle as for the direction. In each scenario, one person has a higher speed than the main group. Figure 7 shows that the model extracts correctly the man who is faster than the others on all the axes. Table 2 (right-side) provides the success rates for the three axes in selecting the salient person (the one having different speed compared to the three others). The figures of this

Fig. 7. Speed scenarios along 3 axes. The white blobs locate the person which has different speed on the X axis (first row), on the Y axis (second row) and on the Z axis (third row).

table, even if the overall performances of the speed are slightly lower the ones of the direction, lead to the same remarks than for the previous section: the key role of the *Z* saliency map for the *Z* axis scenario is confirmed and also the fusion issue which slightly decrease the overall system performances.

5 Discussion and Conclusion

We presented a novel near real-time (20 fps for small-sized videos not optimized) bottom-up saliency model. This model uses motion-based 2D or 3D features, but it can be easily extended to other motion features or static features. The use of the depth information has proven, especially in close scenes with frontal view, its crucial importance. The quantitative results on a real scenario show substantial success rate increase in selecting the abnormal motion when the depth information is used along to the classical 2D features. Moreover, the proposed algorithm can handle small motion of the camera without important performance decrease. The fusion issue which leads to a slight performance decrease compared to the best results of the separate *(X,Y)* and *Z* maps can be solved by using a common method for feature extraction for all the exes as a 3D optical flow. The use of depth features opens perspectives for small groups and gesture analysis in frontal views.

Acknowledgements. This work is funded by Numediart (www.numediart.org), Walloon region, Belgium.

References

1. Mancas, M.: Relative influence of bottom-up and top-down attention. In: Paletta, L., Tsotsos, J.K. (eds.) WAPCV 2008. LNCS, vol. 5395, pp. 212–226. Springer, Heidelberg (2009)
2. Dhavale, N., Itti, L.: Saliency-based multifoveated MPEG compression. In: Proceedings of Signal Processing and Its Applications, pp. 229–232 (2003)
3. Tsotsos, J.K., Liu, Y., Martinez-Trujillo, J.C., Pomplun, M., Simine, E., Zhou, K.: Attenting to visual motion. J. of Computer Vision and Image Understandig (2005)
4. Parkhurst, D.J., Niebur, E.: Texture contrast attracts overt visual attention in natural scenes. European Journal of Neuroscience 19(3), 783–789 (2004)
5. Itti, L., Baldi, P.: Bayesian Surprise Attracts Human Attention. Advances in Neural Information Processing Systems 18, 547 (2006)
6. Le Meur, O., Le Callet, P., Barba, D., Thoreau, D.: A Coherent Computational Approach to Model Bottom-Up Visual Attention. PAMI, 802–817 (2006)
7. Bruce, N.D.B., Tsotsos, J.K.: Saliency, attention, and visual search: an information theoretic approach. Journal of Vision 9(3), 5 (2009)
8. Boiman, O., Irani, M.: Detecting Irregularities in Images and in Video. International Journal of Computer Vision 74(1), 17–31 (2007)
9. Ouerhani, N., Huegli, H.: Computing visual attention from scene depth. In: Proc. of Int'l Conf. on Pattern Recognition, vol. 1 (2000)
10. Microsoft Kinect sensor, `http://www.xbox.com/kinect`
11. Max MSP, `http://cycling74.com`
12. FTM library, `http://ftm.ircam.fr/index.php/Main_Page`
13. Farnebäck, G.: Two-Frame Motion Estimation Based on Polynomial Expansion. In: Bigun, J., Gustavsson, T. (eds.) SCIA 2003. LNCS, vol. 2749, pp. 363–370. Springer, Heidelberg (2003)

Scene Understanding through Autonomous Interactive Perception

Niklas Bergström, Carl Henrik Ek, Mårten Björkman, and Danica Kragic

Computer Vision and Active Perception Laboratory
Royal Institute of Technology (KTH), Stockholm, Sweden
{nbergst,chek,celle,danik}@csc.kth.se

Abstract. We propose a framework for detecting, extracting and modeling objects in natural scenes from multi-modal data. Our framework is iterative, exploiting different hypotheses in a complementary manner. We employ the framework in realistic scenarios, based on visual appearance and depth information. Using a robotic manipulator that interacts with the scene, object hypotheses generated using appearance information are confirmed through pushing. The framework is iterative, each generated hypothesis is feeding into the subsequent one, continuously refining the predictions about the scene. We show results that demonstrate the synergic effect of applying multiple hypotheses for real-world scene understanding. The method is efficient and performs in real-time.

1 Introduction

Human interaction with the environment is often done in terms of objects. To that end, one could say that objects define an atomic structure onto which specific semantics, such as an action, can be defined or applied. However, the definition of what constitutes an object is non-obvious and depends on contextual factors of the scene where the task plays a crucial role. Take the example of interacting with a television remote control. One possible task would be to pass it to someone, while a different objective would be to mute the volume. In the first instance it is sufficient to separate the remote from the remainder of the scene, while for the latter a specific button needs to be isolated in order to successfully perform the task. This is one simple example of how additional information such as the task defines the concept of an object. A different example is that of a dinner table where we are likely, in a situation with no other knowledge than visual information, to *assume* the cutlery to be the objects while both the table and the table cloth to belong to the background. This shows that we have over time developed strong priors in terms of what constitutes an object. For a robot that is to operate in manmade environments and cooperate with humans in an effortless and unintrusive manner, it needs to be able to generate and maintain the state of the environment, of which objects are a fundamental building block. This is a challenging task which puts significant demands on both the sensory and information processing system of the agent.

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 153–162, 2011.
© Springer-Verlag Berlin Heidelberg 2011

There has been a significant amount of work on detection and extraction of objects in indoor environments. Being one of the richest sources of information, a significant effort has been aimed at extracting objects from visual data.

In the computer vision community this is referred to as object segmentation [21]. Being a, per definition, severely ill-constrained problem, assumptions about instances and categories of objects are commonly defined and learned *a-priori*. Without instance models or categorical priors, different assumptions have been exploited in the literature: that object edges are aligned with intensity edges, that the object has a different appearance than the background [8], etc. Extending the notion of objects as spatially confined regions in three dimensions, implies that on object occupies a certain volume. This has been exploited in systems such as [22] where a laser scanner is used to extract a depth map of the scene. However, without putting the scene in some form of context, like assuming specific places or part of the environment, the concept of an object is still very ambiguous. One possibility of resolving this is to incorporate human interaction into the system in order to refine the estimation in an iterative manner [18].

Motivated by the success of approaches exploiting an iterative approach, refining the hypotheses over time, we adopt a similar methodology. We developed a fully autonomous system, where the robot interacts with the environment to confirm and improve the generated hypotheses through interaction. Our framework is formalized in terms of maintaining a set of object hypotheses, each feeding information forward in a sequential manner continuously refining the individual estimates. We rely on two object properties in our approach: one is texture by modeling object appearance and the second one is geometry by exploiting rigidity assumption. We show how these are integrated in a probabilistic manner, providing a robust estimate of the object hypotheses. The framework is evaluated in a real-world robotic scenario [4].

The remainder of the paper is structured as follows: in Sec. 2 we detail more related work, while the iterative framework we propose is described in Sec. 3. The appearance and the rigid object hypotheses are described in Sec. 3.1 and Sec. 4 respectively. Experiments are presented in Sec. 5 and in Sec. 6 we conclude.

2 Related Work

An object detection and modeling system used on a robot should be capable of real-time performance and require minimal human intervention. Image segmentation methods, like [9,20], do not consider objects, but rather split the image into coherent parts based on color and intensity information. Methods like [18] successfully segment out objects from the background, but require that the objects is framed to initialize the segmentation. Using a single point as initialization is more suitable for robots, as this only requires the robot to find a probable location of an object in the image. There are several methods exploiting this approach [1,6,17], but [1,17] are computationally expensive. Since we aim at real-time performance, we build upon our original work in [6,7], which, contrary to the other two approaches, has the additional advantage of being easily extendable to handle multiple objects simultaneously, as demonstrated in [3].

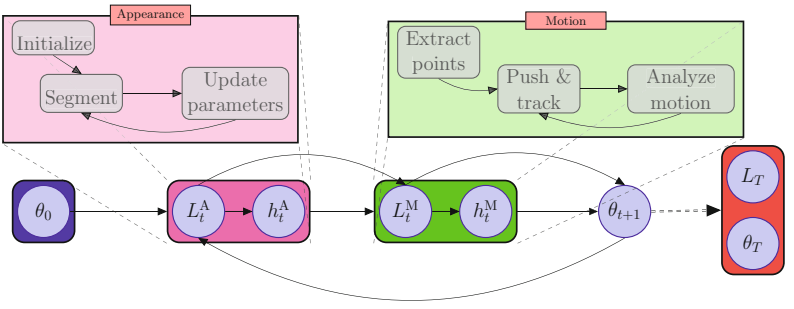


Fig. 1. The proposed model consists of two separate blocks for hypothesis generation, **Appearance** (magenta) and **Motion** (green). The former is initialized by θ_0. Each block employs different assumptions of what constitutes an object based on separate sensory domains generating hypotheses about the state of the scene θ. Each hypothesis is fed into the subsequent block in a sequential manner, iteratively refining the state estimate. Once a stable estimate has been obtained, the result are the objects extracted from the background L_T and their learned appearance models.

Similarly to our approach, [2,18] take advantage of an iterative approach, but require a human expert for guidance. We let the robot itself interact with the scene to gain additional information about its structure. The idea has been used in [13] where a robot segments a scene by pushing objects. However, object positions are assumed known, and if there are several objects moving at the same time, these will be regarded as the same object. [12] assumes rigid object parts and aims to infer kinematic structures of objects through feature tracking. Contrary to our work, they assume only planar motion. Other approaches segment motion using factorization e.g. [10]. These however require a significant motion to be induced on objects compared to our method.

3 System Overview

A diagram of the system is shown in Fig. 1. Each block generates an object hypothesis and by communicating this in a sequential manner, the object hypothesis is iteratively refined. We exploit sensory data from three modalities; color/intensity, depth and motion. Color and intensity are provided directly by the video stream and depth is provided either through stereo reconstruction or sensors such as Kinect [16]. In Sec. 3.1, we detail the approach for building object hypotheses based on appearance and in Sec. 4, we describe the methodology for exploiting interaction by monitoring the relative motion patterns of objects.

Our system is iterative; each hypothesis block taking the current state of the system θ_t as input and generating a labeling L_t of its input modality in terms of object association. This labeling generates a hypothesis about the objects in the scene, updating the state θ_{t+1} of the system. We employ two different hypotheses blocks, each generating labelings based on different assumptions. We will now briefly outline the blocks along with the initial conditions, and the way they interact.

Initial Hypothesis: The parameter set θ_0 is used to initialize the system and holds prior information of the state of the scene, e.g. number of objects, their appearances and positions. This can be provided by different sources, e.g. a human [5] or an attention system [14]. For the experimental evaluation in Sec. 5 the sole assumption is that there is at least one object present in the scene.

Appearance Hypothesis: The first part in the iterative loop consists of the appearance hypothesis, described in Sec. 3.1. The appearance is extracted from regular color images and a sparse depth map from a stereo system. In order to generate a hypothesis, this block requires that at least one pixel in the image is labeled as belonging to an object. We use the method described in [7] to identify this point. The output is a dense labeling L_t^{A} of every pixel in the image and a model of the appearance of each detected object. The labeling L_t^{A} describes a hypothesis h_t^{A} about the number of objects, their location and extent in the image. Further, a model of the appearance of each object is built.

Rigid Motion Hypothesis: From the hypothesis h_t^{A}, we assume one of the two following scenarios: (1) the object hypothesis is correct, or (2) the appearance is not sufficient for disambiguation and have therefore merged several objects into one. By interacting with the scene based on our belief and exploiting the assumption about object rigidity, we generate a sparse labeling L_t^{M}. From this labeling, a hypothesis h_t^{M}, either supporting or opposing h_t^{A}, is generated. In Sec. 4 the details of this approach is explained.

In an iterative manner, we use the motion hypothesis to rectify the appearance model resulting in a dense labeling of objects in the image space, consistent both with the appearance and the motion assumption. Further, we acquire a model of the appearance of each object detected. Due to the ordering of the two hypothesis blocks, we will refer to the motion hypothesis as a means of rectifying the appearance hypothesis. However, each hypothesis generates a labeling in terms of objects and would therefore also work on their own. This forms the central argument of this paper: the complementarity of the different modalities facilitates the disambiguation process. Our sequential framework results in a '*divide-and-conquer*' approach where one hypothesis is used as input to validate the subsequent one.

3.1 Appearance Hypothesis

In [3] we presented a real-time, multi label framework for object segmentation which uses a single point to initialize each foreground hypothesis. Using pixel colors represented as histograms in HSV space, foreground objects are modeled as 3D ellipsoids, while the background is modeled as a combination of planar surfaces, such as a table-top, and uniform clutter. This is done statistically using an iterative EM-framework, that maximizes the likelihood of the corresponding set of model parameters θ_t, given color and disparity measurements. By means of belief propagation, the unknown segmentation is marginalized, which is unlike typical methods using graph-cuts that simultaneously find a MAP solution of both parameters and segmentation. The resulting segmentation L_t^A is the most likely labeling given θ_t after EM convergence. Thus, the method can be viewed

Fig. 2. Examples of scenes where initializing with one point results in both objects captured by one segment (left), and how this is resolved by initializing with two points instead (right)

more as modeling objects than a segmentation approach, which makes it suitable for our particular purpose, where robust estimation of object centroids and extents is of essence.

In cases where the modeling is unable to capture the complexity of the scene, the segmentation system can be expected to fail. In particular, the disparity cue, while helping capture heterogeneously colored objects, also captures other parts of the scene in close proximity to the object. This is true for objects placed on a supporting surface, as the difference in disparity is insignificant in the area around the points of contact. In [3] this is compensated for with the inclusion of a surface component in the background model. This does not solve the problem of two objects standing in close proximity though, which are often captured by the same segment. However, as shown in [3], initializing with one point on each object will often solve this problem, see Fig. 2.

From the current segmentation L_t^A we get a hypothesis h_t^A detailing the composition of the scene. Due to the issues discussed above, we cannot be sure of the correctness of this hypothesis, in particular *whether segments correspond to one or more objects*. To verify the correctness of h_t^A, the hypothesis has to be tested. In the next section, we will show how this can be done by pushing a hypothesis and verifying whether the generated optical flow is consistent with it. If the hypothesis is incorrect, the next iteration of the loop will be informed that the segment in question contains two objects.

4 Rigid Motion Hypothesis

The appearance assumption may fail when objects are placed close to each other, a common situation in manmade environments. One possibility of recovering from such failure would be to push things around, lift them or look at the scene from a different angle, something humans commonly do. Our approach in a robotic setup is to interact with the scene but alter it as little as possible. The cost of e.g. lifting an object may also be high if an incorrect hypothesis leads to the object being dropped. We therefore take the approach of inducing motion by carefully pushing on the object hypothesis. Thus the problem is to infer from motion in the scene, *whether the motion is produced by one or several objects*. For this, we require that objects are rigid and, if more than one object, they move differently with respect to each other.

Fig. 3 shows an example of the different steps of the method. Given a segment from L_t^A, we want to evaluate if it consists of a single or several objects. To

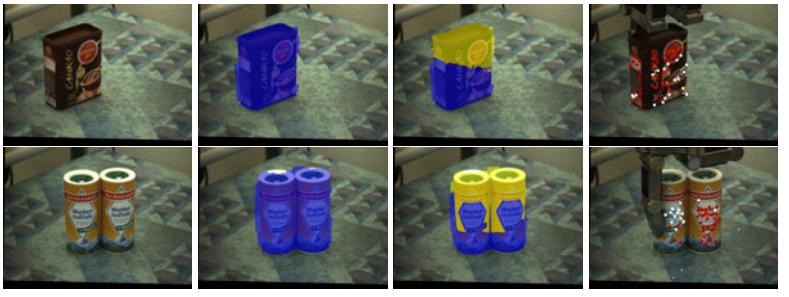


Fig. 3. From left to right: Scene with one or two objects, initial segmentation, clustering based on k-means and one instance of the clustering from motion. Notice that in the two object case, the k-means clustering is not very accurate. Even so, the method is able to realize that there are actually two objects.

that end, we generate a weak hypothesis by clustering, into two centers, the pixels belonging to specific hypothesis in the spatial-color domain. By applying a push onto one of the centers in a direction orthogonal to the vector between the clusters we hope to minimize the risk of similar motion in the case of two objects. For detecting motion, we extract feature points inside the current appearance hypothesis using [19] and track them using the optical flow based approach in [15]. On average between 150 and 300 points are tracked. Furthermore, as motion is analyzed in 3D space, we filter out points for which we have no valid disparity.

4.1 Motion Discrimination

To perform motion analysis, we exploit the fact that *distances between each pairs of points on a rigid object are constant* under translation and rotation of the object, while *distances between pairs of points on different objects will change*. For the following discussion we assume the existence of two objects. We observe that the difference between corresponding point pairs before and after a motion will be zero for point pairs on the same object, and non-zero for point pairs on different objects. We denote the distances at time t with matrix D^t,

$$D^t = \begin{bmatrix} d^t(1,1) & \dots & d^t(1,N) \\ \vdots & \ddots & \vdots \\ d^t(N,1) & \dots & d^t(N,N) \end{bmatrix}, \; d^t(i,j) = \|p_i^t - p_j^t\| \tag{1}$$

and the changes in distance since last update with $Q^t = D^t - D^{t-1}$. Here N is the number of tracked points. Note that the point positions p_i are here expressed in 3D metric coordinates. A column Q_i^t in Q^t can be interpreted as the change in distance from point i to every other point from time $t-1$ to time t. This difference will be zero for points on the same objects, and non-zero for the other points. Hence all vectors $Q_i^t \in \mathbb{R}^N$ belonging to the same object will have zeros for the same dimensions. Thus a vector Q_i^t resides in one of two distinct subspaces, $\mathcal{V}_K$ and $\mathcal{V}_L$, corresponding to objects O_K and O_L. We know empirically that the eigenvectors,

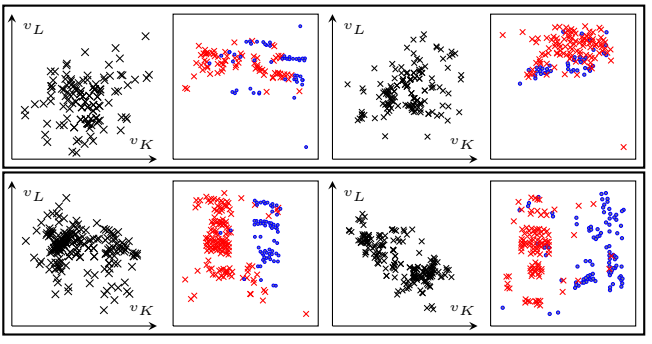


Fig. 4. The black points is the projection of the motion data onto its two dominant eigen-components, $v_K v_L$, while the red and blue points are projected back to the image space, and indicate the associations based on clustering in the motion space. The top box shows two examples of scenes containing one object, while the bottom box exemplifies the occurrence of two objects.

v_K and v_L, corresponding to the two largest eigenvalues of Q^t, reside in $\mathcal{V}_K$ and $\mathcal{V}_L$ respectively. We thus project each Q_i^t on these eigenvectors:

$$q_i^t = [v_K \; v_L]^T \, Q_i^t \tag{2}$$

If there are in fact two objects and the signal-to-noise ratio is sufficient, the points in the resulting 2D motion space will form two clusters. However, as Fig. 4 shows, looking at this space it can be hard to distinguish between the case of one object and two objects. Therefore, we first cluster points in the *motion space*, which is done using a two component Gaussian mixture model, and then look at the clustered points in the original *image space*, to verify whether the clustering in the motion space made sense. In image space, points from two objects will be partitioned in distinct clusters, while in the case of one object, such a pattern is not observable (see Fig. 5). The reason for this is that in the ideal case, if one object is observed, Q^t would be a zero matrix. Therefore any non-zero entries in Q^t will be the result of noise.

To distinguish between the one and two object cases, we look at image point distances; intra- (Eq. 3), and inter cluster distances (Eq. 4).

$$e_{K,K}^t = \frac{1}{K^2} \sum_{i \in O_K, j \in O_K} \|p_i^t - p_j^t\|, \;\; e_{L,L}^t = \frac{1}{L^2} \sum_{i \in O_L, j \in O_L} \|p_i^t - p_j^t\| \tag{3}$$

$$e_{K,L}^t = e_{L,K}^t = \frac{1}{KL} \sum_{i \in O_K, j \in O_L} \|p_i^t - p_j^t\| \tag{4}$$

Here K, L denote the number of points in respective clusters. We then compute the ratio $r_e^t = (e_{K,K}^t + e_{L,L}^t)/(2e_{K,L}^t)$. In the case of one object, the classes are more or less randomly distributed over the point set in image space. Therefore the difference between intra- and inter cluster distances will be smaller than in the case of two objects, where the classes in the point set are grouped. Hence, r_e^t will be smaller in the case of two objects compared to one object.

Fig. 5. Examples of the clustering of the tracked points. While there is no real pattern in the case of one object, in the other case the points are clearly grouped in one left and one right cluster. Note that the second frame in the upper row and third frame in second row have all points except one assigned to one GMM component. These cases, which occur due to outliers, we do not include in the updates of Eq. 5.

The ratio r_e^t will in turn decide h_t^M. For robustness, we integrate observations from several consecutive time steps and update the robot's belief about the current state to produce the final hypothesis. We model the robot's belief μ of the assumption "there are two objects in the scene", with a beta distribution:

$$p(\mu|a,b,l,m) \propto \mu^{a+l-1}(1-\mu)^{b+m-1} \tag{5}$$

Here a and b are hyper parameters, while l and m are based on the history of observations. An observation agreeing with the statement will give an update to l, and a disagreeing observation to m. We update m and l as follows:

$$l \leftarrow l + f_l(r_e^t),\ m \leftarrow m + f_m(r_e^t);\quad f_l(x) = [1+\mathrm{e}^{-u(v+x)}]^{-1},\ f_m(x) = 1 - f_l(x)$$

Here u and v are parameters governing the offset and steepness of the sigmoid function. Using $u = 20$ and $v = -0.8$ gives us satisfactory results. Furthermore, we set the hyper parameters $a = b = 1$, which gives a uniform prior on μ.

To summarize: In Sec. 3.1 and 4.1 two methods are presented using (1) appearance and (2) motion, to create a partitioning of the scene in terms of objects. While (1) results in a dense labeling, (2) produces a sparse segmentation in terms of pixels in an image. While they both can be applied as stand-alone methods, we in this work greatly benefit from integrating both methods in an iterative scenario, thus exploiting both appearance and motion in the segmentation.

5 Experiments

The experimental setup consists of an Armar III active stereo head with foveal and wide angle cameras, a KUKA robot arm with 6 DoF and a Schunk Dexterous Hand with 7 DoF. For the experiments the foveal views are used. We use open loop control for pushing.

In order to evaluate the added benefit of including the motion hypothesis, we run experiments on scenes containing one or two objects for which the appearance model predicts a single object. Objects were placed in close proximity at

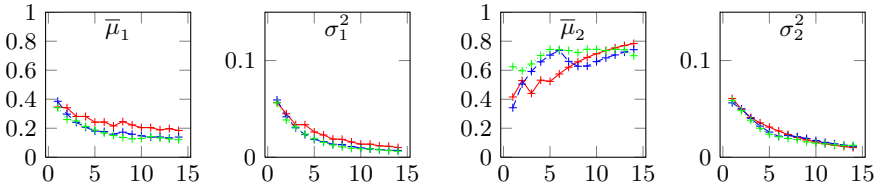


Fig. 6. The plots show three typical examples of the progression through 15 frames for the mean and variance of the beta distribution. The left and right plots show the behavior for one and two objects respectively.

random locations on a surface with the requirement that in case of two objects, at least 1/4 of the tracked points belong to each object. The scene was initialized with one segment as described in [3], and after convergence the motion modeling was initiated. A weak hypothesis and a pushing motion, as described in Sec. 4, was generated, feature points extracted, and the pushing motion executed. The movements of the feature points were tracked for 15 frames and classified offline.

Fig. 6 shows some plots of the mean and variance of the Beta-distribution for some example scenes. We treat an example as correctly classified, if the mean of the beta-distribution reaches above 0.7 for two objects, and below 0.3 for one object. The thresholds where set by experimental validation. 50 experiments were run, evenly distributed between the two classes and each with different configurations of objects. For these experiments, the classification rate was of 92 %. The incorrectly classified scenes were due to e.g. interference with the robot finger and objects moving too similarly to each other. However, these could potentially be rectified through another iteration through the framework. In most cases classification according to above could be done long before the 15 frames had been processed. The tracked points in most cases only have to move on average less than 0.5 cm for a correct classification. This means that for an online scenario the robot could stop the push motion as soon as it has made a classification, update θ_{t+1} and feed this back to the appearance method.

6 Conclusions

To interact with an unknown unstructured environment, a robot has to reason about what constitutes an object. While easily solved by humans, given no prior information this is a challenging task for a robot. Appearance based object segmentation methods are bound to fail due to the problem being inherently ill-posed. The problem is often solved by letting a human correct the errors, which is unfeasible in a robotic scenario. In this paper we proposed a system for object segmentation driven by one appearance based and one motion based method. The first produces a hypothesis about the scene. The robot then seeks to validate this hypothesis itself by pushing it, and analyzing whether the resulting motion is compatible with it, using an assumption of rigid objects. The result is in turn fed back to the appearance based method for producing a more correct segmentation. We have shown that the method performs successfully in the vast majority of the cases in our experiments with very small impact on the scene.

References

1. Bagon, S., Boiman, O., Irani, M.: What is a good image segment? A unified approach to segment extraction. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part IV. LNCS, vol. 5305, pp. 30–44. Springer, Heidelberg (2008)
2. Batra, D., Kowdle, A., Parikh, D., Luo, J., Chen, T.: icoseg: Interactive co-segmentation with intelligent scribble guidance. In: CVPR, pp. 3169–3176 (2010)
3. Bergström, N., Björkman, M., Kragic, D.: Generating Object Hypotheses in Natural Scenes through Human-Robot Interaction. In: IROS (2011)
4. (July 2011), http://www.csc.kth.se/~nbergst/videos
5. Johnson-Roberson, M., Bohg, J., Skantze, G., Gustafson, J., Carlson, R., Rasolzadeh, B., Kragic, D.: Enhanced Visual Scene Understanding through Human-Robot Dialog. In: IROS, San Francisco, USA (2011)
6. Björkman, M., Kragic, D.: Active 3d scene segmentation and detection of unknown objects. In: ICRA, pp. 3114–3120 (2010)
7. Björkman, M., Kragic, D.: Active 3d segmentation through fixation of previously unseen objects. In: Proceedings of the British Machine Vision Conference, pp. 361–386. BMVA Press (2010)
8. Boykov, Y., Veksler, O., Zabih, R.: Fast approximate energy minimization via graph cuts. IEEE Trans. Pattern Anal. Mach. Intell. 23(11), 1222–1239 (2001)
9. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Trans. Pattern Anal. Mach. Intell. 24(5), 603–619 (2002)
10. Goh, A., Vidal, R.: Segmenting motions of different types by unsupervised manifold clustering. In: Proceedings of CVPR, pp. 1–6 (2007)
11. Johnson-Roberson, M., Skantze, G., Bohg, J., Gustafson, J., Carlson, R., Kragic, D.: Enhanced visual scene understanding through human-robot dialog. In: 2010 AAAI Fall Symposium on Dialog with Robots (2010)
12. Katz, D., Brock, O.: Manipulating articulated objects with interactive perception. In: Proceedings of the IEEE ICRA, Pasadena, USA, pp. 272–277 (2008)
13. Kenney, J., Buckley, T., Brock, O.: Interactive segmentation for manipulation in unstructured environments. In: ICRA 2009, USA, pp. 1343–1348 (2009)
14. Kootstra, G., Bergström, N., Kragic, D.: Fast and automatic detection and segmentation of unknown objects. In: Proceedings of the IEEE-RAS International Conference on Humanois Robotics, Nashville, TN, December 6-8 (2010)
15. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: IJCAI, pp. 674–679 (1981)
16. Microsoft Corp. Redmond WA. Kinect for Xbox 360
17. Mishra, A.K., Aloimonos, Y.: Active segmentation. I. J. Humanoid Robotics 6(3), 361–386 (2009)
18. Rother, C., Kolmogorov, V., Blake, A.: "GrabCut": interactive foreground extraction using iterated graph cuts. ACM Trans. Graph. 23(3), 309–314 (2004)
19. Shi, J., Tomasi, C.: Good features to track, Tech. report, Ithaca, USA (1993)
20. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 22(8), 888–905 (2000)
21. Stein, A.N., Stepleton, T.S., Hebert, M.: Towards unsupervised whole-object segmentation: Combining automated matting with boundary detection. In: CVPR. IEEE Computer Society, Los Alamitos (2008)
22. Strom, J., Richardson, A., Olson, E.: Graph-based segmentation for colored 3d laser point clouds. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2131–2136 (October 2010)

A Cognitive Vision System for Nuclear Fusion Device Monitoring

Vincent Martin[1], Victor Moncada[1], Jean-Marcel Travere[1], Thierry Loarer[1], François Brémond[2], Guillaume Charpiat[2], and Monique Thonnat[2]

[1] CEA, IRFM, Bldg. 507, F-13108 Saint Paul Lez Durance, France
[2] INRIA, PULSAR research team, 2004 routes des Lucioles, BP93, F-06902 Sophia Antipolis, France
vincent.martin@cea.fr

Abstract. We propose a cognitive vision-based system for the intelligent monitoring of tokamaks during plasma operation, based on multi-sensor data analysis and symbolic reasoning. The practical purpose is to detect and characterize in real time abnormal events such as hot spots measured through infrared images of the in-vessel components in order to take adequate decisions. Our system is made intelligent by the use of a priori knowledge of both contextual and perceptual information for ontology-driven event modeling and task-oriented event recognition. The system is made original by combining both physics-based and perceptual information during the recognition process. Real time reasoning is achieved thanks to task-level software optimizations. The framework is generic and can be easily adapted to different fusion device environments. This paper presents the developed system and its achievements on real data of the Tore Supra tokamak imaging system.

Keywords: cognitive vision system, infrared monitoring, ontology, multi-sensor event fusion, thermal event recognition, real-time vision, ITER.

1 Introduction

Tokamaks are complex devices operated to produce controlled thermonuclear fusion power by magnetic confinement of a plasma (fully ionized gas) in a torus. Even if the temperature drastically decreases from core plasma to edge plasma (from 10^8 to 10^4 °C), the equilibrium of the plasma discharge requires a direct contact with Plasma Facing Components (PFCs) exposed to very high heat fluxes in the range of 10-15 MW.m^{-2}. As a consequence, these PFCs must be continuously monitored to prevent irreversible damages. Infrared (IR) or visible imaging diagnostics (i.e. sensor data analysis system) are routinely used as plasma control systems during operation. The developed systems consist of detecting a high increase of the temperature (i.e. of the IR signal) beyond fixed levels for a set of predefined Regions of Interest (RoIs) [8]. In the perspective of ITER[1], with a network equipped with 36 cameras (one mega pixels at 500Hz each)

[1] International Thermonuclear Experimental Reactor, first plasma in 2020.

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 163–172, 2011.
© Springer-Verlag Berlin Heidelberg 2011

for the IR and visible imaging diagnostic, a complementary approach must be found to alleviate this intensive user-interaction demand. In addition, the future use of metallic components in ITER adds complications in surface temperature estimation of exposed PFCs because of multiple light reflections, making more intricate temperature threshold settings.

Therefore, there exists a real need in improving the performance of such imaging diagnostics. One challenge is to design an intelligent vision system for automatic recognition of thermal/plasma events (also called phenomena) and to embed it into the fusion device environments (see Figure 1 for typical thermal events on two infrared views at Tore Supra). First attempts towards phenomenon recognition for machine protection issues in visible and IR videos are presented in [9] and [7]. Murari et al. [9] propose a bottom-up approach for the automatic recognition of two specific events (plasma instability patterns and dust particles) observed with fast visible and infrared imaging systems during plasma experiments. Martin et al. [7] propose an approach combining prior knowledge of perceptual and contextual information for the automatic recognition of electrical arcing events in infrared videos.

In this paper, we more specifically address the problem of integration of techniques developed so far for global scene understanding with an emphasize on real time system requirements. We propose to use a more advanced image understanding framework based on knowledge driven reasoning. To ensure a high degree of self-adaptability to varying acquisition conditions, we also propose to analyze images without assumptions on absolute temperature measurements. Our framework is inspired from cognitive vision paradigms recently explored [3] in the field of video surveillance applications [12]. More precisely, we propose a system that is reusable for different fusion devices, that can handle unforeseen situations, that can adapt to its environment (i.e. plasma condition awareness), and that can reason from memorized or learned relevant information.

2 Proposed Approach

Our cognitive vision system must satisfy three major constraints specific to the fusion device environment. First, it must ensure the maximum security level: a detected but unclassified event remains potentially dangerous (e.g. an unforeseen overheating area), and thus should be handled by the machine protection system. The second constraint refers to the system versatility. Since all fusion devices provide their own diagnostics (i.e. different types of cameras) and several sets of possible phenomena (about 20 per machine), creating all the vision tasks for each environment becomes an intractable task and thus requires to use a shared formalism for knowledge description of both contextual and perceptual information. System performance in term of computational time represents the third major issue since each vision task must fulfill the real time constraint of the considered fusion device environment. Our vision system meets these requirements thanks to the following abilities:

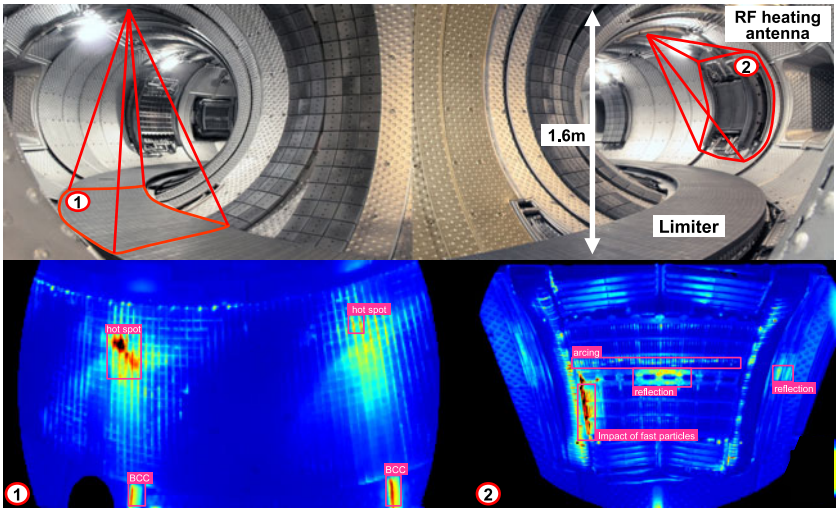


Fig. 1. Infrared monitoring of PFCs in the TS tokamak during a plasma discharge. Temperatures range from 120°C (blue) to 900°C (dark red). The bounding boxes represent manual annotations defined with the help of physicists. BCC stands for *Badly Cooled Component*.

Multi-sensor Data and Event Fusion. We merge information at different levels. In a first case, the goal is to find cross-correlation between event features (e.g. similarity between hot spot temperature evolutions) observed at several scene locations. In a second case, an event can generally be detected by several diagnostics (e.g. by IR and visible cameras, by spectroscopy). Combining video events with other sensor events (e.g. a curve peak) helps in achieving high robustness of the system as illustrated in [12].

Ontology-Driven Vision Task Composing. The unification of information and especially visual and contextual data can be achieved with ontologies, as demonstrated in [5] and [4]. Our ontology is used to link the different semantic representations related to (1) the diagnostics describing the perceived environment, (2) the scene observed by the camera network, (3) the plasma discharge scenario parameters and (4) the phenomena to be recognized by the system.

Real Time Vision. To avoid combinatorial explosion of the recognition process based on multi- event hypothesis solving, most of the reasoning is deterministic and performed during the ontology-driven vision task composing by means of spatio-temporal and logic constraint propagation. This approach makes possible task-level optimizations as task pooling and task parallelization mandatory for real time reasoning. Real time constraints imposed by the acquisition systems are also handled thanks to hardware acceleration of low-level algorithms as describe [6].

2.1 Event Modeling for Vision Task Composing

We have developed a software platform dedicated to vision task composing, tuning and launching called PInUP (Plasma Imaging data Understanding Platform). We propose a formal method to infer the vision task hierarchy from prior knowledge described in our ontology. One advantage of this method is to perform two task-level optimizations so as to decrease the CPU loading and thus to save computational time. The first level concerns task pooling used to avoid multiple launching of the same vision algorithm (e.g. detection algorithm). The second level concerns task parallelization which aims at minimizing the length of linear task sequences and is a prerequisite for CPU multi-threading based computing.

The phenomenon ontology branch (see figure 2) contains the class hierarchy of interesting events for machine protection purposes. Each event categorization process is associated with at least one specialized vision task composed of four pipelined algorithms. The specialization of each algorithm is deduced from the attributes of the phenomenon ontology branch. The typical event recognition process consists of:

- A trigger algorithm based on the Plasma scenario attributes such as plasma states. These attributes describe the necessary plasma conditions for the event to occur. This algorithm triggers the launching of the event categorization process.
- A low level detection algorithm deduced from Diagnostic attributes attached to the event. For instance, detection of transient hot spots is performed by a motion detection algorithm.
- A spatial filtering of detected pixels based on the Scene attributes specifying the possible locations of the event. We use an interactive 2D scene model where the user has the possibility to fill symbolic and numerical attributes attached to each defined zone.
- A set of specialized feature descriptors based on the Visual concept attributes (e.g. shape, size, time duration, etc.) selected from prior knowledge of the phenomenon characterization. The corresponding range values are learned from representative training samples.

This ontology-driven construction process leads to the work flow of vision tasks presented in figure 3. Currently, a graphical tool let end-users (e.g. diagnosticians) compose the vision task hierarchy in a user-friendly way by means of component block manipulation.

3 Application to Thermal Event Recognition at Tore Supra

The main purpose of thermal event recognition is to handle unforeseen situations as unexpected hot spots. People in charge of the experiments can then focus either on filtered events (e.g. for plasma control issues) or on hot spots remaining unclassified to check their dangerousness.

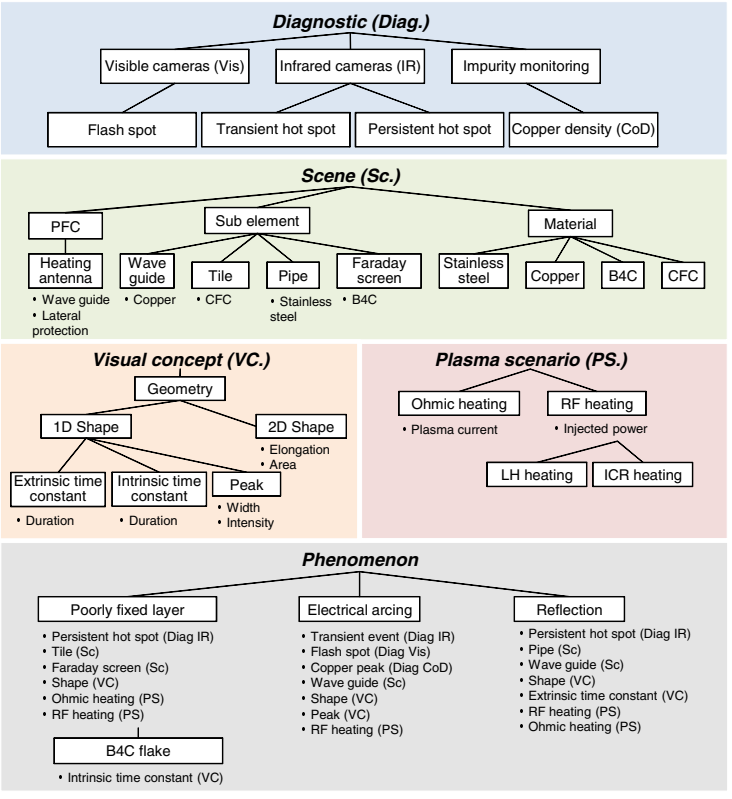


Fig. 2. Part of the ontology developed for thermal event recognition. The five main branches are represented with corresponding class hierarchies and attributes.

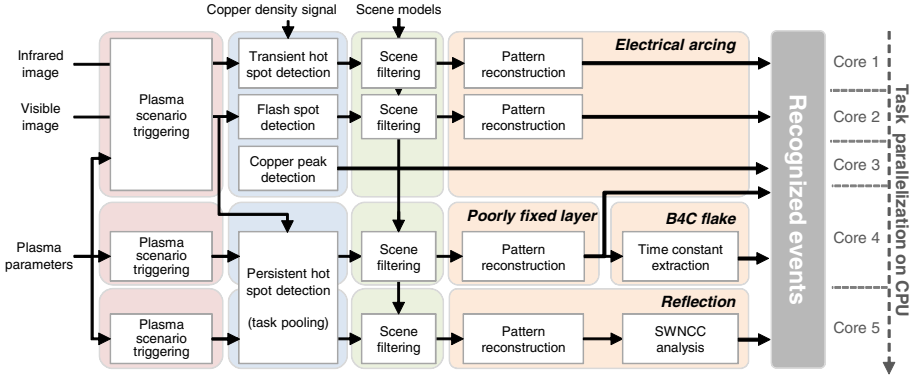


Fig. 3. Work flow of vision tasks for the four phenomena modeled in figure 2

3.1 Hot Spot Detection

Physically, a hot spot is a local area on a PFC where the temperature measured is above an accepted range of values, and is then considered as overheating. Since temperature calibration is still an open problem in tokamak environments, we propose to define the detection of hot spots as a spatial or temporal local image contrast analysis problem, therefore ensuring a high degree of self-adaptability to varying acquisition conditions.

Transient Hot Spot Detection. Some events can be characterized thanks to their temporal signature as electrical arcs which last only few dozen of milliseconds. A fast change detector based on pixel intensity is then the best appropriate solution to discriminate them against the other types of hot spots (see figure 4(b)). To this end, we adopt a pixel-wise background modeling and subtraction technique developed by Butler et al. [2].

Persistent Hot Spot Detection. All hot spots lasting more than few dozen of milliseconds are considered as persistent. Since time is not necessary a discriminant clue for these hot spots, we adopt a local adaptive thresholding technique for their detection. After extensive tests of state-of-the-art algorithms [10], we found that the efficient implementation of the Sauvola's method based on integral images [11] gives the best results on our data (see figure 4(c)). Using some improvements for computational purposes, the thresholded image $q(x, y)$ from an input image I with pixel intensities $p(x, y)$ is such that:

$$\forall (x,y) \in I, \quad q(x,y) = \begin{cases} 0 & \text{if} \quad p(x,y) < \tau(x,y) \quad \text{or} \quad p(x,y) \leq R \\ 255 & \text{otherwise} \end{cases} \tag{1}$$

where

$$\tau(x,y) = \mu(x,y) + k \left(\mu(x,y) - \max_I(p)\right) \left(\frac{2\sigma(x,y)}{\max_I(p) - \min_I(p)} - 1\right) \tag{2}$$

with $\mu(\cdot)$ and $\sigma(\cdot)$ are the average and standard deviation values of pixel intensities in the spatial neighbourhood centered on (x, y), and $R = \arg\max\{hist(I)\}$.

We have also extended the algorithm to adaptive multi-threshold image segmentation through a n-pass procedure in order to better discriminate very hot spots inside hot regions as seen in figure 4(c).

3.2 Hot Spot Categorization

The next steps after detecting hot spots consist in extracting semantic information based on their appearance and behaviour. In the scope of machine protection issues, this step is of primary importance to assess hot spot dangerousness.

(a) (b) (c)

Fig. 4. Hot spot detection results on an infrared image of TS heating antenna (a). Transient hot spots (b) and persistent hot spots (c) are clearly discriminated thanks to the two dedicated algorithms.

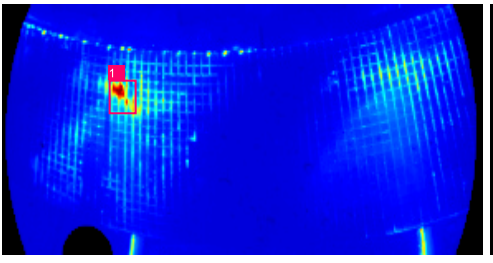
Persistent Hot Spot Categorization. Among all the modeled thermal events, one have to pay attention to the reflection event. Indeed, reflections patterns correspond to false hot spots which are not *per se* dangerous. They can be observed on reflective materials e.g. copper, stainless steel but generally not carbon considered closed to a black body behaviour. Reflections appear when PFCs in direct contact with the plasma experience surface temperature in the range of $1000 - 2000°C$. At Tore Supra, reflections arise from hot regions mainly located on the device floor called the limiter. A direct approach to match a reflection source and a reflection pattern is to measure the similarity between the temporal evolutions of their temperature. To this end, we compute the normalized cross-correlations (NCC) on a sliding time window (SWNCC) of width T between candidates of both sources f (hot spots on the limiter) and reflection patterns g (hot spots on metal-made PFCs) using the maximal temperature of each hot spot as input feature:

$$\text{SWNCC} = \frac{1}{T} \sum_{u=t-T}^{t} \frac{(f(u) - \overline{f})(g(u) - \overline{g})}{\sigma_f \sigma_g} \tag{3}$$

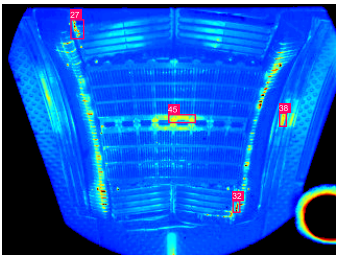
with f and g the maximal temperatures of the detected hot spots and $\overline{f}, \overline{g}, \sigma_f, \sigma_g$ their respective averages and standard deviations over the time period T.

Figure 5 presents the result of multi-camera event feature fusion performed on two synchronized infrared cameras. The goal is to characterize reflection patterns on the monitored heating antenna while applying radio-frequency (RF) plasma heating. As seen in figure 5, the temperature evolution of three patterns (no. 32, 38 and 45) are well-correlated (SWNCC>0.9) with the hot spot pattern detected on the limiter (no. 1) and then can be classified as reflection patterns. On the contrary, the pattern no. 27 has clearly an independent thermal behaviour, and will remain classified as *persistent hot spot* for further analysis and event classification refinement.

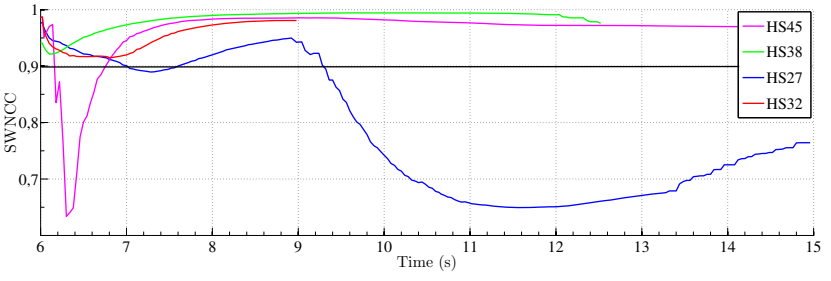
Transient Hot Spot Categorization. We present in figure 6 the result of arc recognition with infrared and visible imaging diagnostics during a same plasma discharge.



(a) Hot spot detected on the limiter and used as heat source for the computation of SWNCC.



(b) Candidates of reflection patterns used for the computation of SWNCC.



(c) Results of SWNCC ($T = 10s$) between the temperature evolutions of the hot spot in (a) and hot spots in (b) during a plasma discharge. Hot spots detected on metallic components of the heating antenna with a SWNCC>0.9 are considered as reflection patterns.

Fig. 5. Example of multi-camera event feature fusion for the recognition of reflection patterns based on thermal behaviour cross-correlation

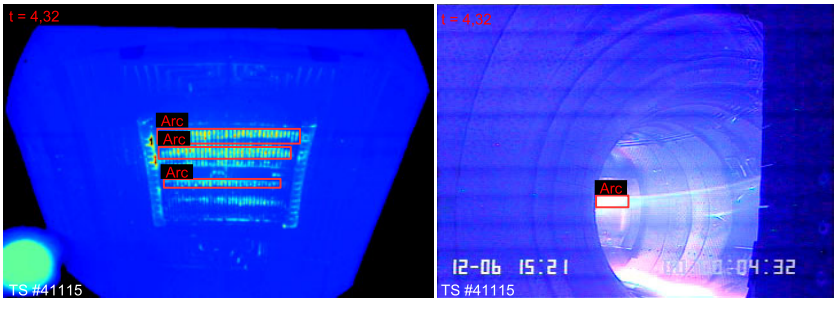


Fig. 6. Synchronized IR (left) and visible (right) camera views aiming the same PFC (heating antenna) from different lines of sight inside the TS tokamak. The arc event is successfully recognized by the system in the two cases.

Although both sensors data and observed scenes are completely different, the system achieves the recognition of the same arc event appearing in front of the same heating antenna. This ability of correlating events recognized with several diagnostics is considered as an essential element for plasma control system reliability in case of unexpected failure of one diagnostic during plasma operation.

4 Validation of Experimental Results

Our vision system is currently implemented on the TS tokamak in parallel of the existing RoI-based monitoring system. The infrared viewing system is composed of 7 infrared video cameras (320×240 pixels at 50Hz) monitoring one part of the toroidal limiter and the five RF heating antennas as seen on top of figure 1 (resp. ① and ②). Our system has been tested and validated with plasma physicists during the last experimental campaign on about 50 plasma discharges lasting between 15 and 90 seconds. Performance of arc recognition based on the presented framework has been evaluated in [7] thanks to existing ground truth data (visual annotations). Results shows a precision of 98% and a sensitivity of 92%. The validation of the SWNCC method used for the characterization of reflection patterns is based on a qualitative comparison with simulated infrared images obtained with a realistic photonic simulation code described in [1]. Results shows that recognized reflection patterns are in accordance in terms of location and size with those found in simulated images. Further quantitative evaluation using this new simulation tool are foreseen to assess the effectiveness of the SWNCC method for reflection pattern recognition in different contexts (i.e. various plasma conditions and monitored PFCs).

5 Conclusion

In this paper, we demonstrate that a cognitive vision system based on qualitative imaging analysis can achieve a physical interpretation of observed phenomena during plasma operation in tokamaks. From a computer vision viewpoint, this real system is made original by the merging of multiple sources of information (multi-camera and multi-sensor data fusion) at different levels (pixels and event features) during the recognition task, its combination of different software optimization schemes for real time computations, and the use of ontology-driven inference mechanisms for system re-usability on different tokamak environments. The developed software platform is now daily used at Tore Supra during plasma operation as a computer-aided decision support system and is going to be installed on JET[2], which is currently the world's biggest tokamak. On-going work concerns the system performance evaluation to prepare its integration within the plasma control system of Tore Supra.

Finally, this pioneer work for the domain is also an opportunity for both computer science and plasma science communities to progress together for preparing both the safety and physics exploitation of ITER.

[2] Joint European Torus.

Acknowledgment. This work was supported in part by the European Communities under the contract of Association between EURATOM, CEA and the French Research Federation for fusion studies. The views and opinions expressed herein do not necessarily reflect those of the European Commission. This work was also supported by the French Stimulus Plan 2009-2010.

References

1. Aumeunier, M.H., Travere, J.M., Loarer, T., Benoit, F.: Simulation vs. reality: a step towards a better understanding of infrared images in fusion devices. IEEE Transactions on Plasma Science (to appear, August 2011)
2. Butler, D., Bove Jr., V.M., Sridharan, S.: Real-time adaptive foreground/background segmentation. EURASIP J. Appl. Signal Process. 2005, 2292–2304 (2005)
3. Eggert, J., Wersing, H.: Approaches and Challenges for Cognitive Vision Systems. In: Sendhoff, B., Körner, E., Sporns, O., Ritter, H., Doya, K. (eds.) Creating Brain-Like Intelligence. LNCS, vol. 5436, pp. 215–247. Springer, Heidelberg (2009)
4. Gomez-Romero, J., Patricio, M., Garcia, J., Molina, J.: Ontological representation of context knowledge for visual data fusion. In: Int. Conf. on Information Fusion, pp. 2136–2143 (2009)
5. Maillot, N., Thonnat, M., Boucher, A.: Towards ontology-based cognitive vision. Machine Vision and Applications 16, 33–40 (2004)
6. Martin, V., Dunand, G., Moncada, V., Jouve, M., Travere, J.M.: New FPGA-based image-oriented acquisition and real-time processing applied to plasma facing component thermal monitoring. Rev. of Sci. Instr. 81(10), 10E113 –10E113–4 (2010)
7. Martin, V., Travere, J., Brémond, F., Moncada, V., Dunand, G.: Thermal event recognition applied to protection of tokamak plasma-facing components. IEEE Trans. on Instr. and Meas. 59(5), 1182–1191 (2010)
8. Moreau, P., Barana, O., Brémond, S., Colas, L., Ekedahl, A., Saint-Laurent, F., Balorin, C., Caulier, G., Desgranges, C., Guilhem, D., Jouve, M., Kazarian, F., Lombard, G., Millon, L., Mitteau, R., Mollard, P., Roche, H., Travere, J.M.: RF heating optimization on Tore Supra using feedback control of infrared measurements. Fusion Engineering and Design 82(5-14), 1030–1035 (2007)
9. Murari, A., Camplani, M., Cannas, B., Mazon, D., Delaunay, F., Usai, P., Delmond, J.: Algorithms for the automatic identification of MARFEs and UFOs in JET database of visible camera videos. IEEE Trans. on Plasma Science 38(12), 3409–3418 (2010)
10. Sezgin, M., Sankur, B.: Survey over image thresholding techniques and quantitative performance evaluation. Journal of Electronic Imaging 13(1), 146–168 (2004)
11. Shafait, F., Keysers, D., Breuel, T.M.: Efficient implementation of local adaptive thresholding techniques using integral images. In: SPIE Conference Series, vol. 6815 (January 2008)
12. Zouba, N., Brémond, F., Thonnat, M.: Multisensor fusion for monitoring elderly activities at home. In: Proceedings of AVSS 2009, pp. 98–103. IEEE Computer Society, Washington, DC, USA (2009)

Knowledge Representation and Inference for Grasp Affordances

Karthik Mahesh Varadarajan and Markus Vincze

Automation and Control Institute, TU Vienna, Austria
{kv,mv}@acin.tuwien.ac.at

Abstract. Knowledge bases for semantic scene understanding and processing form indispensable components of holistic intelligent computer vision and robotic systems. Specifically, task based grasping requires the use of perception modules that are tied with knowledge representation systems in order to provide optimal solutions. However, most state-of-the-art systems for robotic grasping, such as the K- CoPMan, which uses semantic information in mapping and planning for grasping, depend on explicit 3D model representations, restricting scalability. Moreover, these systems lacks conceptual knowledge that can aid the perception module in identifying the best objects in the field of view for task based manipulation through implicit cognitive processing. This restricts the scalability, extensibility, usability and versatility of the system. In this paper, we utilize the concept of functional and geometric part affordances to build a holistic knowledge representation and inference framework in order to aid task based grasping. The performance of the system is evaluated based on complex scenes and indirect queries.

Keywords: Ontologies, Knowledge Representation, Grasp Affordances, ConceptNet.

1 Introduction

Use of knowledge bases for holistic scene understanding and processing has been a growing trend in computer vision and robotics. In these areas, as well as other systems requiring the use of ontologies, Semantic Web based knowledge acquisition systems have been used extensively. These are typically defined using Web Ontology Languages (OWL), that are characterized by formal semantics and RDF/XML-based serializations. Extensions to OWL have been used in semantic editors such as Protégé and semantic reasoners and ontology bases such as Pellet, RacerPro, FaCT++, HermiT, etc. In the area of semantic text parsing and knowledge management, a number of frameworks such as Framenet, Lexical Markup Framework (LMF), UNL, WordNet and WebKB are available. Alternatively, a number of tools for conceptual knowledge management have also been developed recently. These include reasoners and concept ontologies such as Mindpixel, Cyc, Learner, Freebase, YAGO, DBpedia, and MIT ConceptNet. These semantic reasoners and ontology databases can be directly exploited for applications in robotic manipulation.

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 173–182, 2011.
© Springer-Verlag Berlin Heidelberg 2011

The most significant of semantic knowledge acquisition systems for robotic vision systems is KnowRob (Knowledge Processing for Robots) [1], which uses reasoners and machine learning tools such as Prolog, Mallet and Weka, operating on ontology databases such as researchCyc and OMICS (indoor common-sense knowledge database). In the case of KnowRob, the data for the knowledge processing stems from three main sources: semantic environment maps, robot self-observation routines and a full-body human pose tracking system. Extensions to KnowRob, such as the K-COPMAN (Knowledge-enabled Cognitive Perception for Manipulation) system [2], enable autonomous robots to grasp and manipulate objects.

All the above frameworks for knowledge acquisition based object grasping and manipulation suffer from the fact that they require the use of explicit model databases containing object instances of the query to be processed, in order to obtain successful object recognition. K-COPMAN, for instance, uses CAD for matching 3D point clouds in order to identify the queried object in the given environment. Furthermore, while using semantic knowledge of the scene in order to improve object recognition and manipulation, these systems are largely devoid of performing implicit goal-directed cognitive tasks such as substituting a cup for a mug, bottle, jug, pitcher, pilsner, beaker, chalice, goblet or any other unlabeled object, but with a physical part affording the ability to hold liquid and a part affording grasping, given the goal of 'bringing an *empty cup*' and no cups are available in the work environment.

In order to alleviate these issues, we utilize the concept of part affordances. Gibson proposed the original idea of affordances grounded in the paradigm of direct perception. Physical affordances define the agent's interaction possibilities in terms of its physical form [3]. For example, stable and horizontal surfaces are needed to support objects, objects need to have a brim or orifice of an appropriate size, in order to be functional as a container to drink from. Additional examples of affordances studied with respect to robotic manipulation in [3] include 'sittability' affordance of a chair that depends on body-scaled ratios, doorways affording going through if the agent fits through the opening, and monitors afford viewing depending on lighting conditions, surface properties, and the agent's viewpoint. The spectrum of affordances have been extended to include social-institutional affordances, defining affordances based on conventions and legally allowed possibilities leading to mental affordances. Affordances based on History, Intentional perspective, Physical environment, and Event sequences (HIPE) leading to functional knowledge from mental simulations have been studied in [4]. Affordances serve as key to building a generic, scalable and cognitive architecture for visual perception. 'Affordance based object recognition' or recognition based on affordance features is an important step in this regard.

2 Overview

Most prior work in robotic grasping makes use of 3D model instance representations. The primary contribution of this paper is in providing a unified and holistic knowledge assimilation and deployment framework that is intended for robotic grasping. Since the framework is devoid of 3D model instance representations, it lends itself to extensibility to compound shape detection and grasping through the Recognition by Components (RBC) theory.

2.1 Ontology of Concepts

The fundamental basis of our framework revolves around the theme of 'Conceptual Equivalence Classes'. These classes are defined as sets of objects that are interchangeable from the view-point of usage for the primary functionality of the object. Hence, objects such as mugs, cups and beakers form an equivalence class. Bags and baskets also form an equivalence class, so do cans and bottles, bikes and motorbikes and so forth. Equivalence classes can be uniquely defined and recognized in terms of their (a) Part Functional Affordance Schema and (b) Part Grasp Affordance Schema. It should be noted here that the definition of conceptual equivalency class used here is distinct and unrelated to the equivalency class definitions provided by the OWL framework, which uses only textual or named entity equivalency.

2.2 Knowledge Ontology Based on Textual Semantics

In our framework, we employ WordNet [5] for generating textual unit definitions for concepts or objects queried for. While WebKB provides improvements over WordNet, while returning results that are restricted to nouns (of specific interest to our framework), the standalone nature of WordNet recommends its usage. WordNet provides a lexical database in English with grouped sets of cognitive synonyms (synsets), each expressing a distinct concept. It also records the various semantic relations between these synonym sets, such as hypernyms (higher level classes), hyponyms (sub-classes), coordinate terms (terms with shared hypernyms), holonyms (encompassing structure) and meronyms (constituent parts). The system interacts with the WordNet interface based on the queried term to obtain a possible match. The system also assimilates concept 3D geometric shape information such as Sphere, Cylinder, Cube, Cone, Ellipsoid, Prism, etc., 2D geometric shape information such as Square, Triangle, Hexagon, Pentagon, Ellipse etc. and abstract structural concepts such as Thin, Thick, Flat, Sharp, Convex, Concave etc. by parsing the concept definition. Additionally, information on material properties of the concept such as Metal, Wood, Stone, Ceramic etc. and part functional affordance properties (based on terms such as Cut, Contain, Store, Hold, Support, Wrap, Roll, Move, Ride, Enter, Exit, Gap, Hole) are also obtained and stored by the system.

2.3 Knowledge Ontology Based on Visual Features

While visual unit definitions can be used to improve the performance of the system or to obtain instance level recognition, our novel framework for conceptual equivalence class recognition and grasping system does not require the use of these databases and hence is 3D/2D model free. Furthermore, it should be noted that from the viewpoint of grasping using range images, monocular image information is largely superfluous. Instance level recognition, if necessary in future revisions to the system, can be carried out using a bag of features approach working with SIFT/SURF or other state-of-art feature descriptors on labeled image or 3D shape databases (such as LabelMe, LabelMe 3D and ImageNet).

2.4 Knowledge Ontology Based on Conceptual Properties

For the case of conceptual unit definitions, we employ the Open Mind Common Sense (OMCS) [6] based ConceptNet framework. ConceptNet has been used in the context of robotic task management [7]. The particular choice of this ontology database is due to its exhaustiveness, ease of use and suitability of attributes with respect to our affordance framework. The ontology provides English language based conceptual groupings. The database links each concept with properties such as 'InstanceOf' and 'SymbolOf' – possibile semantic replacements, 'ConceptuallyRelatedTo' – possible functional/ conceptual replacements, 'PartOf' – encompassing structures, 'Receives Action', 'CapableOf', 'UsedFor' – possible functional affordances as well as 'MadeOf', 'HasProperty' etc. that provide further information about the concept. The use of these properties enables the part affordance based equivalence class selection.

2.5 Knowledge Ontology Based on Grasp Affordances

For the case of part grasp affordance definitions, a number of systems are available. These can be used for limiting the large number of possible hand configurations using grasp preshapes. Humans typically simplify the task of grasping by selecting one of only a few different prehensile postures based on object geometry. One of the earliest grasp taxonomy is due to Cutkosky [8]. In our system we employ the 'Human Grasping Database' [9] from KTH-Otto Bock. This taxonomy lists 33 different grasp types hierarchically assimilated in 17 grasp super-types. It is possible to most of these grasp types to geometric shapes they are capable of handling. Each query concept is defined (as a whole or in parts) to provide grasp affordances of the types listed in the taxonomy database.

2.6 Knowledge Ontology Based on Part Functional Affordances

The most important component of the presented system is the Part Functional Affordance Schema. This component essentially performs the symbol binding – mapping concepts: in our case – the *Conceptual Equivalence Classes* to visual data in the form of 3D geometries. While various schemes for affordance definitions have been studied in the past, we utilize a set of part functional affordance schema, largely with respect to objects found in households and work environments. These affordances are based on functional form fit of the Conceptual Equivalence Classes. A representative section of the part functional affordance schema is presented in Table 1. Note that the functional affordance here is defined with respect to objects of the class being able to perform the defined function.

The scale of each part is also defined with respect to a discrete terminology set based on comparative sizes – (finger (f), hand (h), bi-hand (b), arm/knee (a), torso (t), sitting posture (i), standing posture (d), non-graspable (n) etc.). The conceptual equivalence classes are defined based on joint affordances of parts of the objects, along with their topological relationships.

Based on these attribute definitions, the equivalence classes can be uniquely represented. Examples of equivalence classes are provided in Table 2. Note that (ga) denotes grasp affordance and (pa) denotes part affordance.

Table 1. Representative Part Functional Affordance Schema

Part Functional Affordance	Geometric Mapping	Examples
Contain - ability	High convexity	Empty bowl, Cup
Support - ability	Flat - Convex	Plate, Table
Intrinsic contain - ability	Cylinder/Cube/Cuboid/Prism	Canister, Box
Incision - ability	Sharp edge (flat linear surface)	Knife, Screwdriver
Engrave - ability	Sharp Tip	Cone, Pen
2D Roll - ability	Circular/ Cylindrical	Tire, Paper Roll
3D Roll - ability	Spherical	Ball
Weed - ability[a]	Linear textural structures	Comb, Brush
Filter - ability[a]	Bi-linear textural structures	Grid, Filters
Wrap(p) -ability	w.r.t. given shape	Shoe, Glove
Connect - ability[a]	Solid with support (m)	Plug, USB Stick

Table 2. Example Equivalence Class defintions

Equivalence Class	Definition
Basket	1v2, b-a, handle (ga), opening (pa: containability)
Plate	h-b, (ga), (pa: supportability)
Cup	1h2, f-h, handle (ga), opening (pa: containability)
Chair	1os2, a-i, 2x(pa: supportability)
Canister	h-b, (pa: intrinsic containability)
Box	h-i, (pa: intrinsic containability)
Plug	1v2n, f-h, support, contact (pa: connectability (m))
Knife	1h2, f-h, grip, blade (pa: incisionability)
Bike	b,a,a, 1v2(3hv4), seat (pa: supportability), 2xwheels (pa: 2drollability)
Laptop	b-a, (pa: supportability)
Pen	f-h, grip, tip (pa: engravability)
Ball	h-a, (pa: 3drollability)
Spoon	1h2, f-h, grip, opening (pa: containability)
Spatula	1h2, f-h, grip, opening (pa: supportability)
Faucet	1h2, f-h, pipe, orifice (pa: filterability)
Suitcase	1v2, b-a, handle, box (pa: intrinsic containability)
Desk	a-d (pa: supportability)
Cabinet	a-d (pa: intrinsic containability)
Stair	nx(pa: supportability)
Shoe	opening (pa: containability), (pa: wrappability/ ellipsoid)
Key	1v2n, f-h, support, contact (pa: connectability (m))
Brush	grip, bristles (pa: weedability)
Shelf	nx(pa: supportability)
Scissors	2xblade (pa: incisionability)
Cars	4xwheels (pa: 2drollability) (intrinsic containability)

3 Query Evaluation

For any given query term, the system checks for availability of concept definition in the following list of attributes in a sequential order. The first database to be queried for is (a) the Part Affordance Schema. If unavailable, the system checks for the availability of a concept in the Part Affordance Schema that is matched using (b) the synsets of the queried term, followed by the 'InstanceOf' and 'SymbolOf' properties from ConceptNet, if necessary. If a match is not found, the system tries to use (c) the ConceptuallyRelatedTo property returned by ConceptNet (in response to the query term) to define possible alternatives for the object to be found. Alternatively, (d) the coordinate terms of queried object are searched for in order to obtain a conceptual replacement object. If a match is still not found, the system searches in (e) the holonym list and (f) the 'PartOf' list from ConceptNet. This is followed by matching for (g) 'ReceivesAction', 'CapableOf', 'UsedFor', which denote possible functional equivalency of the objects.

The frequency scores on each of these properties are also returned as a measure of confidence in the object found. If no matches are found in the Part Affordance Schema for the queried object or any of the alternatives to be searched for, as suggested by the above list of related objects, the system parses the definitions of the queried object returned by both WordNet and ConceptNet to search for structural properties associated with the object. These include shape geometry information such as cylindrical, spherical or cuboidal or its alternate surface forms as well as abstract geometrical property terminologies such as flat, thick, thin, concave or convex.

Material properties of the object from the parsed definitions such as wood, stone or metal, (as well as those returned by the 'MadeOf' property from ConceptNet) as well as functional affordances from WordNet are stored as properties of the concept being queried for. While it is possible that the given range scene can be searched for the required object entirely based on the geometry information or the defined geometries (from the Part Functional Affordance Schema) based on a matched affordance property returned from parsing the concept definitions, the confidence level (based on frequency scores and weighted by property confidence measures) returned by such an unit recognition scheme is very low. Furthermore, based on a learned appearance database of different material types (such as wood, stone or metal), the classification can be improved if monocular scene imagery is also available. Such a material classification approach can also be used to select salient regions in the scene in order to reduce computation requirements of the range image processing.

3.1 Detection of Part Affordances

As discussion earlier, the Part Functional Affordance Schema defines unique symbol binding from affordance concepts to observables in terms of functional geometry mapping. While certain affordances are defined based on geometrical shape structures such as cylinders, cubes, cuboids and spheres or continuous space parametric variations of these shapes (as defined by superquadrics), other affordances are defined in terms of abstract geometrical attributes such as flat, concave, convex, sharp tip, sharp edge, linear textural structures, bi-linear textural structures. Joint affordances are defined in terms of more than one part. While detection results of the first set

(geometrical shape structures) is directly available from the superquadrics, results for the second set (abstract geometries) can be inferred from the superquadrics. Since superquadrics model objects or parts as convex structures, presence of a concavity (such as the open cylindrical portion of a cup) can also be verified using visibility tests for cloud points and normals (for e.g. belonging to the inner surface of the cup, in comparison with a solid cylinder). Other attributes such as flatness and sharpness, linear and bi-linear textures can also be roughly estimated based on measures of size, shape and thickness of the quadric.

3.2 Detection of Grasp Affordances

Most of the grasp affordances based on the Otto Bock Grasping Database, can be uniquely represented in terms of geometrical shapes. For e.g., the small diameter affordance can be structurally defined as a superquadric with a high linear dimension value along one axis and small diameters along the others. This also holds true of prismatic affordance, though the diameter is much smaller. Power disk is suited for disk type structures of the size of the palm, parallel extension for cuboidal structures and distal for objects with disjoint ring shaped parts.

3.3 Query Matching

In the given scene of interest, the queried object for the given task is found using attributed graph matching of the concept node built for the query with all geometrical objects found in the scene. Among the several attributed graph matching approaches [10, 11] available, we use a low complexity approach based on Heterogeneous Euclidean Overlap Metric (HEOM) using the Hungarian Algorithm [11] for the matching process. Each object in the scene is represented as a graph with its parts defining nodes along with vector attributes that may be symbolic (such as affordances) or metric (scales). Given the limited number of objects in a given scene, the matching process is fast and accurate. In the case that more than one object is found in the scene, the nearest object is selected for manipulation.

4 Results and Evaluation

The performance of the concept evaluation algorithms for a given scene is demonstrated using a set of queries.

For the first scene (Fig. 1), a search query for ‘jug’ is presented. It should be noted that the query ‘jug’ is not available in our equivalence class database, hence causing the search to be non-trivial. Using WordNet based parsing, renders the part affordance of ‘containability’ with a weight measure of 2 (out of 10), based on frequency scores for primary (from definition text) and secondary characteristics (from other attributes). ConceptNet also renders the ‘containability’ affordance along with a ‘HasA’ attribute of ‘handle’ which provides the grasp affordance for the given case. The attributed graph for the given query is simple and is composed of nodes for ‘containability’ part affordance and a ‘handle’ – small diameter grasp affordance with an overall weighted confidence score of 1.66/4 (using concept and textual unit definitions of 1 and 3 respectively). The range image processing algorithms yield both

the mugs in scene as results (prioritized by the closest object), since these objects contain concavities (affordance: containability) and handles (grasp affordance) that match the query graph attributes exactly (normalized HEOM score of 1).

For the second scene (Fig. 2), a search query - 'bag' is presented. Again, since no equivalence class has been defined for the term 'bag', the computation of the search is non-trivial. For the given case, WordNet and ConceptNet render the 'containability' affordances along with the 'handle' grasp affordance. In addition, ConceptNet renders the scale parameter to be 'large' and equivalent to that of a 'box'. The confidence score on the resulting affordance description is 3.64/4 (since WordNet returns a high frequency score of 8). Since the queried scene contains 2 true 'bags', the range processing algorithms return both the bags as query results. Again the normalized HEOM score is 1, indicating a perfect match for known attributes. It can also be seen that the confidence in the result is high for the second scene, as compared to the first, since the rate of occurrence of the object in typical scenes (reflected in the frequency score from WordNet) is higher.

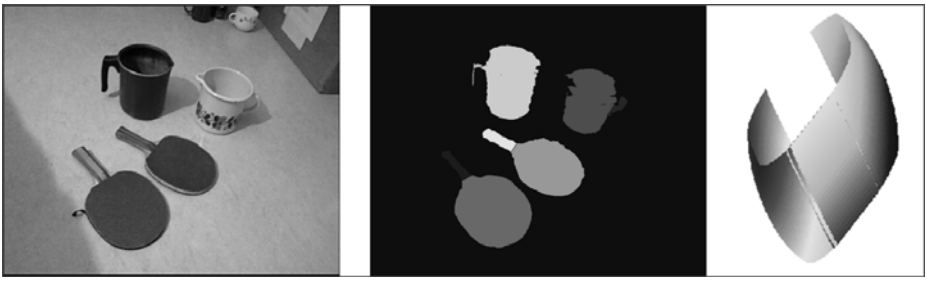


Fig. 1. Left to Right: (a) Input Scene, (b) Detected objects and their corresponding parts in the point cloud (c) Fitting of a cylinder corresponding to the 'jug' in the scene

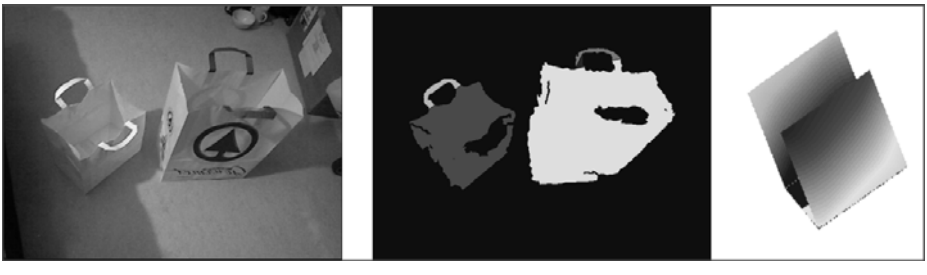


Fig. 2. Left to Right: (a) Input Scene, (b) Detected objects and their corresponding parts in the point cloud (c) Fitting of a cuboid corresponding to the 'bag' in the scene

5 Conclusion and Future Work

In this paper, we have presented a scalable knowledge assimilation and deployment framework for robotic grasping that is free of 3D model instance representations. We have used the paradigm of '*Conceptual Equivalence Classes*' and uniquely defined

them in terms of the minimalistic features of Part Functional Affordances and Part Grasp Affordances, leading to implicit cognitive processing for successful goal attainment. We have also provided a practical pathway for symbol binding – from concepts to observables by defining functional geometry mappings. The system is also capable of knowledge of affordance and interaction modes for unknown/ un-modeled objects based on partial information obtained from the constituent parts.

Currently, the number of part functional affordances supported by the system is quite limited. We plan to extend the number and range of the supported functional affordances in the future. This would also necessitate more advanced algorithms for the attributed graph matching. Furthermore, the current system is geared towards robotic grasping and manipulation while being capable of functional class level object recognition. As such, it uses only range information for the processing, without the need for 2D/3D databases. Extension of the scheme to perform instance level object recognition will necessitate the use of these databases. Moreover, while current system has been evaluated on a stand-alone system, actual deployment of the system on a robot with an arm and gripper for grasping is ongoing research. Finally, while the current system is intended to serve as a core component for goal-directed object recognition and manipulation, it can be used in a more holistic system for semantic visual perception such as the K-COPMAN.

References

[1] Tenorth, M., Beetz, M.: KnowRob — Knowledge Processing for Autonomous Personal Robots. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS (2009)

[2] Pangercic, D., Tenorth, M., Jain, D., Beetz, M.: Combining Perception and Knowledge Processing for Everyday Manipulation. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (2010)

[3] Raubal, M., Moratz, R.: A Functional Model for Affordance-Based Agents. In: Rome, E., Hertzberg, J., Dorffner, G. (eds.) Towards Affordance-Based Robot Control. LNCS (LNAI), vol. 4760, pp. 91–105. Springer, Heidelberg (2008)

[4] Barsalou, L., Sloman, S., Chaigneau, S.: The HIPE Theory of Function. In: Carlson, L., van der Zee, E. (eds.) Representing Functional Features for Language and Space: Insights from Perception, Categorization and Development, pp. 131–147. Oxford University Press, New York (2005)

[5] Fellbaum, C. (ed.): WordNet: An Electronic Lexical Database. MIT Press, Cambridge (1998)

[6] Havasi, C., Speer, R., Alonso, J.: Conceptnet 3: A Flexible, Multilingual Semantic Network For Common Sense Knowledge. Recent Advances in Natural Language Processing, 27–29 (September 2007)

[7] Xiong, X., Hu, Y., Zhang, J.: EpistemeBase: a Semantic Memory System for Task Planning under Uncertainties. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS (2010)

[8] Cutkosky, M.R.: On grasp choice, grasp models, and the design of hands for manufacturing tasks. IEEE Transactions on Robotics and Automation 5(3), 269–279 (1989)

[9] Feix, T., Pawlik, R., Schmiedmayer, H., Romero, J., Kragic, D.: A comprehensive grasp taxonomy. In: Robotics, Science and Systems Conference: Workshop on Understanding the Human Hand for Advancing Robotic Manipulation, Poster Presentation (June 2009)
[10] Hofman, I., Jarvis, R.: Object Recognition Via Attributed Graph Matching. In: Australasian Conference on Robotics and Automation, ACRA (2000)
[11] Jouili, S., Mili, I., Tabbone, S.: Attributed graph matching using local descriptions. In: Blanc-Talon, J., Philips, W., Popescu, D., Scheunders, P. (eds.) ACIVS 2009. LNCS, vol. 5807, pp. 89–99. Springer, Heidelberg (2009)

Towards a General Abstraction through Sequences of Conceptual Operations

Gregor Miller, Steve Oldridge, and Sidney Fels

Human Communication Technologies Laboratory
University of British Columbia
Vancouver, Canada V6T1Z4
{gregor,steveo,ssfels}@ece.ubc.ca
http://hct.ece.ubc.ca

Abstract. Computer vision is a complex field which can be challenging for those outside the research community to apply in the real world. To address this we present a novel formulation for the abstraction of computer vision problems above algorithms, as part of our OpenVL framework. We have created a set of fundamental operations which form a basis from which we can build up descriptions of computer vision methods. We use these operations to conceptually define the problem, which we can then map into algorithm space to choose an appropriate method to solve the problem. We provide details on three of our operations, `Match`, `Detect` and `Solve`, and subsequently demonstrate the flexibility of description these three offer us. We describe various vision problems such as image registration and tracking through the sequencing of our operations and discuss how these may be extended to cover a larger range of tasks, which in turn may be used analogously to a graphics shader language.

Keywords: OpenVL, Computer Vision, Abstraction, Language, Vision Shader.

1 Introduction

One of the main problems preventing widespread adoption of computer vision is the lack of a formulation that separates the need for knowledge of a concept from knowledge of algorithms. In this paper we present a first step towards an abstraction layer over computer vision problems. We introduce a new abstraction which allows us to describe common vision sub-tasks which we build upon to represent more sophisticated problems.

We work on the basis that algorithms and their parameters are low-level details with which a general user should not be concerned. The development of our abstraction layers is motivated by four reasons: 1) access is subsequently possible by those who are not experts in the field; 2) advances in the state-of-the-art can be incorporated into existing systems without re-implementation; 3) multiple back-end implementations become possible, allowing development of

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 183–192, 2011.
© Springer-Verlag Berlin Heidelberg 2011

hardware acceleration or distributed computing; and finally, 4) the abstractions provide a mechanism for general comparison of algorithms, thereby contributing to researchers in the field as well as general users.

This idea has been applied successfully in many other fields, notably the OSI reference model in networking [7] and OpenGL in graphics [20]. We have previously developed abstractions within sub-categories of computer vision, including access to and organisation of cameras [16], transport and distribution of vision tasks [2], and investigating the benefits of separating management and analysis of image data [1]. We also developed a conceptual framework for the effective development of computer vision analysis interfaces [17] and began the development of a vision shader language [18]; this paper presents advances on both of these contributions. Our main contribution is a novel abstraction layer for simpler access to sophisticated computer vision methods, presented in a general and extensible framework developed specifically for computer vision.

OpenCV [3], Matlab and similar frameworks are extremely useful but do not provide a user experience at the level we are proposing. These are libraries consisting of algorithms with complicated parameters. For example, the excellent OpenCV face detector [27] requires a large XML file (the result of extensive training on images of faces) as well as other image-based parameters, and implements a particular solution for a given set of training data (frontal faces only, etc.). Our abstraction is not intended as a replacement for existing libraries, but to complement them by providing a larger audience with access to the sophisticated methods.

Our abstraction is formulated through recognition of the common tasks within computer vision, abstracting these individually as *operations* (Section 3) and then providing a mechanism to define sequences which represent a more sophisticated task (Section 4). We provide detail on three of our operations (`Detect`, `Match`, `Solve`) and demonstrate the flexibility that can be achieved using such a small set.

2 Previous Work

Many attempts have been made to develop computer vision or image processing frameworks that support rapid development of vision applications. Image Understanding systems attempted to make use of developments in artificial intelligence to automate much of the vision pipeline [15,13,6]. The Image Understanding Environment project (IUE) [19] in particular attempted to provide high-level access to image understanding algorithms through a standard object-oriented interface in order to make them accessible and easier to reuse. More recently the OpenTL framework [22] has been developed to unify efforts on tracking in real-world scenarios. All of these approaches essentially categorise algorithms and provide access to them directly, which is at a lower-level than we are proposing in this paper.

Visual programming languages that allow the creation of vision applications by connecting components in a data flow structure were another important attempt to simplify vision development [14,25]. These contained components such

as colour conversion, feature extraction, spatial filtering, statistics and signal generation, among others. Declarative programming languages have also been used to provide vision functionality in small, usable units [26,23], although they are limited in scope due to the difficulty of combining logic systems with computer vision. While these methods provide a simpler method to access and apply methods, there is no abstraction above the algorithmic level, and so users of these frameworks must have a sophisticated knowledge of computer vision to apply them effectively.

There are many openly available computer vision libraries that provide common vision functionality [3,5,10,24,28]. These have been helpful in providing a base of knowledge from which many vision applications have been developed. These libraries often provide utilities such as camera capture or image conversion as well as suites of algorithms. All of these methods provide vision components and algorithms without a context of how and when they should be applied, and so often require expert vision knowledge.

One previous attempt at overcoming the usability problems associated with image understanding is discussed in the RADIUS project [8], which employed user-manipulated geometric models of the scene to help guide the choice of image processing algorithms. This operates at a higher-level than our proposed method, however it trades off power, breadth and flexibility to provide its abstraction. The abstraction we present in this paper is aimed to be extensible enough to provide accessible vision methods across the entire field.

3 Conceptual Operations

Many computer vision problems can be divided into smaller sub-problems and solved by providing solutions to each sub-problem. This applies conceptually as well as algorithmically and so we base our idea of *operations* on this principle. We allow the user to conceptually describe vision tasks by dividing the problem into conceptual sub-tasks, then the description is analysed and a suitable method selected. For example, image registration is typically solved by matching identical regions across images, and globally optimising for the alignment. There are many different methods for each stage of this problem, and some which combine them into a single step [21]. Under our approach, a user would describe the problem as a correspondence search, followed by a global optimisation. Our abstraction framework would then interpret this sequence and select the most appropriate method. This is the main contribution of our work, since the abstraction may select individual algorithms for match and solve, or one which does both, and hides the details from the user. Not only does this lead to simpler access to vision, but also opens the possibility of multiple implementations, by different universities and companies, in both software and hardware.

Our operations use various inputs and outputs, task parameters and constraints, all of which contribute to the problem description. For example, in a correspondence search (abstracted by our `Match` operation) we use constraints to define the search space, problem parameters to indicate the number of matches

Table 1. Example variable values and the problem conditions described using `Match`. N can easily be substituted for $|\mathcal{I}|$ to apply to all images rather than a subset. However, this will not enable search within the source image: this is only accessible via the explicitly defined cases in (a) and (i).

Example	Search Space	N_D	N_I	Problem Description
(a)	$Image$	1	0	Single match in source image
(b)	$Image$	1	1	Single match in single other image
(c)	$Image$	1	N	Single match in each of N images (excluding source)
(d)	$Image$	K	1	K matches in single other image
(e)	$Image$	K	N	K matches in each of N images
(f)	Set	1	$\|\mathcal{I}\|$	Single match from the set of images
(g)	Set	K	$\|\mathcal{I}\|$	K matches from the set of images
(h)	Set	K	N	K matches from subset of N images
(i)	Set	K	0	K matches in source image

(in a given number of images) and variances to indicate the differences across images. The operations are explained in the following section.

3.1 Operations

We have a small suite of operations we currently use to provide solutions for detection, tracking, correspondence, image registration, optical flow, matting and background subtraction. We do not attempt in this paper to provide a complete and finished formulation - this is a piece of on-going work, and our current set is intended to be a proof-of-concept which we will continue to expand upon. There is also a substantial quantity of subtle tweaks and defaults which could be made within an implementation; for this paper we are focussing on the abstraction, and will extend the work in future to define details of a framework implementing the abstraction.

Match: Our `Match` operator is used to extract a set of features $\mathcal{F}$ from a set of images $\mathcal{I}$ (containing $|\mathcal{I}|$ images) and find correspondences among $\mathcal{F}$. For a given feature $f \in \mathcal{F}$ multiple correspondences may be found within $\mathcal{I}$, or even within a single image $I \in \mathcal{I}$. The current problem defines which matches are important so we have developed a set of constraints and parameters to describe which features should be selected.

Problems which include correspondence can be described using three parameters: the number N_M of matches required; the number N_I of images to match across (where $N_I \leq |\mathcal{I}|$); and whether to return N_M matches per image (in which case $N_M N_I$ matches are returned) or for $\mathcal{F}$ (where N_M matches are returned). N_M can be specified as an exact, minimum or maximum requirement. The distinction between per-image and entire-set correspondence allows us to define problems which treat a set of images as a single input (such as image registration), require matches from some but not all images, and require unique matches across the set or the images. We also include an option to allow search

Table 2. Example variable values and the problem conditions described using `Detect`. N can easily be substituted for $|\mathcal{I}|$ to apply to all images rather than a subset.

Example	Search Space	N_D	N_I	Problem Description
(a)	*Image*	1	1	Single detection in single image
(b)	*Image*	1	N	Single detection in each of N images
(c)	*Image*	K	1	K detections in single image
(d)	*Image*	K	N	K detections in each of N images
(e)	*Set*	1	$\|\mathcal{I}\|$	Single detection from the set of images
(f)	*Set*	K	$\|\mathcal{I}\|$	K detections from the set of images
(g)	*Set*	K	N	K detections from subset of N images

within the feature's source image (by default this is not the case) and an option for the trade-off between feature strength against density of search.

An important aspect of the correspondence problem is applying the correct method to account for variances across images. We can allow for spatial variance and constrain the search for a match in other images using some distribution over the surrounding area centred at the current feature. Other appearance-based properties can be defined, such as variance in blur, intensity, scale, colour, etc. which will aid in the selection of an appropriate method to determine correspondence.

`Match` provides an abstraction over correspondence, which can be used as an input to another operation to define a different task: typically it is used in conjunction with `Solve`. In Section 4 we explore this relationship, examining the problems which can be expressed using the two operations together with each set of conditions.

We use the following notation for `Match`:

$$\texttt{Match}\ (\mathit{Image}|\mathit{Set},\ \mathit{Exact}|\mathit{Min}|\mathit{Max})\ [N_M, N_I]\ \mathit{variances},\ \mathit{images} \quad (1)$$

The user can choose between *Image* and *Set* for correspondence search, and then *Exact*, *Min* or *Max* for the interpretation of N_M. The *variances* are specified as a distribution over a range (e.g. uniform, Gaussian) and the input is the set of *images*. If N_I is zero, the operation will only return matches in the image from which the feature was generated (regardless of search space used). If N_I is one, the operation will return matches from one *other* image from the image where the feature was generated. Table 1 outlines some possible descriptions and corresponding results for our `Match` variables.

Detect: This operation is similar to `Match` except instead of conceptually matching all features to all others, it finds image regions in the set of images which match a user-supplied template. The template may be an example image or a high-level description of a detection problem. It has similar constraints to `Match` and provides a set of detected image regions which match the provided template. As with matching, a distinction must be made whether the number of detections is in the context of every image or across the set of images.

We use the following notation for `Detect`:

$$\texttt{Detect}\ (Image|Set,\ Exact|Min|Max)\ [N_D, N_I]\ template,\ images \quad (2)$$

Table 2 specifies a few of the different forms of detection which may be expressed using this abstraction. In (b), we specify a per-image search and ask for a single result from a single image: this describes a search for a particular region throughout a set of images, and returning the most likely detection. Example (c) goes one step further and requests a single detection in each image. If this were to be qualified with Min then the result would be at least one detection, however likely or unlikely, from each image. (d) presents an interesting case, where multiple detections are requested in a single image. The user does not choose which image this is: rather the framework decides which image had the best detections and chooses these. From the table it can also be seen that the descriptions in (b) and (f) are equivalent, since we are asking for a single detection from any image (but only one) from the set in (b), and we are asking for a single detection across the set of images in (f).

Solve: The `Solve` operation covers a wide range of functionality representative of optimisation algorithms. Within the context of the computer vision problems which we have so far explored the two solutions which may be solved for are spatial transforms and correspondences. The role of this operation will continue to expand as we abstract more problems and methods (e.g. we are working on the problem of matting, where `Solve` is used to optimise the boundary between two image regions). In both cases the input is a set of correspondences from `Match` or a set of detections from `Detect`. The operation's conceptual task description is slightly different from those previous, because the *type* of output requested is used as part of the description: currently we use the types of *transforms* and *matches*. We also provide a variable N_S to define how many solutions are requested (although sometimes this is not required).

There are two distinct models for solutions returned by `Solve`: *Local* and *Global*, and the meaning is dependent on the current context. If solving for a transform with correspondences, local will return a transform per match (e.g. optical flow) and global will return a transform per image (e.g. image registration). If more than one match per feature is available, N_S is used to determine how many solutions should be returned. This allows the solution operation to take existing matches into account and optimise over these as additional information and provide the best solution. There is no problem type defined for finding a solution using detections as input.

`Solve` may also be used to optimise the number of correspondences by constraining them to produce a subset of correspondences which are more accurate with respect to the task, or to provide the most likely path through a set of images for a given match/detection (which is a form of tracking for a constraint down to a unique match per image, although this is not very sophisticated).

We use the following notation for `Solve`:

$$\texttt{Solve}\ (Local|Global)\ [N_S]\ (matches|detections)\ (transforms|matches) \quad (3)$$

Table 3. Problem types when sequencing a `Solve` with a `Match` operation. Registration and optical flow become the most apparent choices for these scenarios, however with additional abstractions this may lead to structure-from-motion, self-calibration and 3D fusion.

Sequence	Table 1	Output Type	Constraint	Problem
(i)	(c)	Transform	Global	Registration [4]
(ii)	(e)	Transform	Global	Stochastic Registration [9]
(iii)	(b)	Transform	Local	Image differencing
(iv)	(c)	Transform	Local	Optical Flow [12]
(v)	(e)	Transform	Local	Stochastic Optical Flow [11]
(vi)	(e)	Matches	Local	Feature Tracking (local matches as prior)
(vii)	(e)	Matches	Global	Feature Tracking (all matches as prior)

The solve operation is used in conjunction with other operators. In Section 4 we explore the relationship of the solve operator in conjunction with other operations.

4 Sequencing Operations

Interpreting the sequence of our operations (and their associated inputs, outputs and parameters) to infer the problem and select an appropriate method to solve that problem is one of our contributions. Combining the operators `Match` and `Solve` allows for the description of an even greater range of vision problems. As with our investigation of detection, we have explored the intricacies of each set of options on the vision problem. The flexible nature of our operations also leads to combinations of options which are not associated with specific or well known vision problems. We hope this will lead us to novel solutions to problems which may be solvable with combinations of existing methods, or to provide descriptions of problems which have yet to be investigated.

Table 3 demonstrates the different problem types when sequencing `Match` then `Solve` operations for various parameters, based on the `Match` examples defined in Table 1. The example in Table 3(i) states that given a set of images, find a single correspondence in each of N images for each $f \in I_0$, then globally solve for a single transform per-image: this is a basic image registration. The same formulation with a local solve would produce a set of transforms which provide a measure of optical flow, defined in Table 3(iv). Variations of the parameters allow us to describe image differencing, shown in Table 3(iii); we can also ask for more than one match so that we can optimise for the best match later when more data is available (Table 3(ii) and 3(v)).

When solving for a set of correspondences the constraints placed on the optimisation guide the reduction of or path through correspondences. We may use the set of matches found for a given set of images as the prior for optimising the path over the matches, for tracking, or for pruning the number of matches using

the appearance models of the matches as a prior to solve for the best match. For example a set of detected objects with multiple detections per image may be constrained by the last known position and motion model of a previous detection in order to improve detection or to track an object. Similarly a set of features may be constrained to reduce the set of features while maintaining features across the image as we see in adaptive non-maximal suppression for image registration [4]. The examples in Table 3(vi) and 3(vii) are for the case where `Solve` is asked to produce *matches*, in the case where `Match` returns multiple *matches* per feature. The result is an optimisation of the path through the images for each feature; for local, each path is evaluated individually, and for global each path is evaluated with knowledge of the others.

5 Conclusion

We have presented our novel abstraction for various computer vision tasks through our small and flexible set of operations which may be sequenced to infer a larger problem. Our research is in the preliminary stages, investigating the effectiveness of our abstraction for describing various low-level tasks within vision with a view to expanding in the future to encompass successively more sophisticated problems. With the detailed representations of `Match`, `Detect` and `Solve` we have been able to describe correspondence, image registration, optical flow, detection and primitive tracking. After the descriptions have been analysed and the problem inferred, the abstraction may select an appropriate method to solve the user's problem.

This is a small part of a very large problem within computer vision, and we are working to expand the language model, notation and the abstraction to cover more issues, and expand the utility of our OpenVL framework. We are simultaneously creating an implementation of the OpenVL framework which provides the language model coupled with implementations of the vision tasks it abstracts. With this framework we hope to provide computer vision to a much larger audience in an intuitive and accessible manner.

References

1. Afrah, A., Miller, G., Fels, S.: Vision system development through separation of management and processing. In: Workshop on Multimedia Information Processing and Retrieval. IEEE, Los Alamitos (2009)
2. Afrah, A., Miller, G., Parks, D., Finke, M., Fels, S.: Hive a distributed system for vision processing. In: Proc. 2nd International Conference on Distributed Smart Cameras (September 2008)
3. Bradski, G., Kaehler, A.: Learning OpenCV: Computer Vision with the OpenCV Library, 1st edn. O'Reilly Media, Inc., Sebastopol (2008)
4. Brown, M., Lowe, D.G.: Recognising panoramas. In: Proceedings of the Ninth IEEE International Conference on Computer Vision, October 16, vol. 2, pp. 1218–1225 (2003)

5. Camellia, http://camellia.sourceforge.net/
6. Clouard, R., Elmoataz, A., Porquet, C., Revenu, M.: Borg: A knowledge-based system for automatic generation of image processing programs. IEEE Trans. Pattern Anal. Mach. Intell. 21, 128–144 (1999)
7. Day, J.D., Zimmermann, H.: The OSI reference model. Proceedings of the IEEE 71, 1334–1340 (1983)
8. Firschein, O., Strat, T.M.: Radius: Image Understanding For Imagery Intelligence. Morgan Kaufmann, San Francisco (1997)
9. Fitzgibbon, A.W.: Stochastic rigidity: Image registration for nowhere-static scenes. In: IEEE International Conference on Computer Vision, vol. 1, p. 662 (2001)
10. Gandalf, http://gandalf-library.sourceforge.net/
11. Gupta, S., Gupta, E.N., Prince, J.L.: Stochastic formulations of optical flow algorithms under variable brightness conditions. In: Proceedings of IEEE International Conference on Image Processing, vol. III, pp. 484–487 (1995)
12. Horn, B.K., Schunck, B.G.: Determining optical flow. Artificial Intelligence 17(1-3), 185–203 (1981)
13. Kohl, C., Mundy, J.: The development of the image understanding environment. In: in Proceedings 1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 443–447. IEEE Computer Society Press, Los Alamitos (1994)
14. Konstantinides, K., Rasure, J.R.: The khoros software development environment for image and signal processing. IEEE Transactions on Image Processing 3, 243–252 (1994)
15. Matsuyama, T., Hwang, V.: Sigma: a framework for image understanding integration of bottom-up and top-down analyses. In: Proceedings of the 9th International Joint Conference on Artificial Intelligence, vol. 2, pp. 908–915. Morgan Kaufmann, San Francisco (1985)
16. Miller, G., Fels, S.: Uniform access to the cameraverse. In: International Conference on Distributed Smart Cameras. IEEE, Los Alamitos (2010)
17. Miller, G., Fels, S., Oldridge, S.: A conceptual structure for computer vision. In: Conference on Computer and Robot Vision (May 2011)
18. Miller, G., Oldridge, S., Fels, S.: Towards a computer vision shader language. In: Proceedings of International Conference on Computer Graphics and Interactive Techniques, Poster Session, SIGGRAPH 2011. ACM, New York (2011)
19. Mundy, J.: The image understanding environment program. IEEE Expert: Intelligent Systems and Their Applications 10(6), 64–73 (1995)
20. Neider, J., Davis, T.: OpenGL Programming Guide: The Official Guide to Learning OpenGL, Release 1, 1st edn. Addison-Wesley Longman Publishing Co., Inc., Boston (1993)
21. Oldridge, S., Miller, G., Fels, S.: Mapping the problem space of image registration. In: Conference on Computer and Robot Vision (May 2011)
22. Panin, G.: Model-based Visual Tracking: the OpenTL Framework, 1st edn. John Wiley and Sons, Chichester (2011)
23. Peterson, J., Hudak, P., Reid, A., Hager, G.: Fvision: A declarative language for visual tracking (2001)
24. Pope, A.R., Lowe, D.G.: Vista: A software environment for computer vision research (1994)

25. Quartz Composer by Apple, http://developer.apple.com/graphicsimaging/quartz/quartzcomposer.html
26. ShapeLogic, http://www.shapelogic.org
27. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, p. 511 (2001)
28. VXL, http://vxl.sourceforge.net/

Girgit: A Dynamically Adaptive Vision System for Scene Understanding

Leonardo M. Rocha, Sagar Sen, Sabine Moisan, and Jean-Paul Rigault

INRIA, Sophia-Antipolis, 2004 Route des Lucioles, BP-93 Sophia-Antipolis, France
Firstname.Lastname@inria.fr

Abstract. Modern vision systems must run in continually changing contexts. For example, a system to detect vandalism in train stations must function during the day and at night. The vision components for acquisition and detection used during daytime may not be the same as those used at night. The system must adapt to a context by replacing running components such as image acquisition from color to infra-red. This adaptation must be dynamic with detection of context, decision on change in system configuration, followed by the seamless execution of the new configuration. All this must occur while minimizing the impact of dynamic change on validity of detection and loss in performance. We present Girgit, a context-aware vision system for scene understanding, that dynamically orchestrates a set of components. A component encapsulates a vision-related algorithm such as from the OpenCV library. Girgit inherently provides loading/caching of multiple component instances, system reconfiguration, management of incoming events to suggest actions such as component re-configuration and replacement of components in pipelines. Given the surplus architectural layer for dynamic adaptation one may ask, does Girgit degrade scene understanding performance? We perform several empirical evaluations on Girgit using metrics such as frame-rate and adaptation time to answer this question. For instance, the average adaptation time between change in configurations is less than $2\,\mu$s with caching, while 8 ms without caching. This in-turn has negligible effect on scene understanding performance with respect to static C++ implementations for most practical purposes.

1 Introduction

Every picture tells a story. One of the primary goals in computer vision is to understand the story in a sequence of pictures. Computer vision practitioners develop algorithms that analyze an individual pictures or video sequences over time to classify patterns and objects into well-known concepts such as cars, people, buildings, and trees. This whole process of analyzing a sequence of images can be subsumed under the topic of *scene understanding*.

A general approach to address the problem of scene understanding for any give image or video is an utopian dream for computer vision scientists. In reality, they develop a multitude of domain-specific algorithms suited to understand specific scenes. These algorithms can be encapsulated into different software components. One possible classification of these components in an image processing pipeline is (a) image acquisition (b) image segmentation (c) blob construction (d) physical object detection (e) tracking and

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 193–202, 2011.
© Springer-Verlag Berlin Heidelberg 2011

(f) action recognition. The configuration of each of these components can vary possibly infinitely in dimensions such as algorithm type, its hardware/software implementation, its parameters and their values and quality of service. The task of scene understanding for a specific context involves placing components (encapsulating algorithm implementations) with appropriate parameters in an image-processing pipeline giving rise to a *vision system*. However, in most situations environmental contexts change. The most common example being the change from day to night and vice versa. This contextual change requires change in configuration of the vision system. The new configuration will possibly use a different set of components and/or parameters to better understand scenes in the new context. However, the very act of dynamic change in configuration of a vision system may result in loss of information or image frames during the adaptation process and raise a question about the continuity and performance of scene understanding. Therefore, we ask: can we dynamically adapt configurations of a vision system at runtime with minimal impact on continuity and performance of scene understanding? This is the primary question that intrigues us. In this paper we present a dynamically adaptive vision system Girgit (signifies Chameleon in Hindi) to address this question.

Girgit is a dynamically adaptive software framework that leverages the dynamic language abilities of the Python programming language. In this paper, tailor Girgit to perform dynamic scene understanding in a video using behavioral components for vision algorithms. Girgit dynamically reconfigures its components, their parameters, or an entire processing chain of components. This dynamic architecture of Girgit appears to be a significant overhead when compared direct static implementations in C++ that leverage performance. Or, is it? This is the question we address through empirical studies on Girgit.

We perform empirical studies to validate Girgit for performance in terms of adaptation time and frame rate. We observe that mean adaptation time between configurations is 6 ms without component caching and less than 2 μs with caching. This negligible adaptation time has very little effect on frame rate. However, memory usage increases in the case of caching. The extra use of memory to cache components is a trade-off for higher performance.

We may summarize the contributions in the paper as follows:

Contribution 1: Using Girgit we demonstrate that it is possible to build dynamically adaptive vision systems that change with context
Contribution 2: We also demonstrate through experimental validation that dynamic adaptation has negligible effect on QoS parameters such as frame rate, and adaptation time.

The paper is organized as follows. Related work in presented in Section 2. In Section 3, we present some foundational material to understand vision systems and dynamic adaptation. In Section 4, we present Girgit's architecture. In Section 5 we present the empirical evaluation of Girgit. We conclude in Section 6.

2 Related Work

The Girgit framework solicits ideas from three areas: vision systems, dynamically adaptive systems, and empirical studies to validate it.

There are a number of vision systems developed in academia and available in the market. An entire magazine entitled *Vision Systems Design* [6] deals with state of the art in vision systems around the world for applications such as surveillance, rescue/recovery, 3D imaging, and robotics. A complete review of each of these systems is out of the scope of this paper. It is however, important to note that most vision systems developed cater to a specific application domain. The Cell Tracker from Carnegie Mellon University [1], for instance tracks stem cells in assays to detect events such as mitosis. The cell tracker uses a fixed set of vision algorithms to achieve this. Dynamic adaptation in Vision Systems is usually oriented to adaptation of the algorithms to variations in context, like in [7]. Our goal, in this paper is go a step further and develop dynamically adaptive vision systems that are generic and can adapt to different contexts and domains by means of changing or adding during runtime components encapsulating algorithms.

Building dynamically adaptive software is a hot area of work in software engineering. This interest in dynamic adaptation comes with maturity in component-based/service-oriented software, dynamic and introspective languages such as Python, and distributed publish-subscribe systems [4]. Dynamic adaptation between a number of components with different parameters presents a large space of variability that is best managed using a high-level model such as in [2] (project DIVA), [11]. Models@runtime [9] is the current trend to manage/reason about dynamically adaptive software. Examples of dynamically adaptive software frameworks include MOCAS [3], a UML based framework for autonomic systems, and RAINBOW [5], that uses graphs to describe the system and the transitions between configurations. Girgit is a similar framework built on Python. We maintain a specification of the adaptive system configuration which is a Model@runtime. Girgit introspectively adapts to the changes made to the model. We apply Girgit's framework to the case study of a vision system.

We evaluate Girgit for QoS using empirical studies. Empirical studies in dynamically adaptive systems validate its functional and non-functional behaviour by exploring the domain of its variability [8] [10]. For the frame rate can be a critical issue in some vision systems, in this paper we focus on studying the impact of dynamic adaptation in continuity quantified as frame rate. In our experiments, we cover all possible component configurations of the vision system built within Girgit to asses the impact of dynamic adaptation.

3 Foundations

The foundations to understand and evaluate Girgit has three dimensions (a) architecture of vision systems in Section 3.1 (b) dynamically adaptive systems in Section 3.2 (c) QoS metrics to evaluate Girgit in Section 3.3.

3.1 Computer Vision Systems

A computer vision system extracts information from a video sequence and recognizes pre-defined patterns. It then generates events that notify of an activity, or position of one or more entities in the video.

A simple vision system is shown in Figure 1 (a). Vision system often starts by acquiring one image at a time via the component image acquisition. The image segementation component operates the incoming image to divide the image into multiple related segments. The blob construction component identifies groups of pixels belonging to the same category to create blobs. The component for object detection analyzes a blob to discover semantic objects such as cars, faces, and people. Feature patterns are used to detect objects. The tracking component maintains a history of a detected object's position. Finally, the event generation component uses information from either one or more of the earlier components to generate events such as intrusion detection, or face detection.

There are several vision systems in operation around the world. Examples of vision systems include the Scene Understanding Platform (SUP) developed by our team PULSAR at INRIA Sophia-Antipolis. A commercial predecessor of SUP called Genius is being developed by Keeneo primarily for video surveillance in places such as the Nice Aeroport. Another, vision system for a totally different application of tracking stem cells is the Cell Tracker developed by Carneige Mellon University.

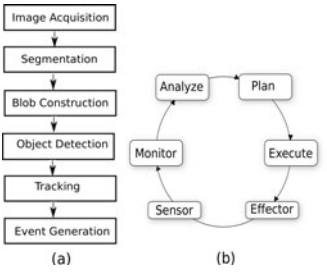


Fig. 1. (a) Vision System (b) MAPE-K View of a Dynamically Adaptive System

3.2 Dynamically Adaptive Software Systems

Dynamically adaptive software systems are built on the monitor-analyze-plan-execute over a knowledge base (MAPE-K) model shown in Figure 1 (b). The MAPE-K loop is a refinement of the Artificial Intelligence community's sense-plan-act approach of the early 1980s to control autonomous mobile robots. The feedback loop is a control management process description for software management and evolution. The MAPE-K loop presented in Figure 1 (b) monitors and collects events, analyzes them, plans and decides the actions needed to achieve the adaptation or new configuration and finally executes reconfigures the software system.

3.3 QoS Metrics

In this paper, we evaluate Girgit based on non-functional Quality of service metrics. We define the metrics as follows:

1. *Frame Rate* - It is the number of frames per second (fps) processed by a chain of vision components at the output.
2. *Adaptation Time* - The time it takes to the system to change from the current running configuration to the following taking in account the loading time of the dynamic libraries and components needed to be able to run.

4 The Girgit Dynamic Adaptation Framework

Girgit is a lightweight[1] framework that allows dynamic reconfiguration of components in a *processing chain*. Girgit consists of three types of components (a) Core components (b) Behavioral components and (c) Event components. The way in which the components are wired together is described by a *model specification*.

Globally, the core components manage the interaction between the behavioral and event components in a dynamically adaptive fashion. The overall architecture for Girgit is presented in Figure 2.

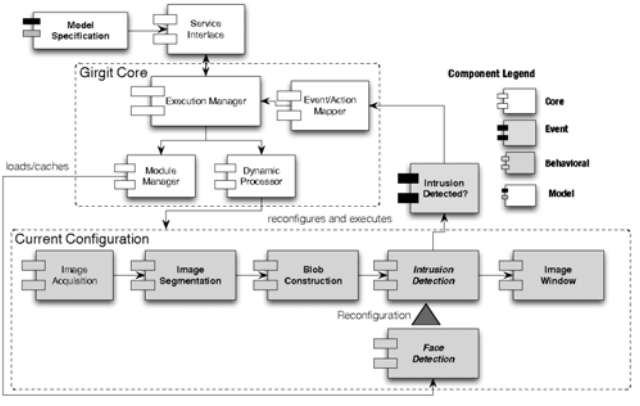


Fig. 2. Girgit's Architecture

4.1 Architecture

Girgit contains core components as shown in Figure 2 and described below:
Model Specification: The model specification specifies the different behavioral, event components, a set of configurations and event/action pairs to change a configuration. A configuration may be defined as a set of behavioral and event components, their parameters, and the interconnection between these components. A model specification is provided using a service interface.
Service Interfaces: Provides the interfaces to interact with Girgit.
Dynamic Processor: Executes the current configuration as described by a model specification and reconfigures either the processing chain or the components.
Execution Manager: Manages the execution. Orchestrates the calls to the Event Manager and the dynamic Processing Chain.
Event Manager: Manages incoming events from event components and suggests actions such as changing of components or complete processing chains. The rules to map events to actions are specified in the model.
Module Manager: Loads and caches instances of components.

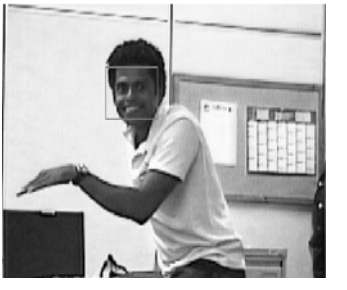
[1] At the submission date 929 lines of Python code according to the sloccount (`http://www.dwheeler.com/sloccount/`) tool

It also contains behavioral components. In this paper, we have behavioral components encoding vision algorithms such as acquisition, segmentation, and blob construction. The event components use information from vision components to return a boolean value for a given event. For instance, if an intrusion detection is an event as shown in Figure 2. A complex event may be encoded in an event component that defines function over boolean values returned by other event components.

4.2 Girgit's Implementation

Girgit is implemented in Python due to its ability to introspect and load modules at runtime/dynamically. There are the following main aspects: (a) The Event Manager checks the events once every loop, if an event matches a rule, then an action must be taken and is informed to the Execution Manager. The actions imply a dynamic reconfiguration act. (b)The dynamic reconfiguration can be applied either to only one component; In this case only one pre-existing component is affected; In the case of a reconfiguration of the chain the graph describing the system is changed and all the components that are needed and do not exist in the system are dynamically loaded (including the needed dynamic libraries). (c) Dynamic resolution of components, all calls to method are resolved in runtime, the Dynamic Processor dynamically finds out the parameters and method to call for every component and manages the data history.

(a) Intrusion Detection



(b) Face Detection

Fig. 3. Dynamic Reconfiguration from Intrusion to Face Detection. Figure 3a is before and Figure 3b is after

4.3 Example Execution in Girgit

We demonstrate dynamic adaptation in Girgit using Figure 3. Girgit starts in intrusion detection mode which is the initial and current configuration. When an intrusion is detected as shown in Figure 3a, an event is generated by an event component as shown in Figure 2. The event manager then decides the new component for face detection. The execution manager instructs the module manager to load the face detection component from harddisk or from memory cache. Finally, the execution manager requests the dynamic processor to connect the face detection component to blob construction and image window. The face detection process ensues as shown in Figure 3b.

5 Empirical Evaluation

We empirically evaluate Girgit to answer the following questions:
Q1. How long does Girgit take to reconfigure or adapt?
Q2. How does adaptation in Girgit affect continuity of operation?

5.1 Experimental Setup

We evaluate Girgit using a total of six different configurations. Each configuration contains a different set of components and/or their parameter settings. Most components encapsulate libraries in OpenCV such as Pyramid segmentation and HAAR object detection. The primary goal was to build an intrusion detection system that switches to face detection in order identify humans in a scene at runtime. We present the number and name of the different configurations on the left side in Figure 4. The components used in the configurations are shown on the right of Figure 4. We also provide the order in the processing chain for these components in the configuration. For instance, OpenCV AVI reader is first in the order in all configurations. The symbol × indicates absence of the component in the configuration.

Configurations		Configurations Details						
Number	Name	Component	C1	C2	C3	C4	C5	C6
C1	SMOOTH_SEGMENTATION	OpenCV AVI Reader	1	1	1	1	1	1
C2	FGD_SEGMENTATION	Image Smoothing	2	×	×	2	×	×
C3	PYRAMID_SEGMENTATION	FGD Background Subtraction	×	2	×	×	×	2
C4	INTRUSION_DETECTION	Pyramid Segmentation	×	×	3	×	2	×
C5	FACE_DETECTION	HAAR Detection	×	×	×	3	3	3
C6	FACE_DETECTION_FGD	Image Window	3	3	4	4	4	4

Fig. 4. Experimental Configurations

Using the six configurations we perform the following experiments to answer questions Q1 and Q2.

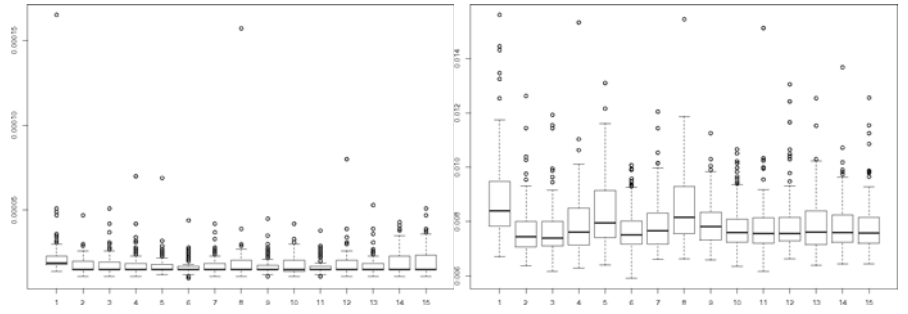
Experiment E1: For a single configuration which is SMOOTH_SEGMENTATION we execute 15 reconfigurations of the same configuration (a) With caching and (b) Without caching. The constant factor here is the configuration that remains fixed. The goal of this experiment is to study stability in adaptation times and frame rate due to dynamic reconfiguration. How close is dynamic reconfiguration to static implementation?

Experiment E2: In this experiment, we execute all pairs of configuration transitions possible using the six available configurations (a) With caching and (b) Without caching. The goal of this experiment, was to introduce variation in configurations and check if this affected adaptation times and frame rate. How stable is the adaptation time when configurations change? The configurations were changed every 5 seconds.

For both experiments we measure the frame rate and adaptation times. We execute the same experiment 100 times to validate the stability of our results. The input video was a long sequence from an office space where there are multiple people entering and leaving the scene. The experiments were executed in the following platform: Linux Fedora 14 x86_64, Intel(R) Core(TM) i7 CPU Q720 @ 1.60GHz, Memory: 8GB.

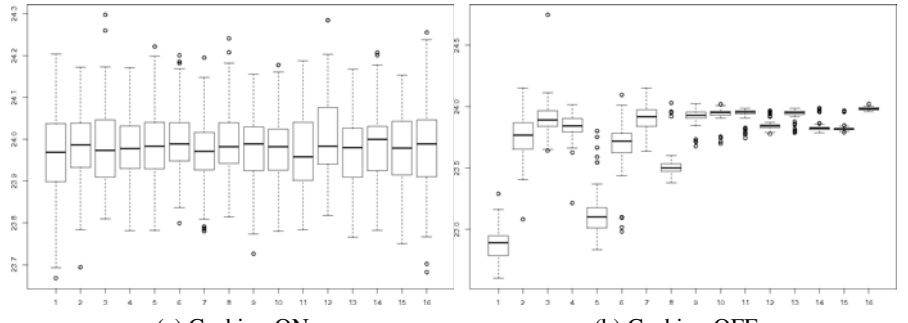
5.2 Results

The results of experiments E1 and E2 are summarized in this section. In Figure 5, we present the adaptation time plots for experiment E1. We observe that there is a



(a) Caching ON (b) Caching OFF

Fig. 5. Boxplot of adaptation time for the SMOOTH_SEGMENTATION configuration with/without caching. X-axis is transition number. Y-axis represent adaptation time in seconds.



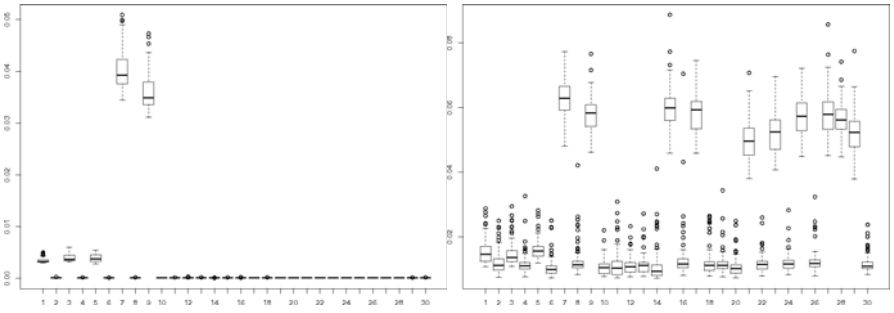
(a) Caching ON (b) Caching OFF

Fig. 6. Boxplot of frame rate for the SMOOTH_SEGMENTATION configuration with/without caching. X-axis is the configuration. Y-axis represent frames per second.

considerable difference between the adaptation times with (Figure 5a) and without (Figure 5b) caching system activated. The caching of components in memory drastically reduces the adaptation time. The mean adaptation time with caching is about 2μs and without is 8 ms. This result addresses question Q1 and demonstrate that Girgit indeed has a low adaptation time.

As a consequence of the low adaptation and reloading time due to caching we observe *no loss* in frame rate as shown in Figure 6a. While without caching there is a small loss (no more than 4%) in frame rate as seen in Figure 6b). These results address question Q2. The continuity in frame rate is largely preserved in Girgit.

In experiment E2, we vary the configurations to see its affect on adaptation time and frame rate. As seen in Figure 7a, the caching has high adaptation times for the first few configurations as components are loaded and cached. However, after the components are cached the adaptation time drops drastically (0.15 ms at most and less than 2 μs mean). However, when the caching system is not used (Figure 7b), the adaptation times are higher peaking at 16 ms and with a mean of 8 ms. This results sheds light on question Q1. It demonstrates that when configurations change caching allows reduction in



(a) Caching ON (b) Caching OFF

Fig. 7. Adaptation times for the 30 possible configuration transitions (transition every 5 seconds) all pairs of the 6 configurations with/without caching. X-axis represent the transitions, Y-axis represent adaptation time in seconds

adaptation times later in the runtime life of Girgit. With respect to Q2 there is also more stability in the adaptation times. The frame rate with caching is also stable (not shown in the paper) when multiple configurations change.

5.3 Discussion and Threats to Validity

The experiments performed on Girgit is within certain bounds. We execute experiments for six vision components. Are our results regarding stability and continuity valid for a large number of different components? This question about scalability can be answered only by creating and running several hundred components or variations of the same component. The logic in the components are libraries in OpenCV. Girgit has limited control over the internals of these libraries. Are these third party components managing memory correctly? We have verified this for the six components in the experiments. However, using components with badly managed memory can result in errors in experimental observations such as memory usage. We demonstrate continuity of Girgit in terms of frame rate for a given configuration. However, continuity can have different semantics. For instance, continuity of tracking an object when context changes. There is yet to analyse the impact of such an adaptive system when a reconfiguration occurs in terms of reliability in cases where history is used in the algorithms for tasks such as tracking and online learning. For stateless algorithms there is no impact of a dynamic reconfiguration event.

The caveats of implementing this kind of systems heavily depend on the algorithms used and the dependences and incompatibilities that might arise between them. Model Driven approaches such as [2], [3], [11] and [5] deal with this issues, but none of them actually study the performance impact as the current work.

Finally, we perform experiments using a long video sequence from a large office with several people coming in and out of a scene. We need to validate Girgit for various scenarios and video sequences.

6 Conclusion

We build a dynamic adaptive vision system using Girgit that clearly separates the dynamic adaptation details from the actual vision components. With an empirical study we demonstrate that there is negligible effect on performance due to dynamic adaptation. This is especially true with *caching* is used. The system can adapt during runtime and add new components that where not previously loaded. A Model Driven approach is popular for creating configurations where components are compatible. There are open issues that should be studied. This issues are with runtime errors in components and studies on how to deal with continuity in the case where algoritms need a certain data history.

References

1. Cell Tracking, http://www.ri.cmu.edu/research_project_detail.html?project_id=579\&menu%_id=261
2. Diva project, http://www.ict-diva.eu/diva/
3. Hameurlain, C.B.N., Barbier, F.: Mocas: a model-based approach for building self-adaptive software components. In: ECMDA (2009)
4. Eugster, P., Felber, P.A., Guerraoui, R., Kermarrec, A.M.: The many faces of publish/subscribe. ACM Computing Surveys 35, 114–131 (2003)
5. Garlan, D., Cheng, S.W., Huang, A.C., Schmerl, B., Steenkiste, P.: Rainbow: architecture-based self-adaptation with reusable infrastructure. Computer 37(10), 46–54 (2004)
6. http://www.vision-systems.com/index.html
7. KaewTrakulPong, P., Bowden, R.: A real time adaptive visual surveillance system for tracking low-resolution colour targets in dynamically changing scenes. Image and Vision Computing 21(10), 913–929 (2003), http://www.sciencedirect.com/science/article/pii/S0262885603000763
8. Kattepur, A., Sen, S., Baudry, B., Benveniste, A., Jard, C.: Variability modeling and qos analysis of web services orchestrations. In: Proceedings of the 2010 IEEE International Conference on Web Services, ICWS 2010, pp. 99–106. IEEE Computer Society, Washington, DC, USA (2010), http://dx.doi.org/10.1109/ICWS.2010.40
9. Morin, B., Barais, O., Jezequel, J.-M., Fleurey, F., Solberg, A.: Models@ run.time to support dynamic adaptation. Computer 42(10), 44–51 (2009)
10. Perrouin, G., Sen, S., Klein, J., Baudry, B., Le Traon, Y.: Automatic and scalable t-wise test case generation strategies for software product lines. In: International Conference on Software Testing (ICST). IEEE, Paris (2010), http://www.irisa.fr/triskell/publis/2010/Perrouin010a.pdf
11. Zhang, C.: Model-based development of dynamically adaptive software. In: ICSE 2006 Proceedings of the 28th International Conference on Software Engineering. ACM, New York (2006)

Run Time Adaptation of Video-Surveillance Systems: A Software Modeling Approach

Sabine Moisan[2], Jean-Paul Rigault[1,2], Mathieu Acher[1], Philippe Collet[1], and Philippe Lahire[1]

[1] I3S, CNRS and University of Nice, France,
first.last@i3s.unice.fr
[2] INRIA Sophia Antipolis Méditerranée, France,
first.last@inria.fr

Abstract. Video-surveillance processing chains are complex software systems, exhibiting high degrees of variability along several dimensions. At the specification level, the number of possible applications and type of scenarios is large. On the software architecture side, the number of components, their variations due to possible choices among different algorithms, the number of tunable parameters... make the processing chain configuration rather challenging. In this paper we describe a framework for design, deployment, and run-time adaptation of video-surveillance systems—with a focus on the run time aspect. Starting from a high level specification of the application type, execution context, quality of service requirements... the framework derives valid possible system configurations through (semi) automatic model transformations. At run-time, the framework is also responsible for adapting the running configuration to context changes. The proposed framework relies on Model-Driven Engineering (MDE) methods, a recent line of research in Software Engineering that promotes the use of software models and model transformations to establish a seamless path from software specifications to system implementations. It uses Feature Diagrams which offer a convenient way of representing the variability of a software system. The paper illustrates the approach on a simple but realistic use case scenario of run time adaptation.

1 Introduction

Video-surveillance processing chains are complex software systems, exhibiting high degrees of variability along several dimensions. On the software architecture side, the number of components, their variations due to choices among possible algorithms, the different ways to assemble them, the number of tunable parameters... make the processing chain configuration rather challenging. Moreover, the number of different applications that video-surveillance covers, the environments and contexts where they run, the quality of service that they require increase the difficulty. Finally the context of an application may (and does) change in real time, requiring dynamic reconfiguration of the chain. To make things even more complex, these variability factors are not independent: they are related by a tangled set of strong constraints or weaker preferences.

This huge variability raises problems at design time (finding the configurations needed by the chain, foreseeing the different possible contexts), at deployment time

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 203–212, 2011.
© Springer-Verlag Berlin Heidelberg 2011

(selecting the initial configuration), and at run time (switching configurations to react to context changes). Many efforts have been made to build libraries or platforms of reusable components for video analysis algorithms. However, assembling a chain for a given application and controlling its configuration at run time remains a tricky issue. Beyond reusing components, one needs to also reuse such concerns as design plans, application templates, typical configurations, etc. This requires to raise the abstraction level. Our approach is to formalize in an unified way the previously mentioned concerns, their relations, as well as the software components implementing video algorithms.

In this paper we describe a framework for design, deployment, and run-time adaptation of video-surveillance systems—with a focus on the run time aspect. Starting from a high level specification of the application type, execution context, quality of service requirements, the framework derives valid possible system configurations through (semi-)automatic model transformations. At run-time, the framework is also responsible for adapting the running configuration to context changes. The proposed framework relies on Model-Driven Engineering (MDE) methods, a recent line of research in Software Engineering[10]. This approach promotes the use of software models and model transformations to establish a seamless path from high level software specifications to system implementation. Moreover the models can be formally analyzed, thus ensuring consistency and validity of the target system.

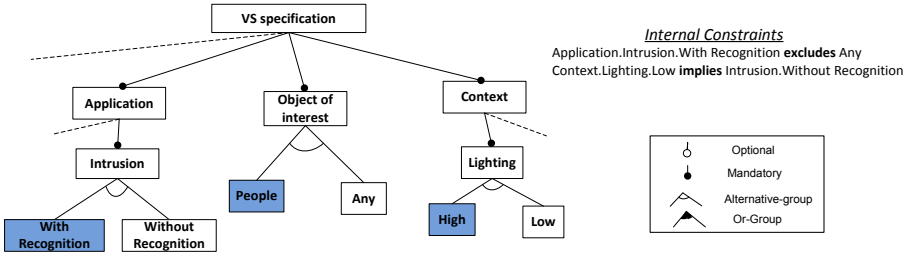
The paper is organized as follows. In the next section we describe the type of software models that we propose to use. Section 3 focuses on the run time adaptation mechanisms; it gives some insights on the methodologies, methods, and tools that we use and sketches an example of a run time adaptation. Section 4 compares our approach with other works.

2 Software Models for Video-Surveillance Systems

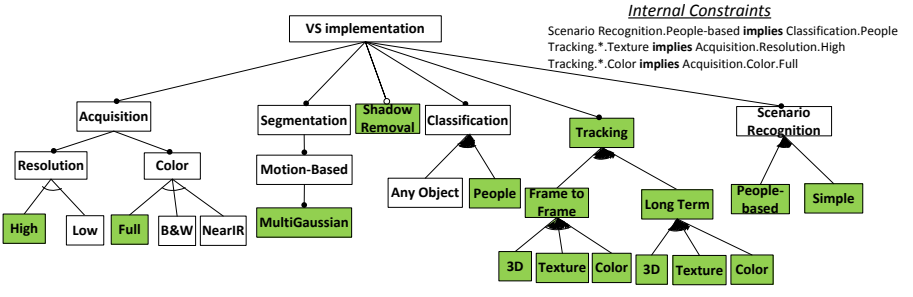
Among all models proposed by MDE, we have chosen the so-called *Feature Diagrams* (FDs) for their ability to represent systems with many possible variation points [18]. Here the "features" correspond to selectable concepts of the systems; they can be at any abstraction level (a feature may correspond to an specification entity such as "Intrusion detection" or to a more concrete element such as "High frame rate"). The features are organized along a tree, with logical selection relations (optional, mandatory features, exclusive choices...) and some constraints that restrict the valid combinations of features (i.e., *configurations*).

Figure 1 gives an example of two feature diagrams, demonstrating their syntax. In the topmost one (Figure 1(a)) all second level features (Application, Object of Interest, Context) are mandatory; People and Any cannot be selected at the same time (alternative group). With the two internal constraints given, the total number of valid configurations of this VS specification feature diagram is 5. The bottom feature diagram (Figure 1(b)) give other examples of the syntax: feature Shadow Removal is optional (whereas its siblings are mandatory); the OR-group of Frame to Frame indicate that any subset of the three features 3D, Texture, and Color is a valid selection.

The tree-like feature diagram is the core of the selection process since it is liable to formal analysis (using propositional logic and satisfiability techniques, see Section 3) and thus leads to valid configurations, by construction.



(a) Excerpt of specification feature diagram



(b) Excerpt of component feature diagram

Fig. 1. Excerpts of feature diagrams for the specification and component models

In the case of video-surveillance, we chose to elaborate two different feature diagrams (see Figure 1): one, the *specification model* (FD_{spec}) is depicted in Figure 1(a) and represents "What To Do", that is the application specification and its context (observation conditions, hardware configuration, objects of interest...); the second, the *component model* (FD_{comp}), is depicted in Figure 1(b) and represents the software components and their assembly, that is "How To Do It". The first model has been obtained after a thorough Domain Analysis relying on our experience in building real video systems. This general model corresponds to the abstract description of the wide range of applications that we wish to address. It is not meant to be modified although we provide a dedicated editor to adapt it if necessary. The second model results from reverse engineering of existing libraries and platforms. It is modified only when the target platform evolves. Each model has its own internal constraints. Moreover, the two models are not independent: they are connected by cross model constraints that formalize the bridge between application requirements and component assemblies that realize them.

The two models are used as follows. First, end users use a simple graphic interface to click-select the features in the specification model that correspond to their application as well as to the possible contexts of evolution. Clearly, this step cannot be automatic, since it corresponds to the specific requirements of a particular real life situation. The outcome is a sub-model (a specialization) of the specification model. Based on the cross model constraints, our framework automatically transforms this sub-model into a sub-model of the component model. The latter represents all possible component configurations of the target video-surveillance system for the given application and contexts

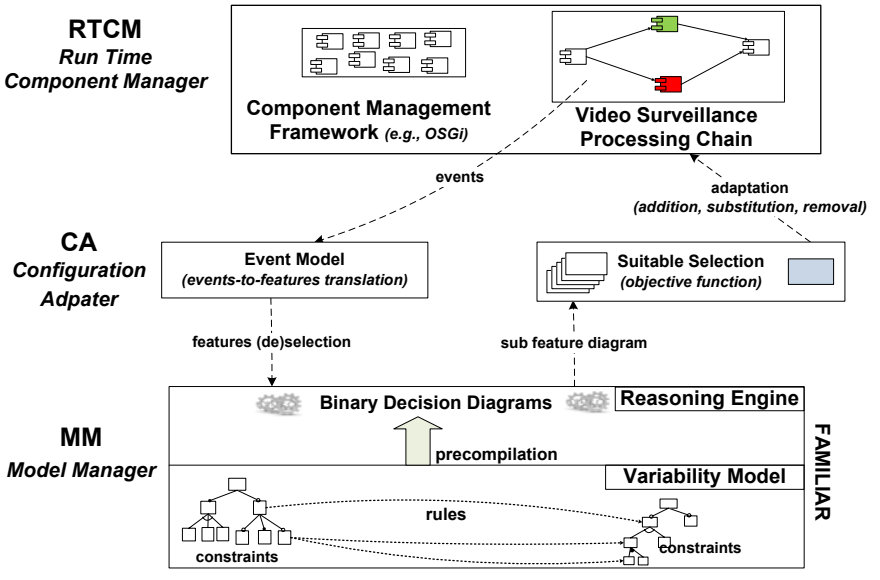


Fig. 2. Run Time Adaptation Architecture

that satisfy the specification and component models and their constraints. Both sub-models will be kept throughout the system life: while the system is running, the two sub-models are used to determine and to apply configuration adjustments in response to context changes.

To deploy and start the system, an initial configuration has to be extracted from the component sub-model, either manually or using some heuristics. This step is not covered in this paper which concentrates on dynamic adaptation.

3 Using Software Models for Run Time Adaptation

3.1 Software Architecture for Run Time Adaptation

To achieve dynamic adaptation, our framework sets up three collaborating modules as shown in Figure 2:

- the **Run Time Component Manager** (RTCM) captures low level events manifesting context changes (e.g., lighting changes); it forwards them to the Configuration Adapter which returns a new component configuration; the RTCM is then responsible for applying this configuration, that is to tune, add, remove, or replace components, and possibly to change the workflow itself.
- the **Configuration Adapter** (CA) receives change events from the RTCM, translates them into feature formalism, and forwards the result to the Model Manager; in return, it obtains a sub-model of component configurations compatible with the change; this sub-FD is a compact representation of a set of valid configurations and

the CA is responsible to select one and to instruct the RTCM to apply it; this selection uses some heuristics, possibly based on a cost function such as minimizing the number of component changes in the processing chain or maximizing the quality of service (e.g., accuracy, responsiveness).

– the **Model Manager** (MM) manages the representation of the two specialized FDs corresponding to the specification and possible component assemblies of the current application together with their constraints; its role is to enforce configuration validity. It is also responsible of the *Event Model* which is a set of rules relating run time events and selection or deselection of features. These rules were elaborated together with the specification and component models. From the CA, the Model Manager receives information about an incoming event; it uses the Event Model rules to select or deselect the corresponding features; it then applies constraints, rules, and model transformations to infer a component sub-model that represents a subset of valid component configurations and that it returns to the CA.

Role of Models at Design and Deployment Time. Before the execution of a system, models are used to verify important properties. Among others, we want to guarantee the *reachability* property, i.e., that for *all* valid specifications, there exists at least one valid software configuration. A brute force strategy which consists in enumerating all possible specifications and then checking the existence of a software configuration is clearly inappropriate, especially in our case where we have more than 10^8 valid specifications and more than 10^6 software configurations. A more scalable technique is to symbolically translate the set of valid configurations of a feature diagram (FD) into a propositional formula ϕ (where each Boolean variable corresponds to a feature) and then to perform reasoning operations on ϕ. In terms of FDs, the reachability property can be formally expressed as follows:

$$\forall c \in [\![FD_{spec}]\!], c \in [\![\Pi_{\mathcal{F}_{FD_{spec}}}(FD_{full})]\!] \quad (1)$$

where $[\![.]\!]$ denotes the set of valid configurations of a FD, FD_{full} is the aggregation of FD_{spec}, FD_{comp} together with cross model constraints, while $\mathcal{F}_{FD_{spec}}$ denotes the set of features of FD_{spec}. Π is the *projection* operator for a FD. Formally, the projection is a unary operation on FD written as $\Pi_{ft_1,ft_2,...,ft_n}(FM_i)$ where $ft_1, ft_2, ..., ft_n$ is a set of features. The result of a projection applied to an FD, FD_i, is a new FD, FD_{proj}, such that: $[\![FD_{proj}]\!] = \{ x \in [\![FD_i]\!] \mid x \cap \{ft_1, ft_2, ..., ft_n\} \}$. The mapping of FDs to propositional formulas [18] and the use of satisfiability techniques allow us to compute the projection of an FD and to check that property (1) holds.

Role of Models at Run Time. During execution, the behavior of the system should be adapted according to contextual changes. A crucial issue is to dynamically select an appropriate software configuration (in terms of features). Again we rely on satisfiability techniques to infer some variability choices by translating FD_{full} (see above) into propositional logic. Features of FD_{full} that are related to *events* are activated or deactivated and the possible values (i.e., true/false) of other features are automatically deduced (by computation of valid domains). In the general case, not all possible values can be inferred and we obtain a sub-FD which compactly represents a subset of the original software configurations. It is then the role of the CA to select a unique configuration from this sub-FD.

Tools for Model Management. The MM relies on FAMILIAR (*FeAture Model scrIpt Language for manIpulation and Automatic Reasoning*) [3], a language dedicated to the management of FDs. In particular, FAMILIAR is used (1) to *model* the variability of the software system and the possible contextual changes ; (2) to *analyze*, at design time, the relationship between FDs and thus ensure some properties of the system in terms of variability ; (3) to *infer*, at run time, a set of valid configurations. Off-the-shelf SAT solvers (i.e., SAT4J) or Binary Decision Diagrams (BDD) library (i.e., JavaBDD) can be internally used to perform FAMILIAR operations. We chose to *precompile* the set of configurations using BDDs[1] which enables a *guaranteed* response time and for which *polynomial* algorithms are available for many operations, for example, the computation of valid domains. As a result, the performance overhead introduced by FDs is negligible. A more costly operation might be the selection of an optimal configuration, depending on the heuristics used.

Tools for Component Management. To perform the physical replacement, removal, or tuning of components, the RTCM relies on a state of the art component management framework, namely OSGi under Eclipse [12]. We are also exploring other "lighter" solutions such as a Python implementation or an INRIA "corporate" framework, DTk.

3.2 Example of a Run Time Adaptation Scenario

To illustrate the approach we now present a simple but realistic use case scenario of run time adaptation. The goal is to detect intrusion in a room (or warehouse) under various illumination conditions. The ideal system thus mainly consists in object detection, with people recognition (to eliminate artifacts) and tracking. However, illumination changes imply dynamic adaptations in the system that may lead to a degraded form of the system. We now detail such an adaptation scenario.

Following the procedure described in section 2, a designer selects from the specification complete model (too big to be displayed in this article) the features corresponding to the application goals and to its possible contexts (in particular the ligthing conditions). The designer also gets rid of the features that do not correspond to any context that can be encountered in the application. The corresponding specification sub-model is (partially) shown on Figure 1(a). The MM automatically verifies its validity. Note that variants remain in this sub-model (in the form of OR nodes) to cope with all possible contexts. Then the transformations (cross model) rules are applied to obtain the component sub-model. For instance, following rule C4 (see table 1(c)), the selection of feature Object of Interest.People in the specification model implies the selection of Scenario recognition.People-based in the component sub-model. The resulting sub-model (displayed in Figure 1(b)) is valid by construction and still contains some variability.

Now we need to choose the initial configuration. Let us suppose that, at the beginning, the scene is under full (artificial) light. The designer then selects the features shown in blue (filled) on figure 1(a) since they correspond to this initial situation. The transformation rules are applied to this configuration. The result is generally a reduced sub-model onto which heuristics or even manual operations (to fine tune parameters,

[1] BDDs are a compact representation of the assignments satisfying a propositional formula and can be used to represent a set of valid configurations of an FD.

Table 1. Examples of specification model rules

(a) Examples of specification model internal rules

1 Object of Interest **defaults** Any
2 Application.Intrusion.With Recognition **excludes** Any
3 Context.Lighting.Low **implies** Application.Intrusion.Without Recognition

(b) Examples of component model internal rules

4 Scenario Recognition.People-based **implies** Classification.People
5 Classification.People **implies** Object of Interest.People
6 Tracking.*.Texture **implies** Acquisition.Resolution.High
7 Tracking.*.Color **implies** Acquisition.Color.Full

(c) Examples of cross model rules

C1 Context.Lighting.Low **implies** Acquisition.Color.Near IR
C2 Context.Lighting.Low **implies** Acquisition.Resolution.Low
C3 Shadow Removal **implies** Context.Lighting.High
C4 Scenario Recognition.People-based **equiv** Object of Interest.People
C5 Scenario Recognition.People-based **equiv** Application.Intrusion.With Recognition

for instance) have to be applied to obtain the initial configuration of the component model. This configuration is also valid, by construction. The initially selected features are shown in green (filled) on figure 1(a).

This configuration is then translated by the RTCM into concrete software components, leading to the following processing chain: *acquisition* with color cameras and high resolution, motion-based *segmentation* using multi-Gaussian, *shadow removal*, *object and people detection* based on size, appearance..., *frame to frame and long-term tracking* based on 3D information, texture and color, and *people intrusion scenario recognition*.

Then suppose that at some time the light switches to "emergency mode". The system has to adapt to this lighting reduction. The corresponding "light dimming" event can be detected by various means: external sensor, internal analysis of the image quality (e.g., during segmentation) or user action. This event is propagated from the RTCM to the MM (through the CA). The MM searches the *Event Model* and finds a rule triggered by this event : **when** Light dimming **select** Context.Lighting.Low. It consequently modifies the Lighting sub-feature in the specification sub-model, changing it from High to Low. Then, internal and cross model rules are applied, causing changes in the component sub-model (tables 1(a), 1(b), 1(c) display a few internal and cross model rules, focusing only on the ones used in this example of runtime adaptation):

- rule C1 selects Acquisition.Color.Near IR (only near infra-red acquisition is possible when light is low)
  - then the contrapositive of rule 7 in turn unselects Tracking.*.Color (frame to frame and long-term tracking can no longer rely on color information)
- rule C2 selects Acquisition.Resolution.Low (reduce resolution to increase pixel size and have more robust local descriptors)
  - then the contrapositive of rule 6 in turn unselects Tracking.*.Texture (frame to frame and long-term tracking can no longer rely on texture either)

- the contrapositive of rule C3 unselects Shadow Removal (of no use when light is low)
- rule 3 selects Application.Intrusion.Without Reco (degraded mode, the system will only detect objects but has not enough light to perform people recognition), which leads to an other series of modifications:
  - unselect Scenario Recognition.People-based (rule C5)
    - ∗ unselect Object of Interest.People (rule C4) and
    - ∗ select Object of Interest.Any (rule 1)
  - unselect Scenario Recognition.People-based (contrapositive of rule 5)

Finally, the new component configuration is sent to the RTCM, which deactivates the no longer used components (*shadow removal* and *people intrusion scenario recognition*) and changes the parameters of the cameras (color and resolution) and of the remaining components (3D box only for tracking algorithms). We obtain a new processing chain, still able to detect object intrusions, but no longer able to precisely recognize people, hence probably leading to more false positive detections; however this is the best that can be done with poor lighting conditions. In addition, the *Event Model* contains a heuristic rule that will warn the system to update the reference image (for motion-based segmentation) and to skip a few frames just after these changes, because both the cameras and the algorithms need some time to adjust to these new conditions.

4 Comparison with Other Works

The existing image and vision libraries propose collections of efficient algorithms [5,1,2]. However, selecting the proper components requires a good knowledge of the intrinsic characteristics of the algorithms, far from the concerns of the target application. Of course, *ad hoc* decision code can solve this problem efficiently for a particular case. Moreover, at the end of the 90's several attempts were made to propose general techniques and tools to bridge the gap between the application requirements and the software implementation [9,6,11]. Some others even address the problem of run time reconfiguration [17,4,13,7,16]. Unfortunately this line of research appears to somewhat slow down. The MDE approach might well favor the revival of this research. This is all the more important since the number of video surveillance installations is exploding: the time between design and deployment must be as short as possible and the run time control should be as automatic as possible. Some surveillance systems (not video) have started to take advantage of the MDE approach [15].

The approach defended in the paper combines and extends techniques developed for models at runtime and software product lines (SPLs). The use of *models at runtime* for specifying and executing dynamically adaptive software systems has proved to help engineers to tame the complexity of such systems while offering a high degree of automation and validation (e.g., see [14]). This approach is generic and application independent. It may also be used to generate adaption code. Last but not least, it can be combined with other complementary paradigms such as learning, inference engines, automata...

Dynamically adaptive systems, such as video surveillance systems, exhibit degrees of variability that depend on user needs and runtime fluctuations in their contexts. The

goal of *dynamic SPLs* is to bind variation points at runtime, initially when software is launched to adapt to the current environment, as well as during operation to adapt to changes in the environment (e.g., see [8]).

5 Conclusion

The primary contribution of the paper is the integration of model-based variability reasoning techniques both for specifying the configurations and controlling the execution of video surveillance systems.

We have tested our approach on simple applications using well-known libraries (OpenCV) on different scenarios. At the moment, 77 features and 10^8 configurations are present in the specification model while 51 features and 10^6 configurations are present in the component model. Once the the video surveillance designer has selected the features required by an application, before deployment, the average number of features to consider at runtime in the component model is less than 10^4. Our experiments show the feasibility of such an approach with a limited performance overhead (if any) compared to traditional run time control where *ad hoc* adaptation code is hardwired and does not rely on the run time availability of an abstract representation of the application and its context evolution.

Yet, many improvements remain to be done. On the component side, we intend to switch from libraries (such as OpenCV) to more component-oriented architectures such as our homemade video-surveillance platform. On the model side, the two feature diagrams and their attached rules and constraints have to be extended and completed to cover more scenarios. On the heuristic side, we intend to develop a panel of intelligent configuration selection heuristics from which a designer could select the most appropriate ones for a given application. These heuristics could be applied both at deployment and at runtime, thus decreasing the need of manual intervention. However, manual operations are likely to be necessary for initial configuration and deployment of a system: many parameters have to be fine tuned and some choices are out of the scope of such a general framework (i.e., stakeholder demands). At run time, if heuristics are not sufficient, manual guidance may be required to select adapted configurations.

Nevertheless, we can already draw the following major advantages of our model-based approach: first, designers can concentrate on their application needs without diving into the technical details of the implementation; second, the same models are seamlessly used from specification to design and to run time adaptation, ensuring system consistency; finally, feature diagrams are a nice and compact way of capturing and reasoning about all the aspects of variability. Indeed, due to its complexity and its huge variability, video-surveillance appears as an ideal candidate for Model-Driven Engineering approaches.

References

1. `http://ltilib.sourceforge.net/doc/homepage`
2. `http://opensource.adobe.com/wiki/display/gil/Generic+Image+Library`

3. Acher, M., Collet, P., Lahire, R., France, R.: A Domain-Specific Language for Managing Feature Models. In: Symposium on Applied Computing (SAC 2011), Programming Languages Track. ACM, Taiwan (2011)
4. Blum, S.A.: From a CORBA-based software framework to a component-based system architecture for controlling a mobile robot. In: Crowley, J.L., Piater, J.H., Vincze, M., Paletta, L. (eds.) ICVS 2003. LNCS, vol. 2626, pp. 333–344. Springer, Heidelberg (2003)
5. Bradski, G., Kaehler, A.: Learning OpenCV: Computer Vision with the OpenCV Library. O'Reilly, Sebastopol (2008)
6. François, A.R.J., Medioni, G.G.: A modular software architecture for real-time video processing. In: Schiele, B., Sagerer, G. (eds.) ICVS 2001. LNCS, vol. 2095, pp. 35–49. Springer, Heidelberg (2001)
7. Georis, B.: Program Supervision Techniques for Easy Configuration of Video Understanding Systems. Ph.D. thesis, Université de Nice-Sophia Antipolis, France (January 2006)
8. Hallsteinsen, S., Hinchey, M., Park, S., Schmid, K.: Dynamic software product lines. Computer 41, 93–95 (2008)
9. Hammes, J., Draper, B., Böhm, W.: Sassy: A language and optimizing compiler for image processing on reconfigurable computing systems. In: Christensen, H.I. (ed.) ICVS 1999. LNCS, vol. 1542, pp. 83–97. Springer, Heidelberg (1998)
10. Kleppe, A., Warmer, J., Bast, W.: MDA Explained: The Model Driven Architecture–Practice and Promise. Addison-Wesley Professional, Reading (2003)
11. Lux, A.: The IMALAB method for vision systems. Machine Vision and Applications (2004)
12. McAffer, J., VanderLei, P., Archer, S.: OSGi and Equinox: Creating Highly Modular Java Systems. Addison-Wesley, Reading (2010)
13. Moisan, S.: Knowledge representation for program reuse. In: European Conference on Artificial Intellignece (ECAI), Lyon, France (2002)
14. Morin, B., Barais, O., Jezequel, J.M., Fleurey, F., Solberg, A.: Models@ run.time to support dynamic adaptation. Computer 42, 44–51 (2009)
15. Le Pors, E., Grisvard, O.: Conceptual modeling for system requirements enhancement. In: Kordon, F., Kermarrec, Y. (eds.) Ada-Europe 2009. LNCS, vol. 5570, pp. 251–265. Springer, Heidelberg (2009)
16. Renouf, A., Clouard, R., Revenu, M.: How to formulate image processing applications? In: International Conference on Computer Vision Systems (ICVS), Bielefeld, Germany, pp. 1–10 (March 2007)
17. SanMiguel, J.C., Bescos, J., Martinez, J.M., Garcia, A.: Diva: a distributed video analysis framework applied to video-surveillance systems. In: 9th International Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS), Klagenfurt, Austria, pp. 207–211 (May 2008)
18. Schobbens, P.Y., Heymans, P., Trigaux, J.C., Bontemps, Y.: Generic semantics of feature diagrams. Computer Networks 51(2), 456–479 (2007)

Automatically Searching for Optimal Parameter Settings Using a Genetic Algorithm*

David S. Bolme[1], J. Ross Beveridge[1], Bruce A. Draper[1], P. Jonathon Phillips[2], and Yui Man Lui[1]

[1] Colorado State University, Fort Collins, CO, USA
[2] NIST, Gaithersburg, MD, USA

Abstract. Modern vision systems are often a heterogeneous collection of image processing, machine learning, and pattern recognition techniques. One problem with these systems is finding their optimal parameter settings, since these systems often have many interacting parameters. This paper proposes the use of a Genetic Algorithm (GA) to automatically search parameter space. The technique is tested on a publicly available face recognition algorithm and dataset. In the work presented, the GA takes the role of a person configuring the algorithm by repeatedly observing performance on a tuning-subset of the final evaluation test data. In this context, the GA is shown to do a better job of configuring the algorithm than was achieved by the authors who originally constructed and released the LRPCA baseline. In addition, the data generated during the search is used to construct statistical models of the fitness landscape which provides insight into the significance from, and relations among, algorithm parameters.

1 Introduction

Recent years have seen significant improvements in computer vision, as demonstrated by measurable progress of standard data sets in areas such as face recognition, object recognition, and action recognition. Much of this improvement comes from combining algorithms within single systems. Therefore, many modern vision systems contain image processing, machine learning, and pattern recognition techniques that work together to solve a specific problem. Unfortunately, tuning these multi-part algorithms is difficult, particularly when changing a parameter in one part of a system may have unforeseen effects on another.

Common practice is to use expert judgment and trial-and-error to search for optimal tunings of parameters. All too often, researchers choose a set of parameters, train the system, and evaluate it on the test data. They then alter a

* This work was funded in part by the Technical Support Working Group (TSWG) under Task SC-AS-3181C. Jonathon Phillips was supported by the Department of Homeland Security, Director of National Intelligence, Federal Bureau of Investigation and National Institute of Justice. The identification of any commercial product or trade name does not imply endorsement or recommendation by Colorado State University or the National Institute of Standards and Technology.

J.L. Crowley, B. Draper, and M. Thonnat (Eds.): ICVS 2011, LNCS 6962, pp. 213–222, 2011.
© Springer-Verlag Berlin Heidelberg 2011

parameter and repeat the process until they "crack" the data set. This process has several problems. Most significantly, the test data becomes an implicit part of the training data. In addition, optimal parameters may still be missed because tests were run on individual components instead of the whole system, or because interactions among parameters were misunderstood. Intuition can also be misleading, with the result being that some good parameters are never tested.

This paper presents a technique that replaces parameter tuning by a human experimenter with a Genetic Algorithm (GA). This has many advantages. The GA can tirelessly evaluate thousands of algorithm configurations, improving the likelihood that the best configurations in the search space will be explored. All parameters are optimized simultaneously, allowing the GA to seek out superior configurations in the presence of complex parameter interactions. The configuration is based on a "fitness function" that evaluates the system as a whole, rather than the performance of subcomponents. The GA is not subject to errors of human judgement that may exclude lucrative regions of the search space.

This research uses a GA to find an optimal configuration for the Local Region Principal Components Analysis (LRPCA) face recognition baseline algorithm, as applied to the the Good, Bad, and Ugly (GBU) challenge problem [11]. This baseline improves on Principal Components Analysis (PCA) by adding pre-processing and post-processing steps as well as multiple subspaces for 14 regions of the face. The results show that the GA configuration outperforms the best known manual configuration and highlights the importance of parameter configurations where performance on the tuning subset varies from 5% to 35%, a factor of 7 change in accuracy simply by tuning parameters[1].

The performance results with the GA parameters must be taken with a grain of salt. These results represent the GA's ability to "crack" the data set, since the test data was evaluated as part of the fitness function. Nonetheless, this is similar to the process followed by many human researchers. Moreover, it enables the second contribution of this paper: the use of a Generalized Linear Model (GLM) to analyze system parameters, revealing which parameters are critical and which sets of parameters are most strongly inter-related. The GA evaluates a system's performance over thousands of parameter value combinations. This creates a treasure trove of test data, which can be mined to determine how system performance is affected by each parameter. In this paper we fit a GLM to the performance data to model how the parameters impact LRPCA performance on the GBU data sets. For the LRPCA algorithm, the results indirectly indicate which regions of the face are most important for the algorithm and have the potential to produce improvements in future versions of the system.

2 Related Work

Parameter tuning for a complex algorithm is a well known problem. Support vector machines (SVMs) are one example of a complex technique with a large

[1] Performance numbers are for the Correct Verification Rate at a 0.001 False Accept Rate.

number of domain-specific parameters, and as a result there are several papers that search for optimal parameters, e.g. [2,10]. Of particular interest is a report by Hsu and Lin [7] in which experts were asked to hand-tune parameters of an SVM, and the results were compared to parameters learned by an automatic grid search of parameter space. In all three cases, the learned parameters outperformed those sets turned by the human experts.

In computer vision where algorithms are built that embody models, parameter estimation is often approached as statistical model fitting. For example, Felzenszwalb's object detector models objects as mixtures of multi-scale deformable parts [4], and much of the technique involves fitting parameters to data. These are "strong" techniques that exploit top-down constraints to guide parameter selection and have proven to be effective when they can be applied.

Unfortunately, the best face recognition algorithms are multi-step systems with interacting components, and the implications of their parameters are often poorly understood. A recent paper by Cox and Pinto [3] uniformly samples parameter space (using many processors) for a face recognition algorithm and shows that the resulting parameters improve performance over hand-tuned configurations. It should be noted that finding optimal parameters differs from the methods of Karurgaru [8] who used a GA to find optimal positions and scales for templates within the face matching process.

Earlier, Givens et al [5] used a generalized linear mixed-effects model (GLMM) to analyze the effects of parameters on an LDA+PCA algorithm [17] but not to search for optimal parameter values. In our approach a GLM is used to model parameter space in a manor similar to Givens et al [5], thereby extracting configuration information about the underlying algorithm. Harzallah et al [6] used a rank-based *Friedman* Test for a similar purpose, however the GLM's model can be better related to the fitness landscape.

3 Searching for Optimal Configurations

3.1 Training, Tuning, and Test Datasets

The Face Recognition Vendor Test 2006 showed that face recognition technology could verify a person's identity with 99% accuracy in high quality images taken under controlled conditions [13]. However, face recognition in uncontrolled conditions is much more difficult. The GBU challenge problem [11] contains three partitions of face images of varying difficulty from uncontrolled environments. The Good partition contains images that are easy to match, while the Ugly partition is extremely difficult, and the Bad partition is somewhere in between. The purpose of GBU is to improve performance on the Bad and Ugly partitions without sacrificing performance on the Good.

The GBU Challenge Problem has a clearly stated protocol for presenting performance results. It requires training be done on an independent set of images that contain no images of the people present in the GBU test data. For this purpose, a set of 673 images from the MBGC Still Image problem [12] that are

disjoint from the people included in GBU is used as a training set. These images are used to train the algorithms basis vectors.

A distinction is drawn between training and tuning. Tuning is the process typically carried out by an algorithm developer where parameters are repeatedly modified and then performance is tested on the challenge imagery. Here, when the GA evaluates the fitness of a particular tuning, it considers the verification rate on a tuning-subset of the actual GBU test data.

If one views the entire GA as a machine learning tool for constructing a better algorithm, the use of the tuning-subset of the test data is a violation of the GBU protocol which requires a separate dataset for training. However, this paper views the role of the GA as a surrogate for what researchers do when tuning algorithms. A goal of this paper is to better understand how tuning-parameters effects performance on a benchmark problem, which requires that the tuning-subset drawn from the test data itself. The tuning-subset used here is composed of approximately 1/6 of the GBU testing images. In future work, the GA will be tested as a method to improve generalized performance, where the GA only has access to training data.

3.2 The LRPCA Baseline Algorithm

The experiments presented use an open source face recognition baseline algorithm called Local Region Principal Components Analysis (LRPCA) [11].[2] LRPCA is based on the well known eigenfaces algorithm [9,14] but includes improvements to the way faces are preprocessed, analyzed, and compared to produce higher accuracy than a simple PCA based approach.

The input to the algorithm is an image containing a face and the coordinates of both eyes. The eye coordinates are used to geometrically normalize the face and the image is then split into 14 smaller images that represent local regions of the face focused on the eyebrows, eyes, nose, mouth, etc. Each region is preprocessed using the Self Quotient Image (SQI) [15] which reduces the effect of lighting, and the pixel values are then normalized to have a mean of 0.0 and a standard deviation of 1.0.

PCA is run on each region to produce a set of basis vectors. A configurable number of eigenvectors can be dropped corresponding to both the largest (PCA Min) and smallest (PCA Max) eigenvalues to further reduce the effect of illumination and noise. This dimension reduction allows the algorithm to better generalize. LRPCA optionally whitens the basis vectors such that, when projected, the training data has a variance of 1.0 in all dimensions.

A weight is also computed for each basis vector which is the between-class variance divided by the within-class variance (σ_b^2/σ_w^2). Vectors with the largest weight are kept where the total number of vectors is a configurable parameter (Total Dimensions). This weight is also used to emphasize vectors that better discriminate among people.

During testing, new faces are normalized and projected onto the basis and the similarity between the faces is measured using correlation. LRPCA was extended

[2] http://www.cs.colostate.edu/facerec/algorithms/lrpca2010.php

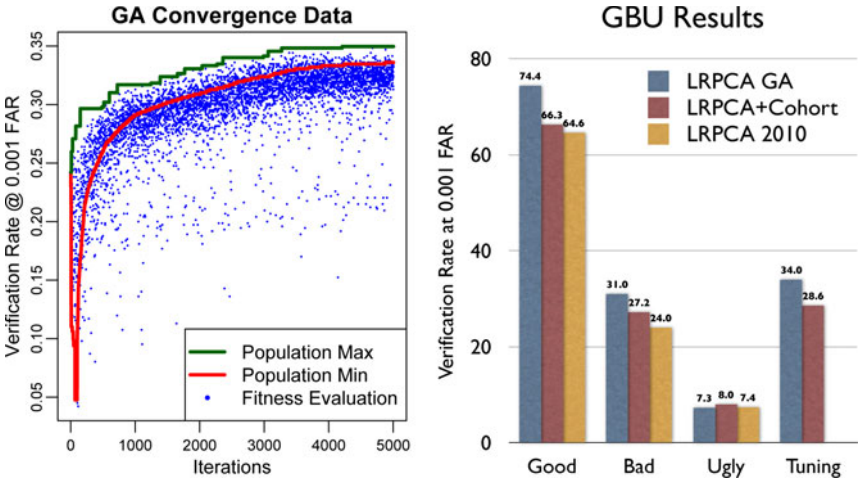


Fig. 1. The left plot shows the convergence of the GA where each blue dot is one fitness function evaluation. The right plot compares the GA tuned algorithm to the manual tuned equivalent (LRPCA+Cohort) and the standard configuration (LRPCA 2010).

with cohort normalization [1] which offers a slight improvement to the verification rates shown in Figure 1. This is done by computing the similarity between each testing image and faces in the training set. The non-match distribution can then be normalized using the following equation:

$$s'(i,j) = \frac{s(i,j) - \frac{1}{2}(\mu_i + \mu_j)}{\frac{1}{2}(\sigma_i + \sigma_j)} \tag{1}$$

where $s(i,j)$ is correlation, and μ_i and σ_i are the mean and standard deviation of non-match scores for test images i and j estimated from the cohort set.

3.3 Genetic Algorithm and Configuration Space

The parameter space was optimized by a rank-based genetic algorithm similar to GENATOR [16] available as part of the PyVision library[3]. Genetic algorithms are stochastic optimization techniques inspired by evolution and natural selection. Algorithm configurations are represented as individuals in a simulated population, where more fit individuals are selected for survival and breeding. In this experiment, the population contains 100 randomly generated individuals and each iteration follows these steps:

1. Two individuals of the population are selected randomly.
2. Those individuals are combined to produce a new individual where configuration parameters are selected randomly from the parents.

[3] http://pyvision.sourceforge.net

Table 1. This table shows the parameters tuned by the GA along with their optimal values

Parameter	Type	Range	Manual Value	GA Value
Region 0: Full Face	Float	0.50 - 1.00	1.00	0.927
Region 1: Left Eye	Float	0.10 - 0.50	0.33	0.433
Region 2: Right Eye	Float	0.10 - 0.50	0.33	0.342
Region 3: Far Left Brow	Float	0.10 - 0.36	0.33	0.360
Region 4: Center Left Brow	Float	0.10 - 0.36	0.33	0.285
Region 5: Center Right Brow	Float	0.10 - 0.36	0.33	0.286
Region 6: Far Right Brow	Float	0.10 - 0.36	0.33	0.360
Region 7: Nose Bridge	Float	0.10 - 0.66	0.33	0.395
Region 8: Nose Tip	Float	0.10 - 0.66	0.33	0.100
Region 9: Left Nose	Float	0.10 - 0.70	0.33	0.211
Region 10: Right Nose	Float	0.10 - 0.70	0.33	0.259
Region 11: Left Mouth	Float	0.10 - 0.20	0.20	0.167
Region 12: Center Mouth	Float	0.10 - 0.20	0.20	0.200
Region 13: Right Mouth	Float	0.10 - 0.20	0.20	0.154
SQI Blurring Radius	Float	0.5 - 20.0	3.0	19.43
PCA Min Dimension	Int	0 - 20	2	19
PCA Max Dimension	Int	100 - 400	250	169
PCA Whitening Enabled	Bool	True/False	True	True
Final Basis Dimensions	Int	100 - 4000	3500	880

3. Small perturbations are made to the new individual to simulate mutation.
4. The new individual is evaluated using the fitness function.
5. If the new individual scores higher than the previously lowest rank individual in the population, that lowest ranked individual is replaced.

The fitness function evaluates each individual by completing the full training and testing process. The algorithm was trained using the configuration in the genetic code and then was evaluated on the tuning-subset at a false accept rate of 0.001. The GA was run on a quad-core Intel i7 with 8 worker processes which completed forty evaluations per hour resulting in 5004 total evaluations. Table 1 summarizes the 19 parameters that were tuned by the GA and also gives the manually selected default parameters as well as the best configuration produced by the GA.

3.4 The Optimal Configuration

Figure 1 shows the GA convergence. Each iteration corresponds to one fitness function evaluation where the fitness scores are shown as blue dots. The green line represents the best known configuration at each iteration and the red line represents the worst configuration in the population. Figure 2 shows how each of the parameters converge throughout the GA run. There are a few parameters where there is no clear preference for any particular value. This suggests those parameters have little effect on performance. Regions 3, 6, and 8 show interesting

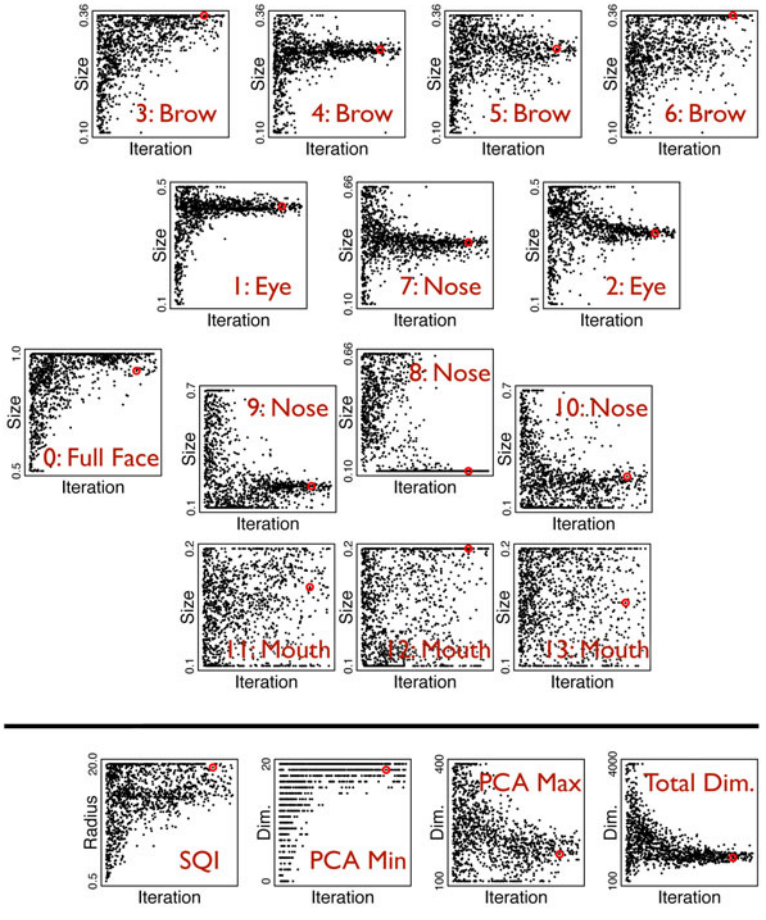


Fig. 2. This figure shows that the configuration parameters converge as optimization progresses. The top shows local region sizes where plots are arranged relative to their locations on the face. The bottom shows the convergence of the radius of the Gausian filter used for the SQI normalization, the minimum and maximum PCA cutoffs, and the total number of basis vectors included in the final configuration. Only configurations added to the population are shown and the best configuration is circled in red (Iteration 4209).

behavior where the best values are at the boundary of the configuration space which suggest the range for those parameters could be expanded.

The best configuration was evaluated on the full GBU challenge problem in Figure 1. This illustrates the benefit of using the GA to search for optimal configurations. Good and Bad performance improved significantly, while the performance on the Ugly partition dropped by a small amount. This suggests that tuning real world systems using GAs may offer important performance increases.

4 Data Mining the Search Space

A more interesting aspect of this work is what the optimization process tells us about the shape of the parameter space. Each fitness evaluation relates a point in that space to a score for the algorithm. During the course of the run the space is sampled thousands of times, with higher density near the optimal solutions.

To understand the configuration space, a GLM was fit to the search results. The response variable $\hat{Y}$ is the score produced by the fitness function and the X_i correspond to the values and squared values of the algorithm parameters:

$$\hat{Y} = \alpha + \beta_1 X_1 + \beta_2 X_1^2 + \beta_3 X_2 + \beta_4 X_2^2 + \cdots \tag{2}$$

The α (intercept) and β_i variables are fit to the dataset to minimize the sum of squared error in the model, which is a second order approximation to the configuration landscape and is used to estimate the importance of each parameter.

Figure 3 illustrates the GLM approximation to the fitness surface. In this case only points that were added to the population are shown. Additionally, whitening always resulted in a better score; therefore, the model was fit to configurations where whitening was turned on. The basic shape of the landscape can be inferred by the top row but the shape is more obvious when the parameters are controlled for by the GLM as shown on the bottom.

This analysis reveals interesting shapes in the fitness landscape. Region 3 suggests a linear response and the range searched by the GA could be extended. Region 7 shows a nice second order response where the best values selected by the GA correspond nicely to the best values suggested by the GLM. Region 8 shows a response curve suggesting the best values may be larger or smaller than what was searched by the GA. Also, a hill climbing approach would not properly optimize this region. The GA, however, maintains multiple configurations in its population and therefore focuses the search on both ends of the range. Total Dimensions are also an interesting case where the model does not appear to fit the data well and suggests a higher order GLM may be necessary.

While the model presented in the previous section is used to understand the effects of the parameters, it is often important to understand which parameters are effecting each other. Again a GLM is the analysis tool, but the new model will add interaction terms. If a and b are parameters, the original model had the terms for a, a^2, b, and b^2, while the new model adds an interaction term $a * b$.

Initially 153 interaction terms were added to the GLM; however, most terms did not contribute predictive information. A greedy local search reduced the model parameters to the minimum needed to accurately represent the data, as measured by Akaike Information Criterion (AIC). This reduced the model to 78 interactions. This number was further reduced by computing the significance of dropping each remaining term from the model which used an F-test. This resulted in 20 terms that were highly significant ($P < 0.001$).

A few parameters were found to repeatedly participate in the most significant interactions. Region 1 participated four times, Region 2 three times, Region 7 four times, Region 8 three times, PCA Min Dimensions three times, PCA Max

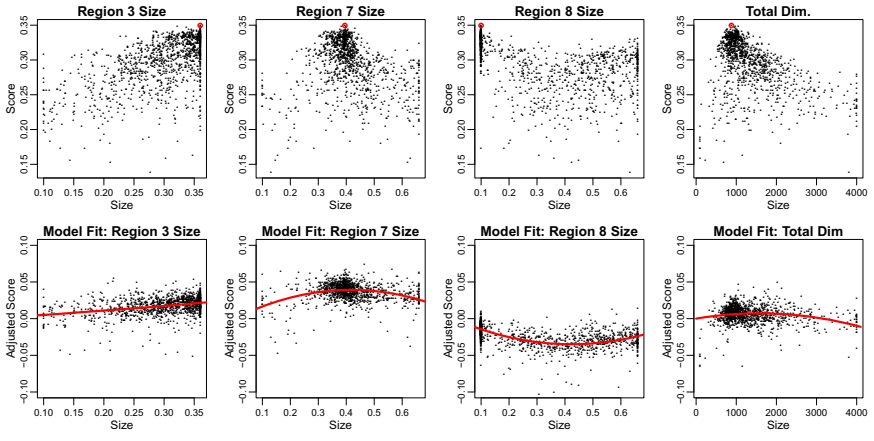


Fig. 3. These figures illustrate the fitness landscape for some of the configuration parameters. The top figures show the raw fitness scores relative to the parameter value. The bottom figures show the adjusted score taking all other parameters into account. The model is shown as a red line.

Dimensions six times, and Total Dimensions six times. The regions participating in these interactions correspond to the eyes, nose bridge, and nose tip which are thought to be the best areas of the face for biometric matching.

5 Conclusions

This paper used a GA to find the optimal parameter settings for the LRPCA algorithm, producing a better configuration than manual tuning. The GA simultaneously optimizes 19 parameters in the context of the complete system, which takes the fitness landscape and parameter interactions into account.

A GLM-based analysis provides additional knowledge of the algorithm by modeling the fitness landscape. This shows when parameters have been set correctly or when additional tuning may be necessary. The analysis identifies which parameters are most significant and which parameters have the strongest interactions. This insight into the parameter space may lead to better performance in future versions of the system.

References

1. Aggarwal, G., Ratha, N.K., Bolle, R.M., Chellappa, R.: Multi-biometric cohort analysis for biometric fusion. In: ASSP (2008)
2. Cherkassky, V., Ma, Y.: Practical selection of SVM parameters and noise estimation for SVM regression. Neural Networks 17(1), 113–126 (2004)
3. Cox, D.D., Pinto, N.: Beyond simple features: A large-scale feature search approach to unconstrained face recognition. In: Face and Gesture (2011)

4. Felzenszwalb, P., Girschick, R., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part based models. T-PAMI (2009)
5. Givens, G.H., Beveridge, J.R., Draper, B.A., Bolme, D.S.: Using a generalized linear mixed model to study the configuration space of a PCA+LDA human face recognition algorithm. In: Perales, F.J., Draper, B.A. (eds.) AMDO 2004. LNCS, vol. 3179, pp. 1–11. Springer, Heidelberg (2004)
6. Harzallah, H., Jurie, F., Schmid, C.: Combining efficient object localization and image classification. In: International Conference Computer Vision (2009)
7. Hsu, C.W., Chang, C.C., Lin, C.J.: A practical guide to support vector classification. From LibSVM (December 2007)
8. Karungaru, S., Fukumi, M., Akamatsu, N.: Face recognition using genetic algorithm based template matching. In: Communications and Information Technology, vol. 2, pp. 1252–1257. IEEE, Los Alamitos (2004)
9. Kirby, M., Sirovich, L.: Application of the karhunen-loeve procedure for the characterization of human faces. T-PAMI 12(1) (1990)
10. Lorena, A., de Carvalho, A.: Evolutionary tuning of svm parameter values in multiclass problems. Neurocomputing 71(16-18), 3326–3334 (2008)
11. Phillips, P.J., Beveridge, J.R., Draper, B.A., Givens, G.H., O'Toole, A.J., Bolme, D.S., Dunlop, J., Lui, Y.M., Sahibzada, H., Weimer, S.: An introduction to the good, the bad, & the ugly face recognition challenge problem. In: Face and Gesture (2011)
12. Phillips, P.J., Beveridge, R., Givens, G., Draper, B., Bolme, D., Lui, Y.M., Teli, N., Scruggs, T., Cho, G.E., Bowyer, K., Flynn, P., O'Toole, A.: Overview of the multiple biometric grand challenge results of version 2. Presentation at Multiple Biometric Grand Challenge 3rd Workshop (December 2009)
13. Phillips, P.J., Scruggs, W.T., O'Toole, A.J., Flynn, P.J., Bowyer, K.W., Schott, C.L., Sharpe, M.: FRVT 2006 and ICE 2006 large-scale results. In: National Institute of Standards and Technology (2007)
14. Turk, M.A., Pentland, A.P.: Face recognition using eigenfaces. In: CVPR (1991)
15. Wang, H., Li, S.Z., Wang, Y., Zhang, J.: Self quotient image for face recognition. In: ICIP (2004)
16. Whitley, D.: The GENITOR algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. In: Int. Conf. on Genetic Algorithms (1989)
17. Zhao, W., Chellappa, R., Krishnaswamy, A.: Discriminant analysis of principal components for face recognition. In: Face and Gesture (1998)

Author Index