Sami Huttunen

METHODS AND SYSTEMS FOR VISION-BASED PROACTIVE APPLICATIONS
SAMI HUTTUNEN

METHODS AND SYSTEMS FOR VISION-BASED PROACTIVE APPLICATIONS

Academic dissertation to be presented with the assent of the Faculty of Technology of the University of Oulu for public defence in Auditorium TS101, Linnanmäki, on 2 December 2011, at 12 noon

UNIVERSITY OF OULU, OULU 2011
Abstract

Human-computer interaction (HCI) is an integral part of modern society. Since the number of technical devices around us is increasing, the way of interacting is changing as well. The systems of the future should be proactive, so that they can adapt and adjust to people’s movements and actions without requiring any conscious control. Visual information plays a vital role in this kind of implicit human-computer interaction due to its expressiveness. It is therefore obvious that cameras equipped with computing power and computer vision techniques provide an unobtrusive way of analyzing human intentions. Despite its many advantages, use of computer vision is not always straightforward. Typically, every application sets specific requirements for the methods that can be applied. Given these motivations, this thesis aims to develop new vision-based methods and systems that can be utilized in proactive applications.

As a case study, the thesis covers two different proactive computer vision applications. Firstly, an automated system that takes care of both the selection and switching of the video source in a distance education situation is presented. The system is further extended with a pan-tilt-zoom camera system that is designed to track the teacher when s/he walks at the front of the classroom. The second proactive application is targeted at mobile devices. The system presented recognizes landscape scenes which can be utilized in automatic shooting mode selection.

Distributed smart cameras have been an active area of research in recent years, and they play an important role in many applications. Most of the research has focused on either the computer vision algorithms or on a specific implementation. There has been less activity on building generic frameworks which allow different algorithms, sensors and distribution methods to be used. In this field, the thesis presents an open and expendable framework for development of distributed sensor networks with an emphasis on peer-to-peer networking.

From the methodological point of view, the thesis makes its contribution to the field of multi-object tracking. The method presented utilizes soft assignment to associate the measurements to the objects tracked. In addition, the thesis also presents two different ways of extracting location measurements from images. As a result, the method proposed provides location and trajectories of multiple objects which can be utilized in proactive applications.

Keywords: human-computer interaction, Kalman filter, object tracking, scene classification, sensor network, shooting mode, smart classroom
Huttunen, Sami, Menetelmä ja järjestelmä konenäköön perustuviin proaktiivisiin sovelluksiin

Oulun yliopisto, Teknillinen tiedekunta, Tietotekniikan osasto; Infotech Oulu, PL 4500, 90014
Oulun yliopisto


Oulu

Tiivistelmä


Asiakirjat:

Preface

The research work for this thesis was carried out in the Machine Vision Group of the Department of Computer Science and Engineering at the University of Oulu during the years 2005-2011.

First of all, I would like to express my gratitude to my instructor, Professor Janne Heikkilä, whose advice and ideas have been invaluable through the years. In addition, I am grateful to him for being so patient and that he has given me also freedom to pursue my own ideas in research.

I am grateful to the reviewers of the thesis, Dr. Markus Koskela and Professor Carlo Regazzoni, for their constructive comments and feedback. I would also like to acknowledge Gordon Roberts for his help with the language revision of the manuscript.

The co-authors, Professor Olli Silvén, Dr. Jari Hannuksela, Dr. Esa Rahtu, Pekka Sangi, Pekka Saastamoinen, Valtteri Takala, and Marko Heikkilä are gratefully acknowledged for providing their contribution to the original publications and system implementations.

The Machine Vision Group has been an excellent place for doing research, and I am grateful to Professor Matti Pietikäinen for offering me the possibility to work in this inspiring group. In general, I would like to thank all my colleagues in the Machine Vision Group for creating a pleasant atmosphere. Especially, I want to mention the relaxing coffee and lunch break discussions on many interesting topics, not always related to our research work.

Financial support for this thesis has come from the Infotech Oulu Graduate School, the Academy of Finland, the Emil Aaltonen Foundation, the Oulu University Scholarship Foundation, the KAUTE foundation, the Tauno Tönning Foundation, the Riitta and Jorma J. Takanen Foundation, and Tekes (the Finnish Funding Agency for Technology and Innovation). This support is gratefully acknowledged.

Finally, I would like to thank my parents, father Tapio and mother Kaija, for their support and encouragement over the years. I also wish to thank my sister Sanna and all my sports-minded friends for the moments outside the research world.

Oulu, October 2011

Sami Huttunen
Abbreviations

Γ  Disturbance matrix
ε  Process noise
η  Observation noise
λij Measurement assignment variable
πj  A priori probability of associating a measurement with the object j
Φ  State transition matrix
"  Chi square distance of histograms

A  Close-view camera’s coverage area
a  Learning rate
dh  Height of an object
dw  Width of an object
f  Fading factor
H  Measurement matrix
I  Identity matrix
i  Scalar index of an observation, i = 1, ..., N
j  Scalar index of an object, j = 1, ..., M
K  Kalman gain matrix
li  Observation i of the true state
M  Number of objects
N  Number of observations
P  Covariance matrix of state estimation uncertainty
Q  Covariance matrix of process noise
R  Covariance matrix of measurement noise
t  Time counter for camera switching
tk  Minimum time for showing a regular camera view
to  Minimum time for showing the object on the document camera
wij  Weight of the position estimate li for the object j
(x, y) Location in image coordinates
xj  State vector of the object j
\hat{x}_j  A priori estimate of x_j
$\hat{x}_j$    A posteriori estimate of $x_j$
$z$    Vector of observations

ACU    Auditorium Control Unit
AmI    Ambient Intelligence
AP    Average precision
AUC    Area under curve
AV    Audio-visual
BOW    Bag-of-words
CAMSHIFT    Continuously adaptive mean shift
CAVIAR    Context aware vision using image-based active recognition
CIF    Common intermediate format, $352 \times 288$ resolution
DEA    Distance education assistant
DSC    Distributed smart camera
EM    Expectation maximization
FOV    Field-of-view
FP    False positive
FPR    False positive rate
HBMA    Hierarchical block matching algorithm
HCI    Human computer interaction
HMM    Hidden Markov model
HOG    Histogram of oriented gradients
IP    Internet protocol
JPDAF    Joint probabilistic data association filter
LBP    Local binary pattern
MHT    Multiple hypothesis tracking
OpenCV    Open source computer vision library
P2P    Peer-to-peer
PC    Personal computer
PPS    Pedestrian protection system
PTZ    Pan-tilt-zoom
QVGA    Quarter video graphics array, $320 \times 240$ resolution
ROC    Receiver operating characteristic
ROI    Region of interest
SVM    Support vector machine

10
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>TCP</td>
<td>Transmission control protocol</td>
</tr>
<tr>
<td>TP</td>
<td>True positive</td>
</tr>
<tr>
<td>TPR</td>
<td>True positive rate</td>
</tr>
<tr>
<td>UI</td>
<td>User interface</td>
</tr>
<tr>
<td>VGA</td>
<td>Video graphics array, 640×480 resolution</td>
</tr>
<tr>
<td>VSN</td>
<td>Visual sensor network</td>
</tr>
<tr>
<td>XML</td>
<td>Extensible markup language</td>
</tr>
</tbody>
</table>
## Contents

Abstract  
Tiivistelmä  
Preface  
Abbreviations  
Contents  

### 1 Introduction

1.1 Background  
1.2 Motivation  
1.3 The contributions of the thesis  
1.4 The outline of the thesis  

### 2 Vision-based proactive systems

2.1 Applications  
2.1.1 Intelligent environments  
2.1.2 Visual surveillance  
2.1.3 Other use cases  
2.2 General structure and basic components  
2.3 Methods  
2.3.1 Object detection  
2.3.2 Object tracking  
2.3.3 Action analysis  
2.4 Summary  

### 3 An automatic camera-based distance education system

3.1 Distance education assistant  
3.1.1 Introduction  
3.1.2 System overview  
3.1.3 User interface  
3.1.4 Image analysis  
3.1.5 Video source selection  
3.1.6 Experimental results  
3.2 Active PTZ tracking  
3.2.1 Introduction
3.2.2 Feature selection and tracking .............................................. 57
3.2.3 Pan-tilt-zoom control ......................................................... 60
3.2.4 Experimental results ........................................................ 62
3.3 Discussion ............................................................................ 64

4 Automatic landscape shooting mode selection for mobile devices 67
4.1 Introduction ........................................................................... 67
4.2 Methods for landscape scene recognition ................................. 69
  4.2.1 Global features .................................................................. 69
  4.2.2 Local features .................................................................... 72
  4.2.3 Classification ..................................................................... 73
4.3 Experimental results .............................................................. 73
  4.3.1 Image sets .......................................................................... 73
  4.3.2 Evaluation criteria .............................................................. 74
  4.3.3 Results .............................................................................. 74
  4.3.4 Computational cost ............................................................ 76
  4.3.5 Real-time implementation ................................................ 78
4.4 Discussion ............................................................................. 79

5 An open framework for distributed sensor networks 81
5.1 Introduction ............................................................................ 81
5.2 Framework description .......................................................... 84
  5.2.1 Architecture and interfaces ................................................ 85
  5.2.2 Example usage scenarios .................................................. 89
5.3 Experimental system .............................................................. 91
  5.3.1 Current interface implementation ........................................ 91
  5.3.2 Demonstration system ........................................................ 92
  5.3.3 Experimental results .......................................................... 94
5.4 Discussion ............................................................................. 95

6 Multi-object tracking ................................................................. 97
6.1 Introduction ............................................................................ 97
6.2 Tracking algorithm ................................................................. 98
  6.2.1 System model ................................................................. 99
  6.2.2 Measurement model ......................................................... 99
  6.2.3 Soft assignment ............................................................... 99
  6.2.4 Implementation details ...................................................... 100
6.3 Obtaining measurements from binary masks ........................................ 102
   6.3.1 Implementation based on color features .......................... 102
   6.3.2 Experimental results ................................................. 104
6.4 Using detector responses as measurements ...................................... 105
   6.4.1 Object detection ......................................................... 106
   6.4.2 Object initialization .................................................. 107
   6.4.3 Object scale ............................................................. 107
   6.4.4 Track termination ...................................................... 108
   6.4.5 Occlusion handling .................................................... 108
   6.4.6 Experimental results .................................................. 108
6.5 Discussion .............................................................................. 109

7 Conclusions .............................................................................. 113

References ................................................................................... 117
1 Introduction

1.1 Background

Human-computer interaction (HCI) plays an important role in modern society. A traditional case of HCI is a single user facing a computer and interacting with it via a mouse or a keyboard (Dix et al. 2003). Due to the ever increasing number of technical devices around us, the way of interacting is changing. Instead of traditional desktop computers, a system typically consists of a number of small embedded devices that are placed into the environment. In the emerging applications, interactions are not always explicit commands, and often involve multiple users (Jaimes & Sebe 2007). As Sebe et al. (2004) state, to truly achieve effective human-computer interaction, there is a need for the computer to be able to interact naturally with the user, similar to the way human-human interaction takes place. This will inevitably lead to a paradigm shift in human-computer interaction from traditional computer-centered to human-centered systems where a user is not interested in the underlying technology and might be even unaware of it. The ultimate goal is that there is no need to learn to use different kinds of interfaces (Harper et al. 2008).

As a remedy, proactive computing is an area of research in which the goal is to develop systems that can adapt and adjust to people’s movements and actions without requiring any conscious control (Tennenhouse 2000). The nature of proactivity is well captured by the Oxford English Dictionary (Oxford University Press 2000) definition of proactive:

“proactive = (of a person, policy, etc.) creating or controlling a situation by taking the initiative or anticipating events; ready to take initiative, tending to make things happen. – ”

That being said, the word proactive can be regarded as an antonym of reactive. As opposed to the reactive computing which responds to your explicit commands by executing the tasks that you specify, proactive computing is carried out continuously on your behalf. Both proactive and reactive approaches have advantages. In a nutshell, a reactive system gives you exactly what you want when you ask for it, whereas a proactive system guesses what you want done for you and does it. Maybe that is why Borriello (2006) notes that there should always be a user interface that is easily seen and
allows the user to check that the system is operating as expected or that can override what the system is doing proactively.

There are many other terms related to proactivity. For example, the terms human computing (Pantic et al. 2009), Ambient Intelligence (AmI) (Augusto et al. 2010, Cook et al. 2009, Remagnino & Foresti 2009), Smart Spaces (Okoshi et al. 2001, Stanford et al. 2003), ubiquitous computing (Weiser 1991, 1993), and pervasive computing (Saha & Mukherjee 2003) share some common ideas. It is difficult to distinguish these concepts from each other, especially if even the creators use them interchangeably. Ronzani (2009) enlightens us on how these concepts have evolved through and in the mass media.

From the point of view of this thesis, AmI seems to be the most closely related term. As Cook et al. (2009) explain, the basic idea behind AmI is as follows. By enriching an environment with technology, for example, sensors and devices interconnected through a network, a system can be built such that it acts as an “electronic butler”. It senses features of the users and their environment, then reasons about the accumulated data, and finally selects actions to take that will benefit the users in the environment. Augusto et al. (2010) also remind that it may not be of interest to a user what kind of sensors are embedded in the environment. Only the services given to the user matter to them. Therefore, the main thrust of research in AmI should be integration of existing technologies rather than development of each elemental device.

The capability to understand human actions is an essential part of many proactive systems. For instance, knowledge on whether a person is sitting, standing or walking is likely to matter, as well as how s/he is using his/her hands, or which way s/he is talking. In addition, observing how humans use and interact with the devices at their disposal may give valuable information. Therefore, it can be said that context (Dourish 2004, Baldauf et al. 2007) is also a valuable concept. In general, context-awareness refers to the capability of an application, service or even an artifact to be aware of its physical environment or situation and respond proactively and intelligently based on such awareness (Baker et al. 2009).

The user’s actions in conjunction with the surrounding environment can be regarded as an implicit input to the system. Based on these kinds of inputs, a proactive system may adapt to the activities and provide the desired responses to relevant events. As an example of modern interaction, an intelligent room can recognize the presence of a person, and respond or react with it; for instance, regulating the temperature or alerting about safety and security, following orders of emergency, urgency or importance and
preferences previously set by the user or learned by the system itself (Remagnino & Foresti 2009).

Visual information plays a vital role in implicit human-computer interaction due to its expressiveness and unobtrusiveness. It is based on non-contact, passive sensing technology, and a large area can be covered by a single sensor. But since raw pixel data in itself is not informative for a system, there is a need for deeper analysis of what actually exist in the images. Computer vision research aims to study how to extract information from an image that is necessary to solve some task (Forsyth & Ponce 2003). With modern computer vision methods, proactive systems can, to mention but a few attributes, sense the movements of objects, detect and track people or other objects, personalize services by recognizing people, recognize gestures, text and other objects or events. It is therefore obvious that cameras equipped with computing power and computer vision techniques provide a unique capability for understanding human intentions. The simultaneous use of several camera based sensor modules in the same environment to support an application gives an even broader range of possibilities.

1.2 Motivation

Computer vision will play an important role in proactive systems of the future. But despite its many advantages, use of computer vision is not always straightforward (Turk 2004). Typically every application sets specific requirements on the methods that can be applied. There are also many different kinds of environments, which makes it hard to develop totally generic computer vision algorithms. Also selecting the most suitable approaches to be used in each proactive computer vision solution requires thorough testing and studying. As a case study, this thesis covers two different vision-based proactive applications. It should be noted that both of the cases are originally introduced by companies who have had a great interest into solving some real-world problems.

One potential application area of camera-based proactive technology is clearly video conferencing. Since a typical video conferencing setup already consists of several cameras, the use of computer vision methods is feasible. Furthermore, distance education can be regarded as a special case of video conferencing. When giving a lecture to remote participants, a teacher needs to constantly select the sources of the video feed between different devices. From the teacher’s point of view, this is annoying, and in practice these actions may easily be forgotten. It is obvious that the usability of the system could be improved significantly by changing the sources automatically based
on the teacher’s actions. Hence, Chapter 3 concentrates on automatic selection and switching of cameras in a distance education environment.

The second application operates on a totally different domain and scale compared to the previous one. Most digital cameras and mobile devices equipped with a camera nowadays support a number of different shooting modes for use in various situations. Typically, a user has to select manually the most appropriate mode, which can be inconvenient. Automatic scene mode selection on mobile devices can be seen as an interesting opportunity and application for proactive computer vision. Because the sensor needed for the analysis already exists, it is reasonable to apply computer vision methods when automating the shooting process. Therefore, Chapter 4 studies different methods that can be used in automatic shooting mode selection on mobile devices.

Distributed smart cameras and sensors have been an active area of research in recent years (Rinner & Wolf 2008). Most of the research has focused on either the computer vision algorithms or on a specific implementation. There has been less activity on building generic frameworks which allow different algorithms, sensors and distribution methods to be used. Given these motivations, this thesis presents an open and expendable framework for development of distributed sensor networks with an emphasis on peer-to-peer networking (Chapter 5). The user is provided with easy access to sensors and communication channels between distributed nodes, allowing the effort to be focused on the development of computer vision algorithms and their use in distributed environments.

From the methodological point of view, many modern visual surveillance systems and intelligent environments need to track multiple objects at the same time. Due to this need in proactive computer vision, this thesis makes it contribution to the field of multi-object tracking. The starting point for the work has been a recently introduced novel method of tracking two distinct motions (Hannuksela et al. 2007). In this thesis, the aforementioned method, which utilizes soft assignment to associate the measurements to the objects tracked, is further extended and applied to multi-object tracking. Chapter 6 also presents two different ways of extracting location measurements from images. As a result, the method proposed provides location and trajectories of multiple objects which can be utilized in proactive applications.
1.3 The contributions of the thesis

This thesis aims to provide new methods and systems for vision-based proactive applications, and it contains a contribution in all of its individual parts and as a whole. In summary, the different aspects of proactive computer vision, covering the range from algorithms to entire systems, are discussed.

Publications (Huttunen et al. 2005, Huttunen & Heikkilä 2006, Hannuksela et al. 2007, Huttunen et al. 2008, Saastamoinen et al. 2008, Huttunen & Heikkilä 2008, Huttunen & Heikkilä 2010, Huttunen et al. 2011) reflect the author’s contribution to the research field. The work has been mainly performed by the author under the supervision of Prof. Janne Heikkilä, who has provided the guidelines for the work.

In the following list, the main contributions of the thesis are summarized:

– A review of the vision-based proactive applications is made.
– An automatic distance education system with an additional PTZ tracking module is presented. In addition, a new method for analyzing document camera usage is proposed. This application has also been implemented as a commercial product by a local company.
– A real-time capable automatic landscape shooting mode selection on mobile platforms is introduced.
– An open framework for distributed sensor networks is proposed and made publicly available.
– A new method for multi-object tracking based on soft assignment of measurements is proposed. The method includes a novel way of extracting measurements from binary masks as well as using detector responses as observations.

1.4 The outline of the thesis

The remainder of the thesis is organized into six chapters. In the following, the content of each chapter is described shortly.

Chapter 2 provides an introduction to vision-based proactive systems. The general structure and basic components of a proactive system are presented, and a literature review of the typical computer vision methods and the main applications is made.

Chapter 3 presents an automated system that takes care of both the selection and switching of the video source in a distance education situation. The camera views from the classroom are used to observe the teacher’s movements and any possible
document camera usage. The system serves as a test-case for proactive computer vision. Section 3.2 extends the automated lecture room system with a pan-tilt-zoom (PTZ) camera system that is designed to track the teacher when s/he walks at the front of the classroom. Tracking is carried out based on the information provided by the pan-tilt-zoom camera itself.

Chapter 4 gives a description of a proactive application targeted at mobile devices. The system presented recognizes landscape scenes which can be utilized in automatic shooting mode selection. First, the chapter studies different approaches that can be used in recognizing landscape scenes. Later, the results serve as a base for the development of real-time automatic landscape mode detection.

Chapter 5 introduces a framework which aims to provide the user with simple and configurable interfaces for accessing sensors and communicating with other nodes. A generic framework which includes these essential parts allows the users to focus on their area of expertise and allow rapid prototyping and development of proactive applications.

Chapter 6 proposes a new multi-object tracking method. The method utilizes soft assignment to associate the measurements to the objects tracked. Due to soft assignment, it is able to cope with inaccurate observations and inter-object occlusions. Sections 6.3 and 6.4 also offer descriptions of two different approaches to extract measurements from images.

Finally, Chapter 7 summarizes the main findings, discusses future work and concludes the thesis.
2 Vision-based proactive systems

This chapter discusses the topics related to vision-based proactive systems. A literature review gives an overview of the main applications and the typical computer vision methods used. From the architectural point of view, the basic components of a typical proactive computer vision system are described.

2.1 Applications

In this section, different application areas are discussed. In all the applications discussed below, a non-intrusive sensory method based on vision is preferable over a method that relies on devices attached physically to the bodies of the human subjects.

2.1.1 Intelligent environments

According to Remagnino & Foresti (2009), an intelligent room can recognize the presence of a person and respond to or react with it; for instance regulating the temperature or alerting about safety and security, following orders of emergency, urgency or importance and preferences previously set by the user or learnt by the system itself. The EasyLiving project (Brumitt et al. 2000) is concerned with the development of an architecture and technologies for intelligent environments. The components of such a system include middleware to facilitate distributed computing, world modeling to provide a location-based context, perception to collect information about world state, and service description to support decomposition of the device control, internal logic, and user interface. Stereo computer vision is used as a way of tracking the location and identity in the example scenario presented by Brumitt et al. (2000). However, they also want to remind us, that even if vision has unique advantages over other sensors for tracking people, it also presents unique challenges. A person’s appearance in an image varies significantly due to posture, facing direction, distance from the camera, and occlusions. It can be particularly difficult to keep track of several people in a room as they move around and occlude each other. Although a variety of algorithms can overcome these difficulties, the final solution must also work fast enough to make the system responsive to the room’s occupants.
One important application for computer vision is a supportive home environment where the cameras can be used to monitor unusual activity. Since wear detectors can often be uncomfortable, clearly the supportive home is a special case of proactive systems where the cameras can offer an unobtrusive way to monitor people at home. In the case of elderly people living on their own, there is a particular need for monitoring their behavior, such as a fall, unusual squatting, or a long period of inactivity (Lin et al. 2006). And from the elderly’s point of view, intelligent home environments can help extend independent, quality living. Chan et al. (2008) review an international selection of leading smart home projects related to the topic. The authors remind us that further research is needed into legal and ethical problems, user and provider acceptance, and user and provider requirements and satisfaction.

Fleck & Strasser (2008) present an application of smart camera technology to elderly care. They have implemented a prototype of a surveillance system consisting of smart cameras capable of tracking persons and detecting persons who have fallen. Privacy preservation is a special concern of this prototype. Another example of a supportive home environment is given by Nait-Charif & McKenna (2004). The method presented in their paper enables the detection of inactivity outside usual zones of inactivity, for example chairs and beds. It is hypothesized that when combined with body pose and motion information it should provide a useful cue for fall detection. In addition, a human-readable description of activity in terms of semantic regions gives a useful summary of behavior.

In addition to intelligent and supportive home systems, proactive technology can be applied to lecture room environments. In McGill’s Intelligent Classroom (Cooperstock 2001), for example, lecturers need not worry about things such as light control or AV device configuration. In the Smart Classroom (Shi et al. 2003, Li et al. 2009), the teacher no longer needs to remain stationary in front of a desktop computer, and for most of the common tasks involved in a class, the teacher no longer needs to use a keyboard and a mouse either. A more detailed review of automated lecture room environments is given in Chapter 3, where we describe our own automated lecture room system.

### 2.1.2 Visual surveillance

In the last decade, visual surveillance in dynamic scenes, especially for humans and vehicles, has been one of the most active research topics in computer vision (Haritaoglu et al. 2000, Hu et al. 2004, Foresti et al. 2005, Plataniotis & Regazzoni 2005, Valera
According to Hu et al. (2004), visual surveillance has a wide spectrum of promising applications, including access control in special areas, human identification at a distance, crowd flux statistics and congestion analysis, detection of anomalous behaviors, and interactive surveillance using multiple cameras. Similarly, Valera & Velastin (2005) state that the demand for remote surveillance relative to safety and security has received significant attention, especially in public places, remote surveillance of human activities, surveillance in forensic applications, and remote surveillance in military applications. The public can be perceived either as individuals or as a crowd. In addition, Valera & Velastin (2005) indicate that a future challenge is to develop a wide-area distributed multisensor surveillance system, which has robust, real-time computer algorithms, which are executable with minimal manual reconfiguration for different applications.

From the historical perspective, one of the pioneers of the research field is the W4 system (Haritaoglu et al. 2000), which is a real-time visual surveillance system for detecting and tracking several people and monitoring their activities in an outdoor environment. It operates on monocular gray-scale video imagery, or on video imagery from an infrared camera. W4 employs a combination of shape analysis and tracking to locate people and their parts (head, hands, feet, torso) and to create models of people’s appearance so that they can be tracked through interactions such as occlusions. It can determine whether a foreground region contains several people and can segment the region into its constituent people and track them. W4 can also determine whether people are carrying objects, and can segment objects from their silhouettes, and construct appearance models for them so they can be identified in subsequent frames. W4 can recognize events between people and objects, such as depositing an object, exchanging bags, or removing an object.

Another famous system related to the topic is Pfinder (Wren et al. 1997). Pfinder is a real-time system for tracking people and interpreting their behavior. It is reported that the system provides interactive performance on general-purpose hardware, has been tested on thousands of people in several installations around the world, and has performed quite reliably.

Räty (2010) reviews concisely the historical development and current state of the three different generations of contemporary surveillance systems. The author states that recently, in addition to the employment of the incessantly growing variety of sensors, the inclination has been to utilize more intelligence and situation awareness capabilities to assist the human surveillance personnel.
As both Hampapur (2008) and Javed et al. (2008) state, traditional surveillance systems leave the entire burden of watching video, detecting threats, and locating suspects to the human operator. This process of manually watching video is known to be tedious, ineffective, and expensive. Intelligent surveillance systems, which are now "watching the video" and providing alerts and content-based search capabilities, make the video monitoring and investigation process scalable and effective. The software algorithms that analyze the video and provide alerts are commonly referred to as video analytics. These are responsible for transforming video cameras from being mere data gathering tools into smart surveillance systems for proactive security. Hampapur (2008) emphasizes that the capability to search through large amounts video data, correlate events across cameras, and correlate video events to other information becomes the basis for moving security from a reactive paradigm, to a rapid investigation paradigm and eventually a proactive paradigm.

As a consequence, a great deal of financial resources have been devoted to research aiming at designing automated surveillance systems that are able to help human operators in detecting interesting situations. For example, the IBM smart surveillance system (S3) is an advanced surveillance system which provides not only the capability to automatically monitor a scene but also the capability to manage the surveillance data, perform event based retrieval, receive real time event alerts through standard web infrastructure and extract long term statistical patterns of activity (Hampapur 2008, Tian et al. 2008). However, Dore et al. (2011) remind us that most of the systems are generally able to report unusual or potentially dangerous events, but not prevent or cope with them. Cognitive surveillance systems (Dore et al. 2011) are intended to overcome this limitation by interacting directly with the environment through actuators present in the monitored scene. Actions performed by the system must provoke a modification of the external world in order to either avoid undesired events taking place or to handle them directly and quickly. In this setting, communication devices or door and window locks, and speakers, are the actuators which directly affect the environment and the subjects. All that being said, visual surveillance can be seen as one of the most interesting and challenging applications of proactive computer vision.

### 2.1.3 Other use cases

An additional key area of robotics research is concerned with developing humanoid robots for assisting humans in everyday tasks (Molina-Tanco et al. 2005, Chan et al.
Mobile robots equipped with cameras and other sensors provide an excellent environment for investigating proactive solutions, because communication with social robots developed for service tasks in homes, nursing homes and retirement homes, for example, must be easy and natural. An intelligent robot will detect and identify the user in order to personalize and customize its services and guarantee security.

Mobile devices are one emerging application area for proactive computer vision due to their user interface limitations. Compared to the applications mentioned above, the contextual information that a mobile device can use is much richer than that of non-mobile settings. On the other hand, people on the move need information relevant to their location and immediate needs. A context for a mobile application includes, for example, time, location, interaction history, a user’s schedule, and so on (Hong et al. 2009). Since most hand-held devices are nowadays equipped with a camera, visual information can be used to give valuable information about the environment in which a user is located, or even users’ activities. However, it seems that there are not many camera-based approaches. Probably the single biggest reason is that a camera consumes power which is critical on mobile devices. For this reason, the current applications are typically meant to assist the image capturing process. As an example, Bourke et al. (2011) describe a novel recommendation process which uses a variety of intelligent and assistive interfaces to guide the user in taking relevant compositions given their current location and scene context. At this point, it is worth noting that the application presented later in this thesis falls into the same category because it is targeted at automatic shooting mode selection.

A final example of proactive computer vision applications aims at improving traffic safety. Advanced driver assistance systems, and particularly pedestrian protection systems (PPSs), have become an active research area of computer vision. Gerónimo et al. (2010) explain that the objective of a PPS is to detect the presence of both stationary and moving people in a specific area of interest around the moving host vehicle in order to warn the driver, perform braking actions, and deploy external airbags if a collision is unavoidable. Trivedi et al. (2007) present investigations into the role of computer-vision technology in developing safer automobiles. They consider vision systems which cannot only look out of the vehicle to detect and track roads and avoid hitting obstacles or pedestrians, but simultaneously look inside the vehicle to monitor the attentiveness of the driver and even predict his/her intentions.
2.2 General structure and basic components

Based on the contents of previous section, the definition of a typical proactive computer vision system is very hard or even impossible. Different applications set their own physical as well as computational requirements. The number of cameras involved in the system may vary from one camera in a mobile device to tens of cameras for monitoring a room or building, and all the way to hundreds of cameras connected in a wide area visual surveillance systems.

Having said that, it is still possible to identify some basic building blocks and principles that all the proactive computer vision systems share. From the physical environment’s point of view, there must be at least sensors and actuators by which the system interacts with the world (Tennenhouse 2000). Later Crowley (2006) has introduced two concepts, perception and action, that operate at a higher level of abstraction than sensors and actuators. While sensors and actuators operate on device-specific signals, perception and action operate in terms of environmental state. Perception interprets sensor signals by recognizing and observing entities. Abstract tasks are expressed in terms of a desired result rather than actions to be blindly executed.

Starting from the sensor level, every system has a camera that can capture images from the surroundings. After that, the acquired image data is analyzed and based on the analysis, the appropriate actions by the actuators take place. As an example, a single camera based sensor can be used simply as a door controller in order to keep the door open as long as there are people walking through it. In summary, a system has to have at least one camera, a processing unit for image analysis, and finally a connection to actuators. The communication between different parts of the proactive system can be accomplished in many different ways. If the parts are physically occupied in a single device, it is possible to use software level commands and interfaces. Otherwise, a wired or wireless channel for communication has to be established.

Following the previous definitions, Pietikäinen et al. (2004) proposed a general framework for proactive computer vision. They divide a proactive system into the following levels: proactive, system, module, and skill. The individual vision modules are able to analyze acquired image data autonomously and communicate with actuators, other sensors or vision modules using simple control signals and messages. According to Pietikäinen et al. (2004), the modules can be identical, but they may as well differ by their skills. At skill level each sensor module implements image analysis methods for tasks such as background subtraction and face detection. At sensor module level, each
module manages communications and controls camera parameters such as pan, tilt, color and geometric calibration. System-wise each module receives the time synchronized analysis results and sample images from the other modules in the same application system and, finally, at the proactive level, responses are generated when certain events occur. Pietikäinen et al. (2004) report that their technological vision is augmenting the environment with low-cost camera based sensor modules that alone are capable of basic scene and event analysis, and together can rapidly learn to recognize more complex situations and provide desired responses.

The simultaneous use of several camera based sensor modules in the same environment to support a proactive application is an interesting research issue. Using multiple cameras in the network provides different views of the scene, which enhances the reliability of the captured events. Currently, the aim seems to be to use autonomous sensor modules leading towards fusion at a higher level (Rinner & Wolf 2008, Soro & Heinzelman 2009). However, the large amount of image data produced by the cameras, combined with the network’s resource constraints require exploration of new means for data processing, communication, and sensor management. Soro & Heinzelman (2009) emphasize that meeting these challenges of visual sensor networks (VSNs) requires interdisciplinary approaches, utilizing vision processing, communications and networking, and embedded processing. In their paper, the authors provide an overview of the current state-of-the-art in the field of VSNs.

As a concrete example, distributed smart cameras (DSCs) are real-time distributed embedded systems that perform computer vision using multiple cameras (Rinner & Wolf 2008). Smart cameras combine video sensing, processing, and communication on a single embedded platform and they deliver some abstracted data of the observed scene. Rinner & Wolf (2008) noted that a smart camera that is part of a network performs huge amounts of computation in order to abstract the raw image data and reduce the bandwidth required to transmit the data. Smart cameras perform a variety of image-processing algorithms such as motion detection, segmentation, tracking, object recognition, and so on. Most importantly, they typically deliver color and geometric features, segmented objects, or rather high-level decisions.

### 2.3 Methods

Vision-based proactive systems rely on different computer vision techniques. The purpose of the following subsection is to provide information about the basic techniques
for object detection and tracking, and finally action analysis. Generally speaking, object
detection algorithms typically determine which parts of the scene correspond to moving
objects, and tracking algorithms associate the movement of objects over time, generating
a trajectory. These two algorithms together take a video stream and decompose it into
objects and events, effectively creating a basis for further action recognition. At the
same time, object detection can also be utilized in recognizing different items people
interact with and interpreting the contents of the scene.

Concerning the topic of improving interaction, it appears that most of the methods
involve observing people directly. Hence, according to Pentland (2000), the research
topic of looking at people, that is, giving machines the ability to detect, track, and
identify people, and more generally, to interpret human behavior, has been a central topic
in computer vision research for years. Already at the beginning of this century, Pentland
(2000) centered on person identification, surveillance/monitoring, 3-D methods, and
smart rooms/perceptual user interfaces to review the state-of-the-art of ”looking at
people”. The paper was not intended to survey the current work on human motion
analysis, but touches on several interesting topics in human motion analysis and its
applications.

2.3.1 Object detection

Nearly every vision-based analysis starts with separating interesting objects from the
rest of the image. For example, vision-based human motion analysis starts with human
detection which aims to segment regions corresponding to people. It is a significant
issue especially in human motion analysis systems since the subsequent processes, such
as tracking and action recognition, are greatly dependent on it.

Background subtraction

Object detection can be achieved by building a representation of the scene, called the
background model, and then finding deviations from the model for each incoming
frame by taking the difference between the frames in a pixel-by-pixel fashion. Any
significant change in an image region from the background model signifies a moving
object. The pixels constituting the regions undergoing change are marked for further
processing. Usually, a connected component algorithm is applied to obtain connected
regions corresponding to the objects. This process is referred to as the background
subtraction, which is a popular method for motion segmentation, especially in those situations with a relatively static background.

Frame differencing of temporally adjacent frames has been a widely used background subtraction technique over the years. It makes use of the pixel-wise differences between two or three consecutive frames in an image sequence to extract moving regions. This kind of technique is very adaptive to dynamic environments, but generally does a poor job of extracting all the relevant pixels; for example, there may be holes left inside moving entities. As an example of this method, in (Zhang et al. 2005) the speaker is detected in a lecture room environment using simple frame differencing.

A large number of different methods for detecting moving objects have been proposed and many different features are utilized for modeling the background. Most of the methods use only the pixel color or intensity information to make the decision, but Heikkilä & Pietikäinen (2006) propose an approach that uses discriminative texture features to capture background statistics. Kim et al. (2005) present an algorithm in which sample background values at each pixel are quantized into codebooks. These codebooks represent a compressed form of background model for a long image sequence, and the authors claim that their approach allows the capture of structural background variation due to periodic-like motion over a long period of time under limited memory.

Background subtraction is simple, but extremely sensitive to changes in dynamic scenes derived from lighting and extraneous events. Therefore, it is highly dependent on a good background model to reduce the influence of these changes as part of environment modeling. Related to this problem, Woo et al. (2010) have lately proposed a motion detection model based upon variational energy, which provides a robust detection method at various illumination changes and noise levels of image sequences without tuning any parameter manually. Another limitation is that motion can also be caused by some other moving objects, which means that further analysis is needed to identify the object. On the contrary, it is difficult, for example, to detect humans standing still.

Static detectors

The second approach investigated, object detection based on machine learning, does not involve the problems with motion detection. The basic idea is that a classifier is trained to decide whether an image bounding box contains an object or not. This classifier is then slid over the entire image to detect the objects at different positions of the image. The training phase requires a data set that contains thousands of positive
samples representing the object of interest in different environments and also negative samples where the object is not present.

There exists a wide variety of detection methods. At one end of the spectrum are part-based detectors (Mikolajczyk et al. 2004, Wu & Nevatia 2005), which represent an object as an assembly of distinct parts. At the other end of the spectrum are detection methods (Gavrila 2000, Viola & Jones 2001) that try to find a specific object as a whole. Furthermore, it is stated in the work of Yang et al. (2002) that images containing faces are essential to intelligent vision-based human-computer interaction, and research efforts in face processing include face recognition, face tracking, pose estimation and expression recognition. To build fully automated systems that analyze the information contained in face images, robust and efficient face detection algorithms are required. The object detector initially proposed by Viola & Jones (2001) and improved by Lienhart & Maydt (2002) is currently widely used for this task. The actual implementation of the detector described above is part of the Intel Open Computer Vision Library (OpenCV) (Bradski & Kaehler 2008, Willow Garage 2011). This is probably one of the reasons behind the success of this detector.

For a comprehensive survey of different human detection methods, one of the best sources of information is the work by Dalal (2006). Also Enzweiler & Gavrila (2009) consider a diverse set of state-of-the-art systems. There are basically many alternatives for the classifier, including Support Vector Machines (SVM) (Vapnik 1998), boosting (AdaBoost) (Freund & Schapire 1996), and artificial neural networks. It seems that the selection of the classifier does not play an important role when considering only the accuracy. From the application point of view, however, there are considerable differences in the computational complexities of the classification methods. Among the most efficient human detection methods published in the literature is the HOG-LBP detector proposed by Wang et al. (2009) that uses a special combination of Histograms of Oriented Gradients and Local Binary Pattern (LBP). This method is also capable of handling partial occlusions.

### 2.3.2 Object tracking

Many proactive applications have in common the need to track humans and their behavior. Computer vision algorithms that are intended to take care of this task must be fast and efficient. They must be able to track in real time and yet not absorb a major share of computational resources. In general, there is no single technique that is useful
for all applications. As Yilmaz et al. (2006) note, typically, assumptions are made to constrain the tracking problem in the context of a particular application.

In principle, tracking over time involves matching objects in consecutive frames using features such as points, lines or blobs. Moreover, this process of estimating over time the location of one or more objects using a camera is referred to as video tracking (Maggio & Cavallaro 2011) or visual tracking (Cannons 2008, Dore et al. 2010). In this thesis, the focus is put on camera based approaches and therefore the use of the single term tracking is unambiguous.

Tracking can be divided into various categories according to different criteria. As far as tracked objects are concerned, tracking may be classified into tracking of human body parts such as hand, face, and leg, and tracking of whole body. Yilmaz et al. (2006) categorize the tracking methods on the basis of the object and motion representations used, and provide detailed descriptions of representative methods in each category. Cannons (2008) divides the visual trackers in the literature into three tracking categories, namely discrete feature trackers, contour trackers, and region-based trackers. The first class of trackers represents targets as discrete features whereas contour trackers provide precise outlines of the target boundaries, which encapsulate the shape of the object as well. Region trackers represent the target with area-based descriptors that define its support and attempt to locate the image region in the current frame that best matches an object template (Cannons 2008).

Visual tracking is one of the fundamental tools necessary in the development of video surveillance and analytics applications. Dore et al. (2010) concentrate on general considerations on the design of visual trackers for video analytics systems, focusing on the trade-off that is usually performed between the accuracy of the target motion assumptions and the robustness of the object appearance representation.

Regardless of the tracking method, we can identify four main components of a typical tracker, which are also discussed in more detail by Maggio & Cavallaro (2011). At the first step, we have to locate and extract information about a target. This method can be based on the methods presented in the previous subsection. Due to increased computational power, it is nowadays possible to use static detectors for every frame instead of just some frames. Therefore, the tracking-by-detection approaches using detector responses directly as observations have become viable option (Breitenstein et al. 2010).

After the localization, an appropriate representation of a target should be selected in order to define the characteristics of the object. The main challenges that have to
be taken into account are related to the similarity of appearance between the target and other objects in the scene as well as to appearance variations of the target itself. Recently Kuo et al. (2010) have focused especially on resolving ambiguities between the different targets. They have presented an approach for online learning of discriminative appearance models for robust multi-target tracking in a crowded scene from a single camera.

At the next step, a method to propagate the state of the target over time is needed. This step links different instances of the same object between frames. Some of the most well-known methods are the mean shift algorithm (Comaniciu et al. 2003), the Kalman filter (Kalman 1960), and the particle filter (Arulampalam et al. 2002).

Finally, a strategy to handle targets appearing and disappearing from the scene is required. This step initializes the track for an incoming object of interest and terminates the trajectory linked to a target leaving the scene.

### 2.3.3 Action analysis

Action analysis has become a very important topic in computer vision, with many fundamental applications in robotics, video surveillance, human–computer interaction (HCI), and multimedia retrieval among others, and a large variety of approaches have been described (Weinland et al. 2010). According to Weinland et al. (2010), visual action analysis has been traditionally divided into sub-topics such as gesture recognition for HCI (Pavlovic et al. 1997, Kisacanin et al. 2005), facial expression recognition (Zhao et al. 2003), and movement behavior recognition for video surveillance (Hu et al. 2004).

### Action recognition

The use of hand gestures provides an attractive alternative to cumbersome interface devices for HCI. In particular, visual interpretation of hand gestures can help in achieving the ease and naturalness desired for HCI (Pavlovic et al. 1997, Kisacanin et al. 2005). This has motivated a very active research area concerned with computer vision-based analysis and interpretation of hand gestures. Bradski (1998) describes part of a larger program to develop a real time Perceptual User Interface. They want to give computers the ability to segment, track, and understand the pose, gestures, and emotional expressions of humans and the tools they might be using in front of a computer or settop.
box. Some results of research on hand tracking and gesture recognition over the years are given by Huang (2005) and Mitra & Acharya (2007). It is worth mentioning that in proactive applications we are especially interested in implicit inputs. Therefore, the methods and systems that are meant to replace the traditional input devices are not of great interest.

Considering the systems that utilize the whole body information instead of just hands, recognition of human action is divided into two main problems. First is the problem of getting whole body motion data. Second is the problem of interpretation of the human motion, which includes modeling of action, feature extraction, classification, and detection of novel action (Mori et al. 2004). Furthermore, according to Aggarwal & Park (2004), understanding human activities involves various steps of low-level vision processing such as segmentation, tracking, pose recovery, and trajectory estimation as well as high-level processing tasks, such as body modeling and representation of action. Even though high-level processing depends on the results of low-level processing, high-level processing also requires some independent and additional approaches and methodologies.

There are several existing surveys within the area of vision-based human motion analysis and recognition, and different taxonomies have been proposed (Poppe 2010). Bobick (1997) uses a taxonomy of movement recognition, activity recognition and action recognition. These three classes correspond roughly with low-level, mid-level and high-level vision tasks. Gavrila (1999) uses a taxonomy of 2D approaches, 3D approaches and recognition, whereas Wang et al. (2003) discuss body structure analysis, tracking and recognition.

An overview of various tasks involved in motion analysis of the human body prior to 1998 is provided by Aggarwal & Cai (1999). The paper gives an overview of the various tasks involved in motion analysis of the human body. The authors focus on three major areas which are motion analysis involving human body parts, tracking of human motion using single or multiple cameras, and recognizing human activities from image sequences, respectively. Later Aggarwal & Park (2004) put focus on the different aspects of high-level processing including human body modeling, the level of detail needed to understand human actions, approaches to human action recognition, and finally, high-level recognition schemes with domain knowledge. In their recent review, Aggarwal & Ryoo (2011) concentrate on high-level activity recognition methodologies designed for the analysis of human actions, interactions, and group activities, discussing recent research trends in activity recognition. The authors deal with both the methodologies
developed for simple human actions and those for high-level activities.

Moeslund et al. (2006) use a functional taxonomy with subsequent phases which are initialization, tracking, pose estimation and recognition. Within the recognition task, scene interpretation, holistic approaches, body-part approaches and action primitives are discussed. Turaga et al. (2008) focus on the higher-level recognition of human activity, and the problem is discussed at two major levels of complexity, namely, "actions" and "activities". A recent work by Ji & Liu (2010) provides a comprehensive survey of the research topic with the emphasis on view-invariant representation, and recognition of poses and actions.

From a methodological perspective, action analysis involved in proactive applications may be thought of as a time-varying data matching problem. One general analytical method for matching time-varying data is the Hidden Markov Model (HMM), which is applied to many systems. HMM consists of a finite set of hidden states, a set of observation states, probabilities of transitions between the states, and initial state probabilities. The success of HMM models in dealing with speech data motivated vision researchers to apply HMMs to visual recognition problems in the beginning of 1990’s (Yamato et al. 1992). A good tutorial for details of HMM is presented by Rabiner (1989).

As an example of action recognition in intelligent environments, we can be mention work by Ren & Xu (2002), which presents a framework for the teacher’s complex action recognition in the Smart Classroom (Shi et al. 2003). With the Hybrid Human Model, basic motion feature are extracted, which includes the two elbow angles and the motion features of the head and two hands. Primitive-based Coupled-HMMs are used for recognition. For the teacher in the smart classroom, there are totally seven kinds of natural actions to be recognized: taking objects from the desk, putting back objects, pointing to the students, pointing to the blackboard (virtual mouse), communication with the students, explaining objects, and drinking water.

Mori et al. (2004) present a recognition method of human daily-life action. The method utilizes a hierarchical structure of actions and describes it as a tree. The actions are modeled by using Continuous Hidden Markov Models which give an output of time-series feature vectors extracted by a feature extraction filter based on human knowledge.
Anomaly detection

Especially in proactive visual surveillance applications there is need to identify situations that require attention and immediate actions. In anomaly detection, the goal is to monitor the behavior of the persons and to recognize deviations from normal behavior patterns referred to as anomalous or abnormal events. Various approaches have been proposed in the literature for anomaly detection (Hu et al. 2004, Morris & Trivedi 2008). One popular approach utilized for example in (Zhang et al. 2009), is based on trajectory analysis, where models of the normal behavior are learned from the trajectories produced by target trackers.

Another approach is to avoid tracking and to use alternative motion representation like optical flow in (Adam et al. 2008). These approaches focus purely on motion information, ignoring abnormality information due to variations of object appearance. Recently, Mahadevan et al. (2010) have proposed an approach where both spatial and temporal abnormalities are recognized. In their work, spatial anomaly is defined using saliency detection, so that spatially abnormal locations are those whose saliency is above some threshold.

2.4 Summary

In summary, on the basis of what has been reported in this chapter, it is obvious that there do exist many applications and methods for computer vision in proactive systems. From the application point of view, visual surveillance and intelligent environments have gained most attention over the years. It is highly probable that especially the systems related to supportive home environments and elderly care will continue to evolve in the future.

When looking at the system level, different applications set their own physical, as well as computational requirements. The number of cameras involved in the system may vary from one camera in a mobile device to tens of cameras to monitor a room or building, and all the way to hundreds of cameras connected in wide area visual surveillance systems. Even though there are a number of different kind of settings, a system has to also have actuators that are needed to interact with the environment.

As far as the methods are concerned, based on the literature, most of the methods used in implicit interfaces involve observing people directly. Unfortunately it seems that the methods are often very task specific and seldom can be generalized over a
wide range of applications. For example, Aggarwal & Ryoo (2011) remind us that a significant amount of progress on human activity recognition has been made in the past ten years, but it is still far from being an off-the-shelf technology. Later in this thesis, the main interest in methods is placed on tracking of persons and their faces. In Chapter 3, the location of a teacher is determined using motion information. Furthermore, the active camera system presented in the same chapter detects and follows the teacher using a static object detector and color-based tracking. Chapter 6 focuses on the field of multi-object tracking and associating measurements to the objects. In a traditional sense, the two applications presented in this thesis do not strictly speaking rely on human action recognition methods reported earlier in this section. On the other hand, they show that observing humans directly is not always essential. For example, in the automated lecture room system (see Chapter 3), use of document camera analysis gives indirect, but valuable information. Similarly, in automatic landscape shooting mode selection (Chapter 4), we observe the user’s intentions using a camera device and its view itself.
3 An automatic camera-based distance education system

One potential application area of vision-based proactive technology is video conferencing. A typical video conferencing setup already consists of several cameras that can provide useful information. A special case of video conferencing is distance education, where the teacher needs to constantly switch the sources of the video feed between the computer, document camera, blackboard camera, overview camera, and possibly other devices. For the teacher, this is annoying, and in practice these actions may easily be forgotten, which might cause the effectiveness of teaching and learning to deteriorate. It is obvious that the usability of the system could be improved significantly by changing the sources automatically, based on the teacher’s actions.

This chapter introduces a proactive system targeted at the classroom environment. Section 3.1 describes in detail an automated system that takes care of both the selection and switching of the video source in a distance education situation (Huttunen et al. 2005, 2008). Using wide-angle cameras, it is easy to keep the speaker in the field of view (FOV) all the time. Unfortunately, this approach results in a low resolution image, where the details can be blurry. To cope with this problem, the camera has to be steered automatically. The second part of the implementation (Section 3.2) concentrates on developing a pan-tilt-zoom tracking system which follows the teacher actively when he/she moves in the classroom (Huttunen & Heikkilä 2006). Finally, the results are discussed in Section 3.3.

3.1 Distance education assistant

3.1.1 Introduction

When developing an automated system for video source switching, there is at least one basic requirement. The system should be reliable in its decisions, because erroneous decisions or constantly changing camera views are irritating from the audience’s point of view. In principle, the decisions should not differ substantially from the decisions made by a human operator. So the problem for constructing the automated system is twofold. The system needs reliable information as a basis for detecting and understanding the
FIG 1. Overview of the distance education assistant (DEA). (Modified with permission from Huttunen et al. (2008), © ACTA Press).

In order to have the necessary information, the system needs to extract such features from the video that could describe the possible events requiring some actions. However, the procedure for extracting the features should be computationally inexpensive to guarantee real-time performance.

Fig. 1 illustrates the concept of this Distance Education Assistant (DEA). The camera views from the classroom are used to observe the teacher’s movements and possible document camera usage. In this case, the stimulus for switching the source is obtained directly from the video cameras, and this offers an unobtrusive way of keeping watch on the person’s actions. Actual video source switching is carried out by the equipment and auditorium control unit installed in the classroom. Again it is not necessary to have any additional sensors apart from the cameras that already exist in a typical distance education classroom system. Before getting down to the actual implementation, a literature review of related lecture room systems is given in the following.

Several projects involve lecture room automation or automated camera management systems related to the topic. In McGill’s Intelligent Classroom (Cooperstock 2001), for example, lecturers need not worry about things such as light control or AV device configuration. Also the Intelligent Classroom developed at Northwestern University (Flachsbart et al. 2000, Franklin & Hammond 2001) observes the teacher’s actions using computer vision and speech recognition algorithms. In the implementation, different plan recognition technologies are utilized to find the lecture’s main focus.

Tsinghua University’s Smart Classroom (Shi et al. 2003) is one of the most advanced lecture room environments in the world. Computer-vision algorithms coordinate eight
video cameras that track the teacher’s movements, switching views as the person points to a page in a textbook or writes on the whiteboard. The computers recognize the positions of the teacher’s arms and zoom in on particular gestures. The system also tracks the trajectory of the laser pointer and responds to simple verbal commands.

Since there is a wide variety of systems with different kinds of architectures, hardware, and operation, reviewing them is not an easy task. However, Xu et al. (2009) give an extensive overview of the human-computer interaction technologies and approaches used in intelligent classroom systems. The authors categorize human-computer interaction in smart classroom systems into virtual assistant, pen, and film crew modules. In this thesis, the systems are divided into the two topics discussed in the following.

**Capture systems**

In addition to device control taking place in real-time, there are also other applications that are closely related to the field of lecture room automation. Multimedia and web-enhanced learning have become increasingly attractive to schools both for financial and technological reasons. Students spend a significant portion of the day listening to and recording the events that occur in classrooms, typically by taking notes with a pen and paper. As a result, the capture of classroom lectures for later access has become a popular research topic with several different approaches and contributions (Brotherton & Abowd 2004). The systems developed for automatic production of digital material share the same principles with the video conferencing application at hand. In both cases it is necessary to analyze the person’s actions based on the camera views and act appropriately.

Zhang & Rui (2008) survey existing approaches for providing automated lecture services. In particular, the authors examine two major challenges in providing such services, namely, how to capture, analyze and render the lecture content automatically, and how to provide a live/on-demand lecture viewing/browsing experience with an automated end-to-end system. In their other work, Zhang et al. (2008) describe a system architecture that minimizes the pre- and post-production time, and a fully automated lecture capture system called iCam2 that synchronously captures all contents of the lecture, including audio, video, and presentation material. No staff is needed during lecture capture and broadcasting, so the operational cost of the system is negligible.

Bellcore’s AutoAuditorium (Bianchi 1998, 2004) is said to be a pioneer in lecture...
room automation and it has been commercially available. It contains multiple cameras for capturing the lecturer and the screen. Based on heuristics, the so called director module selects which video is shown to the remote audience. However, no user study of AutoAuditorium is available. It is, therefore, hard to evaluate how well the system actually performs.

Microsoft Research has introduced an automated camera management system (Rui et al. 2001). The system performs lecturer tracking, audience tracking, and video editing, all fully automatic. A portable solution of the same system (Wallick et al. 2004) can be used in various types of lecture rooms.

One way to automate the video capturing process of a lecture is to use a robotic camera that tracks the movements of a lecturer during the delivery of a traditional class (Cavallaro et al. 2007). The robotic camera is guided by the results of an image processing module based on face detection. The video of the lecturer is synchronized with the presentation slides and with the audio of the lecture.

Nagai (2009) use an AVCHD camcorder and microserver for automated recording. The camcorder makes high-definition video capture boards unnecessary since it records videos as files on its file system, and capturing now becomes an easy task of copying files. With this idea, they can make the system small and cost effective. Lectures are automatically recorded according to the schedule specified by iCalendar data, and recorded videos are automatically processed with a camerawork engine that the authors have developed to generate NTSC resolution videos.

Camera selection

According to Lampi et al. (2007), a major problem with traditional lecture recordings is that they tend to be boring for the students, especially if only the slides and the audio of the lecturer are presented. In order to increase liveliness of the video, the video feed transmitted should be switched from time to time. The problem is then to determine when to change the video source. Therefore selecting the most suitable camera can be regarded as an important ingredient in the automated system presented in this chapter. Hence we briefly review some of the works concentrating on camera selection before going into the actual implementation.

Rui et al. (2004) report on the design of a complete system that captures and broadcasts lectures automatically and they report on a user study and a detailed set of video-production rules obtained from professional videographers who critiqued the
Onishi & Fukunaga (2004) propose computer-controlled camera work that shoots object scenes to model the professional cameramen’s work, and selects the best image among plural video images as a switcher. This system is applied to the shooting of a lecture scene. In the first image, the system estimates a teacher’s action based on the features of the teacher and a blackboard. Next, each camera is automatically directed to a shooting area based on the teacher’s action.

The approach used in FlySPEC (Liu et al. 2002) is quite interesting and exceptional. The camera control system integrates requests from multiple users so that each controls a virtual camera. In other words, the FlySPEC system seamlessly integrates manual and fully automatic control. It can also learn control strategies from user requests. However, it is worth noting that control of the camera may easily become a burden to students and distract them in their learning experience.

Heck et al. (2007) offer a new way to create lecture videos that retains many of the advantages of well-composed recordings, without the cost and intrusion of a video production crew. The authors present an automated system called Virtual Videography that employs the art of videography to mimic videographer-produced videos, while unobtrusively recording lectures.

### 3.1.2 System overview

In this subsection, an overview of the system used for the present study is given. Both the hardware and software components are described.

**Hardware**

The automated system presented in this section is deployed in a lecture room, as shown in Fig. 2. The lecture room is equipped with videoconferencing and AV-systems. The system developed uses Videra Inc.’s Auditorium Control Unit (ACU) (Videra 2009) which controls the equipment installed in the classroom. The classroom also includes a touch sensitive LCD screen for operating the ACU manually.

Currently, the system includes three different cameras, which are connected to the AV-matrix, as Fig. 3 illustrates. From the AV-matrix, video feeds of the overview and document cameras are transmitted to a PC which is equipped with a multichannel video capture card. The DEA, which is installed into the PC, chooses the video source to be
transmitted to remote participants on the basis of the image analysis results. After the video source selection has been made, the DEA sends a command to the ACU which handles actual video source switching by controlling the AV-matrix.

Usually a person giving a lecture wants to show some Microsoft PowerPoint slides or other material which is in a digital form. Presenting such material can be done with a desktop PC or laptop. However, converting the VGA signal to a PAL/NTSC signal and transmitting it through a videoconferencing connection is not the best possible solution. In order to achieve better quality, another way is needed for delivering the material between different participants. One possibility is to use videoconferencing codecs which support the H.239 standard (ITU 2005). This is the preferred method of delivering content because it provides a usually high quality result. Since these codecs are not yet widely spread, a traditional application sharing server is still available to make sure that every participant can join the conference without difficulties. In the event of the teacher wanting to use electronic material or other video sources like DVD or VHS instead, s/he has to select the desired video source manually using the touch screen mentioned above.
Fig 3. Hardware components of the system. (Modified with permission from Huttunen et al. (2008), ©ACTA Press).

Software

The software components described in this subsection are located and executed in a regular PC. The software components of the DEA are shown on the left in Fig. 4. In total there are four basic components, and all of them have their own special task. Also the software architecture of the ACU is described in Fig. 4.

The user interface (UI) can be used to visualize the operation of the system and to configure the properties of the different components. It also takes care of the synchronization between distinct threads required for image processing. Section 3.1.3 provides more information about the UI and its usage purposes.

The image analysis component consists of two subcomponents: teacher localization and document camera analysis. They provide information about the teacher’s current
location and document camera usage respectively. The detailed description of the image analysis methods is given in Section 3.1.4.

Actual video source selection is carried out by the component that utilizes the information provided by the image analysis component. In addition to selecting the source, the same component ensures that the video source is not changed too frequently. The selection process is described in detail in Section 3.1.5.

The final component connects the DEA to the existing auditorium control unit that takes care of the actual source switching. Since these parts are running in the same PC, there is no need for any additional hardware connection. The software based communication between the modules relies on TCP/IP protocols and sockets. After receiving a command through such a socket connection, the ACU interprets it and controls the necessary equipment.

### 3.1.3 User interface

To be able to start and configure the image analysis and decision making components, some kind of user interface is needed. We have, therefore, implemented a simple user interface which can be used to set and save the different parameters of the system. A user can, for instance, select the time limits and the close-view camera’s area to be applied in the video source selection process. The UI also gives visual information about the state of the components when the system is in operational mode.

Fig. 5 shows the main dialog of the user interface. The video capture windows seen on the upper part of the dialog show the current status of the image analysis methods.

![Fig 4. Main software components of the system. (Modified with permission from Huttunen et al. (2008), ©ACTA Press).](image-url)
The video source currently selected can be seen from the third capture window which is located at the bottom of the dialog. The main dialog also contains a progress bar which illustrates when the next video source switching can take place.

### 3.1.4 Image analysis

The software part of the implemented system contains two different image analysis components. One component is needed to define the teacher’s location in the classroom. The other image analysis component detects whether the document camera is actively used or not. In this section, an overview of the image analysis methods is given.

#### Locating the teacher

In order to make decisions concerning the appropriate image source, we need to know the actual location of the teacher in the classroom. This can be done by analyzing the images received from the overview camera that covers the area where the teacher can possibly move.

Locating the teacher in the classroom environment imposes many challenges. The classroom is usually dark and the lighting conditions change rapidly when the teacher switches slides on the screen. Due to these circumstances, most color-based and edge-based computer vision methods are not suitable for this task.
The teacher is usually moving or making gestures during the lecture so that motion information becomes an important and reliable cue. The only problem with this approach is that movements occurring in the audience can be distracting. Fortunately the teacher’s range of movement is usually known, which helps to distinguish the teacher’s movements from those of the audience.

The easiest way to find a moving object is to calculate the difference between two successive video frames (Lipton et al. 1998). Unfortunately, this method is not very useful because it usually detects only the edges of a moving object. To get better results, we use the hierarchical block matching algorithm (HBMA) (Wang et al. 2001), which provides us with more accurate motion information. HBMA is a special case of the multi-resolution motion estimation approach where the two successive video frames are represented by a pyramid. The spatial resolution is reduced by half, both horizontally and vertically, at each increasing level of the pyramid. The motion field obtained at the previous level is interpolated to form the initial solution for the motion at the current level. By using HBMA, the final motion field obtained is typically uniform, thus simplifying the analysis of the overall motion. Motion estimation algorithms like HBMA have been previously used mainly for video coding purposes.

On the basis of motion information received from HBMA, the teacher’s location can be calculated quite easily. HBMA gives motion vectors for each block in the image. To obtain just one image coordinate pair \((x, y)\), we calculate the center of mass of the motion vector field, which is selected as the teacher’s position in the overview camera view, as seen in Fig. 6. The white lines in Fig. 6b represent the motion vectors of the corresponding image blocks, and the calculated location is marked with a black circle.

![Fig 6. Locating the teacher. (a) Motion information of the image blocks and (b) teacher’s location. (Huttunen et al. 2008, ©ACTA Press).](image-url)
**Document camera analysis**

It is very difficult to see from the overall camera view when the teacher is using the document camera actively. To be able to recognize the actions that occur on the document camera, we have to also analyze the images coming from this particular camera. It is worth noting that those systems developed previously have not addressed this problem.

Traditionally, a document camera is taken into use when the teacher wants to show, for example, figures or charts to the audience. The teacher can also write on the paper placed on the document camera and, for instance, go through solutions for exercises, step by step. Therefore it is important to detect usage correctly to ensure that the audience can follow the teaching easily.

In order to analyze document camera usage effectively, we have to detect motion that occurs in the camera view. However, this is not enough, because we should also be able to recognize situations where an object has been placed on the document camera. Due to these requirements, two different image analysis methods have been implemented.

The motion detection method is based on a simple temporal differencing algorithm, which calculates the difference between two consecutive video frames. After the difference is calculated, the difference image obtained is thresholded, and the number of non-zero pixels is measured. If the number of pixels exceeds a certain predetermined value, there is a moving object in the image, for example, the teacher’s hand or pen.

As stated before, the other method implemented detects an object placed on the document camera. The main steps of the method are as follows. First, the edges in the image received from the document camera are detected using a Canny edge detector Canny (1986). In the second phase of the analysis, the number of edge pixels is measured. If the number of pixels is bigger than a selected threshold value, there is an object on the base of the document camera.

Fig. 7 contains examples of the images processed by both the motion and object detection methods. In this example situation, the teacher is pointing to a slide with a pen, so the difference between two successive video frames is quite remarkable, as seen in Fig. 7b. Also the characters and curves on a slide stick out from the white background clearly (Fig. 7c).
3.1.5 Video source selection

Selecting the correct video source plays an important role especially in a distance education situation. For example, it is not reasonable to transmit an empty document camera view to remote sites when the teacher is moving on the other side of the classroom. From the automated system’s point of view, this kind of simple situation can be handled quite easily.

Unfortunately, all the typical events taking place in a classroom are not simple at all. It is nearly impossible to create rules which would cover every possible situation. Some general advice is given by Microsoft Research which has studied the topic thoroughly (Rui et al. 2003). However, utilizing their results in our implementation is difficult because of the differences in the hardware and software platforms.

In our system, the video source selection is made between the overview camera, the close-view camera, and the document camera. The selection is based on information received from the image analysis components. Fig. 8 illustrates the inputs and the outputs of the parts related to the decision making process.

The teacher locating component provides knowledge about the teacher’s location in image coordinates \( (x, y) \). Document camera analysis gives information about the document camera usage. Both the motion and the object variable presented in Fig. 8 can have either the value ‘0’ or ‘1’. The final source selection is carried out by applying simple if-then rules.

The rule-based video source selection process can be modeled using a state machine approach as illustrated in Fig. 9. In total there are four separate states, which correspond to different activities. The state machine ensures that the source switching interval is
Video source selection is based on the teacher's location \((x, y)\) and document camera usage \((\text{motion, object})\). (Modified with permission from Huttunen et al. (2008), ©ACTA Press).

not too short. The user of the system can set the minimum viewing times for different cameras through the user interface. In Fig. 9, the minimum time for showing the object on the document camera is \(t_D\). For the other cameras, the minimum time is marked with \(t_k\). When the selected time limit has been reached, the actual video source selection takes place. If the source has to be switched, a new state is entered and the time counter \(t\) is set to zero.

The selection of the video source proceeds as follows. Initially the overall camera’s view is transmitted to remote participants. If the teacher starts using the document camera, the document camera view is selected. When motion appearing on the document camera view stops, and there is still an object present, the state ”object placed” is entered.

The other possibility is that the person giving a lecture is not using the document camera at that moment. In this case, the selection is made between the overview and the close-view camera. If the teacher moves to the close-view camera’s coverage area \(A\), the close-view camera is activated. Assuming that the conditions described above are not fulfilled, the last possible option is to select the overview camera.

### 3.1.6 Experimental results

Comparing the performance of our system to other similar systems that have been developed is difficult. Every system has its own requirements with regard to hardware and the physical environment. In order to evaluate the performance of our system, we compared the overall video quality of our system to that of a human operator. The
human operator used in the study has previous experience in video production.

To make a fair comparison between our system and the human operator, we had to use video material recorded earlier in a real classroom environment. The total length of the material was approximately 15 minutes. The three cameras in the lecture room were replaced by the videotape recorders, whose outputs were connected to the inputs of the AV-matrix. The test person carried out the source selection using the touch screen based user interface included in the ACU. In total there were three different camera views to choose from. The video source selection was made between the overview camera, close-view camera, and document camera, as shown in Fig. 10. An additional videotape recorder was connected to the output of the AV-matrix. The selections of the automated system and the human operator were recorded for further analysis.

To obtain numerical results, we took samples from both output videos every tenth second leading to 90 different situations in total. The selections made by the automated system and the human operator were compared with each other in these selected situations. Table 1 summarizes the results obtained in the form of a confusion matrix. In other words, the diagonal of Table 1 represents the number of situations where the selections made are the same. In total, the automated system changed the video source
Fig 10. Views of the three different cameras. (a) Overview camera, (b) close-view camera, and (c) document camera. (Huttunen et al. 2008, © ACTA Press).

Table 1. Results of the video source selection.

<table>
<thead>
<tr>
<th>Human operator</th>
<th>Overview</th>
<th>Close-view</th>
<th>Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td></td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>Close-view</td>
<td>2</td>
<td>21</td>
<td></td>
</tr>
<tr>
<td>Document</td>
<td>4</td>
<td>6</td>
<td>19</td>
</tr>
</tbody>
</table>

31 times whereas the human operator made source changes 20 times during the testing.

We can for example see that the human operator has selected the overview camera in 36 situations. The automated system has selected the close-view camera twice, and the document camera four times in these situations. If we look at the bottom of the right column instead, we can see that the automated system has detected the usage of the document camera correctly.

The differences between the selections seen in Table 1 are quite understandable. The human operator can listen to the teacher actively, and in that way receives more information about the current situation. The rule-based decision making system implemented just reacts to the teacher’s movements and document camera activity. An example is a situation where the paper placed on the document camera is not relevant and cannot be determined by the system. This causes the system to select the document camera also in those situations where it should not have done.

When we analyze the results presented in Table 1, we should remember that selecting the most suitable video from multiple views is always based on the human operator’s own opinion. Especially the decision between the overview camera and the close-view
camera is not always trivial. Therefore, the selections made by the test person would not necessarily satisfy all the viewers if the test video recorded was shown to a real audience. It is also worth noting that the video sequence used in testing contained quite a lot of action in order to get comparable results.

In addition to the aforementioned tests, the system developed has been taken into active use in many secondary schools and universities in Finland. The preliminary user feedback from the field has been mainly positive, and the advantages of the automated system seem to be clear. However, there are also users who are suspicious of using this kind of new technology.

3.2 Active PTZ tracking

3.2.1 Introduction

In the proposed system, a pan-tilt-zoom (PTZ) camera is designed to track the teacher when s/he walks at the front of the classroom (Huttunen & Heikkilä 2006). Tracking is carried out based on the information provided by the pan-tilt-zoom camera itself. The initialization phase of tracking utilizes the human head shape and motion information. After the initialization step, the color features used in tracking can be selected and updated online. Finally, the location of the person’s head is found and tracked by applying a modified version of the Continuously Adaptive Mean Shift (CAMSHIFT) (Bradski 1998) algorithm. On the basis of the tracking result, a PTZ camera is steered to keep the target in the camera view. If tracking failure occurs, the system returns to the initialization phase. Fig. 11 gives an overview of the system.

![Fig 11. Overview of the PTZ tracking system. (Huttunen & Heikkilä 2006, ©IEEE).](image)

In general, automatic PTZ tracking can be divided into two categories. The first option, also applied to this work, is to use mechanical PTZ cameras that are controlled through a certain command interface. The other way is to use virtual camera control
based on panoramic video capturing. However, the virtual camera control based systems require that the camera’s video resolution is higher than the required output resolution.

Perhaps the most widely used hardware setting for instructor tracking is a combination of a static panoramic camera and a PTZ camera. For example, AutoAuditorium (Bianchi 1998, 2004) used a static wide-angle camera to watch the entire area of interest, the image is then analyzed by software and the position of the instructor is detected. The position information is used to point the PTZ camera at the object of interest to give a better shot of the object of interest. A similar approach is used in the iCam system (Rui et al. 2004). In this setting, a static panoramic camera equipped with a wide horizontal field view, is used to track the movement of the instructor. Tracking result is then used to guide the PTZ camera in order to keep the instructor at the center of the view.

It is also possible to combine digital cropping and mechanical tracking. A recent approach is introduced by Microsoft Research (Zhang et al. 2005). They present a hybrid speaker tracking scheme based on a single PTZ camera in an automated lecture capturing system. Since the camera’s video resolution is usually higher than the required output resolution, they frame the output video as a sub-region of the camera’s input video. That allows tracking the speaker both digitally and mechanically. According to their work, digital tracking has the advantage of being smooth, and mechanical tracking can cover a wide area. The tracking is based on the motion information obtained from the camera image.

The system introduced by Sun et al. (2005) targets applications such as classroom lectures and videoconferencing. First, wide-angle video is captured by stitching video images from multiple stationary cameras. After that, a region of interest (ROI) can be digitally cropped from the panoramic video. The ROI is located from motion and color information in the uncompressed domain and macroblock information in the compressed domain. As a result, the system simulates human controlled video recording.

A classroom or a meeting room environment is not the only place to apply active PTZ tracking. Especially visual surveillance solutions include some active tracking components. For example, the work done in (Micheloni & Foresti 2005) addresses the problem of continuous tracking of moving objects with a PTZ camera. Firstly, a set of good trackable features belonging to the selected target is extracted. Secondly, their system adopts a feature clustering method that is able to discriminate between features associated with the background and features associated with different moving objects. In another surveillance related work (Greiffenhagen et al. 2001), the task of the two camera system is to continuously provide zoomed-in high-resolution images of the face.
of a person present in the room.

Usually the active tracking methods introduced are based on skin color. The work done by Comaniciu & Ramesh (2000) employs the Bhattacharyya coefficient as a similarity measure between the color distribution of the face model and face candidates. A dual-mode implementation of the camera controller determines the pan, tilt, and zoom camera to switch between smooth pursuit and saccadic movements, as a function of the target presence in the fovea region. Another system (Yang et al. 2006b) combines color-based face tracking and face detection to be able to track faces under varying pose.

One example of active head tracking is Jeong et al. (2005) which utilizes both color and shape information. The shape of a head is assumed to be an ellipse and a model color histogram is acquired in advance. Then, in the first frame, the appropriate position and scale of the head is determined based on user input. In the following frames, the initial position is selected at the same position of the ellipse in the previous frame. The Mean Shift procedure is applied to make the ellipse position converge on the target center where the color histogram similarity to the model and previous one is maximized. The previous histogram in this case is a color histogram adaptively extracted from the result of the previous frame.

The tracking system implemented and described in this thesis is mainly built on existing methods and algorithms which are brought together in a novel way to form a complete PTZ tracking solution. As a result, we have a PTZ tracking system that is able to operate robustly under large scale changes and different lighting conditions. The active tracking methods mentioned above lack the ability to cope with scale changes and update color features in parallel with color-based tracking. Generally speaking, the most important thing is that the system developed does not require any manual control and is able to recover from erroneous situations.

The remainder of this section is organized as follows. Section 3.2.2 contains a review of the initialization and feature selection methods as well as the description of the feature update and tracking algorithms implemented. In Section 3.2.3, we describe our PTZ control hardware and methodology. Preliminary experiments are reported in Section 3.2.4.
3.2.2 Feature selection and tracking

Initialization

In order to start tracking, the initial position of the target should be found. The main problem when using color information is how to get the object color model without a priori knowledge. Especially in those environments with changing lighting conditions, it is nearly impossible to use any fixed color model acquired beforehand. To cope with this problem, some other non-color based method is needed for finding the original position. In this work, the target is the speaker’s head, which can be localized by utilizing shape and motion information. After the initial position of the head is acquired, the color based tracking method can select the features used.

The face detector used in this work has been initially proposed by Viola & Jones (2001) and improved by Lienhart & Maydt (2002). The actual implementation of the detector is based on the Intel Open Computer Vision Library (OpenCV) (Bradski & Kaehler 2008, Willow Garage 2011). Unfortunately the aforesaid face detector can occasionally give wrong results, which makes the initialization more difficult. Since the speaker is usually moving when giving a presentation, it is reasonable to use the head detector only inside the areas where motion has been detected. In addition, the head has to be detected in several successive frames, before the tracking phase can take place. This kind of approach decreases the possibility of a false alarm substantially.

In the current system, a simple video frame differencing method is utilized for detecting motion. Frame differencing is a computationally inexpensive way to reduce the size of the scanned region, thus ensuring real-time operation of the whole method.

Feature selection and update

When the location of the head has been found in the previous phase, the color-based tracking can take place. In the lecture room, the lighting conditions can change rapidly due to, for example, slide changes. This is the main reason that the features used in tracking have to be updated on-line. Therefore the solution used in this pan-tilt-zoom tracking system relies on a modified method originally introduced by Collins et al. (2005b).

In the original method described in (Collins et al. 2005b), the feature update process is invoked only periodically. Unfortunately, the update process is computationally
heavy and therefore time consuming, which means that the object tracked can be lost
during the update process. Since the feature update process can take a long time, a new
approach to the problem is proposed.

In this system, the features used are updated continuously as fast as possible without
causing any unwanted delays to tracking. The feature update process is run in parallel
with the object tracking, as Fig. 12 illustrates. When the new features are available, they
are utilized immediately. Such an approach guarantees that the tracking method can
adapt to sudden changes occurring in the environment. To make tracking more robust,
the original samples in the first frame are combined with the pixel samples from the
current frame.

**Head tracking**

In our system, the size and location of the object changes during tracking due to both
object movement and camera zooming. Therefore, the traditional mean shift cannot be
utilized. The CAMSHIFT algorithm instead has been proposed for the tracking of the
head and face in a perceptual user interface (Bradski 1998).

In order to use the CAMSHIFT algorithm to track colored objects in a video scene, a
probability distribution image of the desired color in the video frame must be created.
Each pixel in the probability image represents a probability that the color of the pixel from an input frame belongs to the object. In the original CAMSHIFT algorithm, the probability image is calculated using only a 1-D histogram of the hue component. Therefore the algorithm may fail in cases where hue alone cannot distinguish the target from the background.

In the system implemented, the probability image is obtained from the log-likelihood ratio images which are generated for every incoming frame using the features selected (Collins et al. 2005b). In the ratio image, object pixels contain positive values, whereas background pixels contain negative values. Only the top three features selected are used to compute the ratio images for the current frame. Fig. 13 illustrates how the ratio image on the right side gives a good basis for tracking.

The center and size of the head are found via the modified CAMSHIFT algorithm operating on every ratio image independently, as shown in Fig. 14. The initial search window size and location for tracking are set using the results reported from the head detector. Since the number of features used is three, we get three ratio images for one incoming frame. The results of the tracking algorithm for the different images are combined by calculating the average. The current size and location of the tracked object are used to set the size and location of the search window in the next video image. The process is then repeated for continuous tracking.

To be able to recover from erroneous situations, the system has to detect tracking failures. In the current approach, the target is reported as lost when the size of the search window is smaller than a threshold value given in advance. Also the number of object pixels in the ratio images has to be clearly above a predetermined threshold in order to continue tracking.
3.2.3 Pan-tilt-zoom control

Hardware

The basis of the system is a Sony EVI-D31 PTZ camera, which has been widely used especially in video conferencing solutions. It offers a wide range of tunable parameters for PTZ control, as presented in Table 2. Since there are many variables that can be adjusted, steering the camera can be done smoothly. This is one of the main reasons for selecting this particular camera model.

Table 2. Technical details of the Sony EVI-D31 PTZ camera.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pan range</td>
<td>200 deg in 1760 increments</td>
</tr>
<tr>
<td>Tilt range</td>
<td>50 deg in 600 increments</td>
</tr>
<tr>
<td>Pan speed range</td>
<td>from 0 to 65 deg/sec in 24 discrete steps</td>
</tr>
<tr>
<td>Tilt speed range</td>
<td>from 0 to 30 deg/sec in 20 discrete steps</td>
</tr>
<tr>
<td>Zoom range</td>
<td>from f5.4 to f64.8 mm in 12 discrete steps</td>
</tr>
</tbody>
</table>
Adequate control of the PTZ camera is an essential part of the system implemented. The camera should react to movements of the target depending on how large those movements are. When the person is moving slowly, the camera should be able to follow smoothly. In the opposite situation, the camera control has to be fast to keep the speaker’s head in the field of view (FOV).

The software components of the PTZ control module are presented in Fig. 15. The user interface of the control system is developed in C++ and is only briefly described here. In practice, it includes buttons for starting and stopping the system, as well as offering a way to configure the parameters of the system.

To provide a communication channel between the hardware components of the system, there is a standard RS-232C serial port connection between a regular Pentium IV PC and a Sony EVI-D31 camera. The control commands from the PC to the camera are sent using Sony’s VISCA protocol. It is worth noting that tracking is not interrupted by the command execution.

When the tracked target, the person’s head in this case, moves to the boundary of the FOV, the PTZ camera is panned/tilted to keep the speaker inside the FOV. The current position and of the head in the image is received from the tracking method described in the previous section.

To ensure smooth operation, the pan and tilt speeds of the camera are adjusted actively when the object is moving. In practice, this means that the pan and tilt speeds are increased if the target is getting out of the FOV despite the current camera movement. When the teacher stops moving, the zoom level of the camera is set to keep the size...
of the head in the image the same all the time. The desired size of the object can be
determined beforehand.

Since the camera pan and tilt speeds are limited to a particular range, and the tracking
method fails, there may be situations when the camera loses the person from the FOV.
To solve this problem, a simple lost and found strategy is utilized. When the target is
lost, the camera is stopped for five seconds. If the teacher does not appear in the view
during this time, the camera zooms out and returns to the initial position waiting until
the initialization method has detected the object again.

3.2.4 Experimental results

Comparing the performance of our system to other similar systems that have been
developed is difficult. Every system has its own requirements for the hardware and the
physical environment. It is also impossible to run all the possible systems at the same
time to compare their real-time performance. However, to find out the performance of
our system, some preliminary real-time tests have been conducted in a room where
the lighting conditions are not uniform. The tests are run on a standard PC (Pentium
IV at 3.0 GHz, 1 GB RAM). The video image size is CIF (352×288) captured by the
aforementioned Sony EVI-D31 camera at 25fps.

In the test sequence presented below, a person is sitting in front of a computer screen
at the beginning. First the person’s head is detected when he is looking at the computer
screen. After a while, the person stands up and starts walking around the room. The
camera is panned and tilted to keep him in the FOV. To visualize the operation of the
system during the test, we selected some clips from the output video (Fig. 16a). When
the person walks away from the camera, the camera zoom is adjusted to keep the size of
the head constant. In the latter situation, the person has returned to the starting point, as
shown in Fig. 16b.

As mentioned earlier, the lighting conditions in the test room are not stable. In
addition, the PTZ camera used includes automatic gain and white balance control, which
means that the color features used for tracking have to be updated constantly. During the
test, approximately 10 frames pass on average before a feature update is ready. Since the
execution time of the feature update process depends on the current size of the object
tracked, the exact time between updates can vary.

As we can see from the test sequence, the system can adapt to lighting condition
changes. Also the PTZ control is able to follow the person without difficulties. However,
Fig 16. Tracking results. (a) The person leaves the table and starts moving around the room and (b) the person walks away from the camera and returns to the starting point. (c) Tracking almost fails when the person stands up quickly. (Huttunen & Heikkinen 2006, ©IEEE).
it is possible that the serial port connection between the PC and the PTZ camera may cause unwanted delays. This problem appears clearly in the sample shown in Fig. 16c where the person stands up rapidly. In this particular situation, the tilt speed is not adjusted quickly enough due to buffering in the command sending and receiving.

To illustrate how the PTZ tracking system could be integrated into the current classroom system, a simple demonstration was also prepared. In the demonstration, the video source was switched between the overview and tracking camera. When the tracking had been initialized and was running, the PTZ camera was selected as the output. In the opposite situation, for example in a case of a tracking failure, the overview camera was selected. The user interface of the demonstration system is shown in Fig. 17. The currently selected output is shown in the capture window of the dialog, on the right.

3.3 Discussion

In this chapter, an automated camera-based distance education system which selects and switches the video source automatically has been introduced. The system observes the teacher’s actions using cameras installed in the classroom. Rule-based video source selection is made on the basis of both the teacher’s location and the document camera activity. The final video source switching is handled by the auditorium control unit
which controls the equipment in the distance education classroom.

The main difference between the system we have proposed and the systems described in Section 3.1.1 is that our system already works in a real lecture room environment. The systems usually need some special hardware or operate only in a laboratory environment. The solution presented in this thesis utilizes the existing classroom equipment and the control unit, which can be selected freely. Therefore, it does not require any hardware designed especially for the system implemented. Finally, the solution described above introduces a way of analyzing the document camera usage which has not been included in previous systems. Otherwise, comparing the performance to other systems is difficult given that every system has its own requirements for the hardware and the physical environment. It is also impossible to run all the possible systems at the same time to compare their real-time performance.

In the second part of the chapter, a PTZ tracking system which can be used to follow the teacher actively in the classroom was introduced. The system detects the lecturer, using head shape information. The features applied to tracking are selected and updated online, which makes the method used tolerant to lighting condition changes. The PTZ module keeps the teacher’s head in the field of view by controlling the pan and tilt speeds as a response to the teacher’s movements.

The results obtained indicate that the decisions made by the automated system and the test person were the same in most cases. The image analysis methods can be provide enough information for video source selection, and especially the document camera plays an important role in the current setting. Also the communication channel between the decision making block and the control unit does not cause any latency when carrying out video source switching. With the PTZ tracking component, the automated video source selection and switching system can be extended to provide better results. Having said that, there are some other issues that should be taken into consideration in future work.

It goes without saying that the rule-based video source selection method cannot handle all situations correctly. The biggest difference between the automatic system and human operator is that a person can listen to the teacher. Therefore, adding speech recognition would increase the reliability of the system but, in spite of their attractiveness, these approaches are outside of the scope of this thesis.

Another alternative for increasing robustness is to deploy multiple cameras around the classroom. It would then be possible to utilize more sophisticated action recognition methods and perform deeper analysis of the teacher’s activity. Also the reliability
of the PTZ tracking component could be improved with the help of static cameras. Unfortunately, the active camera could not be integrated into the automated system due to schedule reasons. In addition, integration of the electronic material and other external video sources to the existing system should be considered.

Despite its shortcomings, the system presented in this chapter can clearly provide relief for the teacher when using the distance education system. It also serves as a demonstration of how proactivity can be put into action in a classroom environment. Since the use of video conferencing technology in distance education is increasing all the time, there is a growing need for the solution presented here. Finally, it is worth mentioning that the Academy of Finland found the work on the Distance Education Assistant very interesting and novel (Academy of Finland 2007).
4 Automatic landscape shooting mode selection for mobile devices

4.1 Introduction

Most digital cameras and mobile devices equipped with a camera nowadays support a number of different shooting modes for use in various situations (Fig. 18). In different modes the camera determines all aspects of settings, including exposure, aperture, focusing, and white balance. For instance, the landscape mode sets the camera up with a small aperture to make sure as much of the scene will be in focus as possible, in other words, it enables a large depth of field.

Usually a user has to select manually the most appropriate mode, which can be difficult and annoying. Automatic scene mode selection can therefore be seen as an interesting opportunity and application for proactive computer vision. Because the sensor needed for the analysis is already present, it is natural to apply computer vision methods in order to automate the shooting process. Interestingly, in this case the sensor and the actuator are squeezed into a one compact device.

When developing a system capable of automatic landscape mode selection, it is essential to be able to recognize landscape scenes. The detection of landscape scenes is a difficult problem given the fact that several landscape scenes have similar objects to those in non-landscape scenes, and vice versa. Furthermore, illumination conditions are equally unpredictable in both cases. Due to the computational restrictions set by the target devices, the primary goal is to find an accurate but still computationally light solution capable of real-time operation. This chapter therefore concentrates on studying

Fig 18. Different shooting modes on a Nokia N900.
different approaches that can be used in recognizing landscape images (Huttunen et al. 2011).

Definition of landscape and non-landscape images is not totally straightforward. In this work, we assume that if there are no distinct and easily separable objects present in a natural scene, the image is classified as landscape. From a photographic point of view, this requirement would mean that as much of the scene as possible should be in focus. As a result of the aforementioned restrictions, the landscape category would contain sunset, beach, mountain, etc., subcategories. Having said that, it is obvious that all images taken indoors should be classified as non-landscape. In this case, the non-landscape branch would consist of indoor scenes and other images containing man-made objects at relatively close distance (Fig. 19).

Recognizing landscape images can be thought of as a special case of scene classification which aims at labeling an image into a set of different semantic categories. In addition to the application at hand, knowledge of the scene type provides important information in a number of applications that deal with consumer photographs and digital cameras. Generally, determining the scene type is the starting point of further image analysis and search in large image collections (Bianco et al. 2008, Datta et al. 2008). From this point of view, the results obtained and described later in the chapter play an important role when a fast method for separating the landscape and non-landscape scenes is required. This kind of classification can serve as a preprocessing step for guiding and speeding-up content-based image retrieval (Smeulders et al. 2000, Vailaya et al. 2001) in large databases and improving accuracy, or for performing automatic image annotation (Datta et al. 2008).

Even though there does exist a number of different approaches concerning scene classification, to the best of our knowledge, there are no other works concentrating on classifying images into the landscape and non-landscape categories. The previous
works differ by the number of the scene classes, the image representations, and the
classification method. The most methods so far have aimed at classifying into a small
number of scene categories, including indoor/outdoor (Kim et al. 2010, Payne & Singh
and subsets of urban and natural scenes (Lazebnik et al. 2006, Oliva & Torralba 2001,
Vailaya et al. 2001). It can be noticed that none of these categorizations is directly
applicable in our problem.

The rest of the chapter is organized as follows. Section 4.2 gives a detailed description
of the different features and classifier used in this study. The experimental results are
presented in Section 4.3. Finally, the conclusions are summarized in Section 4.4.

4.2 Methods for landscape scene recognition

There are two main elements in a typical image classification system. The first one is
responsible for the computation of the feature vector representing an image, whereas the
second part is the classifier, the algorithm that classifies an input image into one of
the predefined categories based on the feature vector. When real-time performance
of the system is considered, they both have an important contribution to the final
outcome. In this section, we describe two approaches for landscape/non-landscape
image classification. We begin with the image representation models followed by the
classifier engine.

4.2.1 Global features

A common way to categorize images is to compute low-level features, such as color
and texture, which are further processed with a classifier engine for inferring high-
level information about the image. These methods assume that the type of scene can
be directly described by the color or texture properties of the image. In fact it has
been shown that low-level features can give very comparable results with many scene
The work done by Serrano et al. (2002) employs low-level color and texture features,
whereas Szummer & Picard (1998) concatenate the histograms in the Ohta color space
with texture and frequency features. Later Payne & Singh (2005) have introduced an
indoor/outdoor classification technique based on edge analysis.

In this subsection, two different approaches based on a global description of
the image content are presented. They both provide one feature histogram which encapsulates the structure of the scene.

**GIST**

One of the most well known global approaches in scene categorization is the GIST descriptor that was initially proposed by Oliva & Torralba (2001). The main idea of this approach is to develop a low dimensional representation of the scene, which does not require any form of segmentation. The authors propose a set of perceptual dimensions (naturalness, openness, roughness, expansion, ruggedness) that represent the dominant spatial structure of a scene. They show that these dimensions may be reliably estimated using spectral and coarsely localized information. Later Oliva & Torralba (2006) describe a formal approach to the representation and the mechanism of scene gist understanding, based on scene-centered, rather than object-centered primitives. It is shown that the structure of a scene image can be estimated by the mean of global image features, providing a statistical summary of the spatial layout properties of the scene.

In the actual implementation, Oliva & Torralba (2001) employ a bank of Gabor filters in the frequency domain, tuned to different orientations and scales. To compute the color GIST description, the image is first divided into a $4 \times 4$ grid on which orientation histograms are extracted. Our implementation builds the descriptor from eight oriented edge responses at four scales and produces a vector of dimension 1536. Most of the works using the GIST descriptor resize the image as a preliminary stage, producing a small square image whose width typically ranges from 32 to 256 pixels. In this study, the images are rescaled to a $240 \times 240$ size, irrespective of their original aspect ratio. This is sufficient since the descriptor does not need to represent the details of an image.

**Local binary pattern (LBP)**

The discrete occurrence histogram of the LBP patterns (Ojala et al. 1996) computed over an image or a region of the image is shown to be a very powerful texture feature. The local binary pattern detects microstructures, for example, edges, lines, spots, and flat areas, whose underlying distribution is estimated by the histogram. By computing the occurrence histogram, structural and statistical approaches are combined.

In LBP (Ojala et al. 1996), the original $3 \times 3$ neighborhood is thresholded by the value of the center pixel. The values of the pixels in the thresholded neighborhood are
Fig 20. (a) The values of the pixels in the thresholded neighborhood are multiplied by the weights given to the corresponding pixels. Finally, the values are summed to obtain the number (241) of this texture unit. (b) The LBP histograms are computed in the center (in) and on the boundary areas (out) separately. The final image representation is then a concatenation of these two histograms (in+out). (Huttunen et al. 2011, ©Springer).

multiplied by the weights given to the corresponding pixels. Finally, the values of the eight pixels are summed to obtain the number of a single texture unit. In order to give better insight into how the LBP code is computed, a simple example is depicted in Fig. 20a.

When we think about landscape images depicting natural scenes, usually the center of the image does not contain any distinctive objects. Therefore it is reasonable to utilize this information by computing the histograms in the center and on the boundary areas surrounding the center separately (Fig. 20b). The final image representation is then a concatenation of these two histograms, providing us with a 512 bins long representation. From here onward it is referred as LBP_{50}, and the basic version of the LBP is annotated by LBP_{b}.
4.2.2 Local features

A common approach in image categorization is to use some local features combined with the bag-of-words (BOW) representation which describes an image as an orderless collection of local features (Csurka et al. 2004, Yang et al. 2007). The basic idea of these approaches is that a set of local image patches is sampled either densely, randomly, or using a keypoint detector. After the sampling, a vector of visual descriptors is computed on each image patch independently (Fig. 21). There is a large number of different methods that can be used for describing the image patch content. One of the most popular approaches is to use SIFT-based descriptors (Lowe 2004, van de Sande et al. 2010), but also histograms or moments can be considered (van de Sande et al. 2010). Regardless of the choice of method, the resulting collection of descriptors is vector quantized and the global word histogram obtained is used as a characterization of the image.

In this study, the descriptors (see Table 3) were extracted using dense sampling with a step size of 10 pixels and default scale defined in the binary implementation (van de Sande et al. 2010). For more information about the descriptors and their implementation details, please refer to (van de Sande et al. 2010). The descriptor quantization was done by k-means clustering, resulting in a vocabulary of 1000 words. To be independent of the total number of descriptors in an image, the sum of the final feature vector was normalized to 1.

Fig 21. The stages of the bag-of-words approach. First, the sample points are picked from the image. Then, for every point a color descriptor is computed over the area around that point. All the descriptors are subsequently vector quantized against a codebook of prototypical descriptors. This results in a fixed-length feature vector representing the image. (Huttunen et al. 2011, ©Springer).
Table 3. Local descriptors. For details on the descriptors, see van de Sande et al. (2010).

<table>
<thead>
<tr>
<th>Type</th>
<th>Descriptors</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIFT</td>
<td>rgsvift</td>
</tr>
<tr>
<td></td>
<td>csift</td>
</tr>
<tr>
<td></td>
<td>huesift</td>
</tr>
<tr>
<td></td>
<td>opposedsift</td>
</tr>
<tr>
<td></td>
<td>rgbsift</td>
</tr>
<tr>
<td></td>
<td>hsvsift</td>
</tr>
<tr>
<td></td>
<td>sifit</td>
</tr>
<tr>
<td>Histogram</td>
<td>huehistogram</td>
</tr>
<tr>
<td></td>
<td>nrghistogram</td>
</tr>
<tr>
<td></td>
<td>rgbhistogram</td>
</tr>
<tr>
<td></td>
<td>transformedcolorhistogram</td>
</tr>
<tr>
<td></td>
<td>opponenthistogram</td>
</tr>
<tr>
<td>Moment</td>
<td>colormomentinvariants</td>
</tr>
<tr>
<td></td>
<td>colormoments</td>
</tr>
</tbody>
</table>

4.2.3 Classification

The Support Vector Machine (Vapnik 1998) is widely used in scene classification, and therefore it is selected as a classifier in this work. Even though the linear SVM is light in terms of computational burden, based on our preliminary evaluations we employ the RBF kernel in this study. In our application, the classification step is carried out only once per image, thus its effect on overall time cost is minimal. When computing the kernels, the distance function is $\chi^2$ with LBP s and local features, whereas the GIST features are compared with the L2 norm. The SVM classifier implementation used in the experiments builds upon a publicly available LIBSVM library (Chang & Lin 2001).

4.3 Experimental results

Comparative evaluation has been carried out between the methods described in Section 4.2. Combinations of the features were not considered because such approaches would be too complex in view of the practical applications.

4.3.1 Image sets

The images used for training and testing of the SVM classifier were downloaded from the PASCAL Visual Object Classes database (Everingham et al. 2007) and the Flickr site (Flickr 2010). All the images mentioned below were manually labeled and resized
to QVGA (320×240) resolution, apart from the GIST which uses 240×240 images.

Training data set

The combined training and validation database contains 1115 landscape images and 2617 non-landscape images. Approximately 20% of the training images were used for validation of the SVM classifier.

Testing data set

The testing database contains 912 landscape images and 2140 non-landscape images. As with the training images, most of the landscape images come from the Flickr database and the non-landscape images originates mainly from the VOC2007 collection.

4.3.2 Evaluation criteria

The classification task will be evaluated by the precision/recall curve, and the principal quantitative measure used is average precision (AP). In addition, the performance will be evaluated by the Receiver operating characteristic (ROC) curve. In this case, the measure used is the area under curve (AUC).

Furthermore, we report the true positive and false positive rates (TPR and FPR, respectively) of the different approaches when the threshold for the SVM decision value is set to zero. In our case, the definitions for the test images are as follows:

– False positive (FP): non-landscape classified as landscape
– True positive (TP): landscape classified as landscape

4.3.3 Results

The precision/recall and ROC curves are illustrated in Fig. 22. For clarity, only the best performing methods are included in the figures, but Table 4 summarizes all the results in a numerical form. It can be seen that the LBP based approaches perform best both in terms of AUC and AP. It is worth noting that the LBPio approach, which concatenates the histograms computed in the image center and boundary area gives a better performance than LBPb.
Fig 22. Result curves of the best performing methods. (Huttunen et al. 2011, ©Springer).

Table 4. Summary of the results.

<table>
<thead>
<tr>
<th>Descriptor</th>
<th>Classification</th>
<th>AUC</th>
<th>AP</th>
<th>TPR</th>
<th>FPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Global</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LBPb</td>
<td></td>
<td>0.972</td>
<td>0.939</td>
<td>0.862</td>
<td>0.055</td>
</tr>
<tr>
<td>LBPio</td>
<td></td>
<td>0.982</td>
<td>0.958</td>
<td>0.882</td>
<td>0.040</td>
</tr>
<tr>
<td>GIST</td>
<td></td>
<td>0.963</td>
<td>0.924</td>
<td>0.809</td>
<td>0.050</td>
</tr>
<tr>
<td>Local descriptors + BOW</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rgfsift</td>
<td></td>
<td>0.969</td>
<td>0.934</td>
<td>0.814</td>
<td>0.045</td>
</tr>
<tr>
<td>csift</td>
<td></td>
<td>0.966</td>
<td>0.926</td>
<td>0.825</td>
<td>0.052</td>
</tr>
<tr>
<td>opponentsift</td>
<td></td>
<td>0.966</td>
<td>0.922</td>
<td>0.828</td>
<td>0.052</td>
</tr>
<tr>
<td>rgbfsift</td>
<td></td>
<td>0.960</td>
<td>0.918</td>
<td>0.804</td>
<td>0.047</td>
</tr>
<tr>
<td>hsvsift</td>
<td></td>
<td>0.959</td>
<td>0.915</td>
<td>0.804</td>
<td>0.050</td>
</tr>
<tr>
<td>sift</td>
<td></td>
<td>0.956</td>
<td>0.901</td>
<td>0.806</td>
<td>0.059</td>
</tr>
<tr>
<td>huesift</td>
<td></td>
<td>0.954</td>
<td>0.902</td>
<td>0.791</td>
<td>0.067</td>
</tr>
<tr>
<td>colormomentinvariants</td>
<td></td>
<td>0.926</td>
<td>0.857</td>
<td>0.737</td>
<td>0.067</td>
</tr>
<tr>
<td>transformedcolorhistogram</td>
<td></td>
<td>0.924</td>
<td>0.851</td>
<td>0.692</td>
<td>0.061</td>
</tr>
<tr>
<td>opponenthistogram</td>
<td></td>
<td>0.909</td>
<td>0.825</td>
<td>0.689</td>
<td>0.079</td>
</tr>
<tr>
<td>rgbhistogram</td>
<td></td>
<td>0.903</td>
<td>0.805</td>
<td>0.683</td>
<td>0.087</td>
</tr>
<tr>
<td>colormoments</td>
<td></td>
<td>0.897</td>
<td>0.811</td>
<td>0.697</td>
<td>0.084</td>
</tr>
<tr>
<td>huehistogram</td>
<td></td>
<td>0.863</td>
<td>0.717</td>
<td>0.525</td>
<td>0.071</td>
</tr>
<tr>
<td>nrghistogram</td>
<td></td>
<td>0.861</td>
<td>0.727</td>
<td>0.601</td>
<td>0.102</td>
</tr>
</tbody>
</table>
Fig. 23 contains a collection of sample images when using the LBP<sub>Io</sub> representation. When looking at the false positive images (Fig. 23c), it can be seen that most of the images contain smooth areas around some object.

![Fig 23](image)

(a) Landscape classified as landscape.  (b) Non-landscape classified as non-landscape.

(c) Non-landscape classified as landscape.  (d) Landscape classified as non-landscape.

Fig 23. Classification examples with LBP<sub>Io</sub> representation. (Huttunen et al. 2011, ©Springer).

### 4.3.4 Computational cost

In order to evaluate the computational cost of the different image representations, the preliminary performance analysis was conducted on a regular Windows PC (Core 2 Duo 3.2 GHz, 4 GB RAM).

Our own LBP C code implementation was evaluated with the Visual Studio 2010 Profiler, whereas the execution times for the different color descriptors were obtained...
using the binaries publicly available (van de Sande et al. 2010). The results are shown in Table 5, and they include the time spent on descriptor computation as well as the total time for SVM classification. It is worth noting that the most time consuming part of the bag-of-words based methods is the word histogram computation.

Unfortunately, the GIST descriptor codes (Oliva & Torralba 2001) are available only for MATLAB. Therefore its performance was measured in these experiments using the C implementation by Douze et al. (2009). Naturally, we made sure that the descriptors provided by the C code were equal to their MATLAB counterparts. Since the GIST descriptor is computed using several filters corresponding to different orientations and scales, its computational cost is substantially higher than that of LBP.

Table 5. Computational cost of the methods.

<table>
<thead>
<tr>
<th>Descriptor</th>
<th>Execution time (s)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Descriptor</td>
<td>Total</td>
</tr>
<tr>
<td>Global</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LBP&lt;sub&gt;so&lt;/sub&gt;</td>
<td>0.001</td>
<td>0.005</td>
</tr>
<tr>
<td>LBP&lt;sub&gt;b&lt;/sub&gt;</td>
<td>0.001</td>
<td>0.003</td>
</tr>
<tr>
<td>GIST</td>
<td>1.473</td>
<td>1.502</td>
</tr>
<tr>
<td>Local descriptors + BOW</td>
<td></td>
<td></td>
</tr>
<tr>
<td>rg sift</td>
<td>0.340</td>
<td>2.699</td>
</tr>
<tr>
<td>cs sift</td>
<td>0.350</td>
<td>2.712</td>
</tr>
<tr>
<td>opponent sift</td>
<td>0.340</td>
<td>2.694</td>
</tr>
<tr>
<td>rgb sift</td>
<td>0.330</td>
<td>2.744</td>
</tr>
<tr>
<td>hsv sift</td>
<td>0.340</td>
<td>2.494</td>
</tr>
<tr>
<td>sift</td>
<td>0.120</td>
<td>0.595</td>
</tr>
<tr>
<td>hues sift</td>
<td>0.290</td>
<td>1.046</td>
</tr>
<tr>
<td>color moment invariants</td>
<td>1.410</td>
<td>1.444</td>
</tr>
<tr>
<td>transformed color histogram</td>
<td>0.100</td>
<td>0.147</td>
</tr>
<tr>
<td>opponent histogram</td>
<td>0.090</td>
<td>0.140</td>
</tr>
<tr>
<td>rgb histogram</td>
<td>0.070</td>
<td>0.118</td>
</tr>
<tr>
<td>color moments</td>
<td>1.340</td>
<td>1.376</td>
</tr>
<tr>
<td>hue histogram</td>
<td>0.180</td>
<td>0.223</td>
</tr>
<tr>
<td>nr histrogram</td>
<td>0.080</td>
<td>0.119</td>
</tr>
</tbody>
</table>
4.3.5 Real-time implementation

Based on the results presented above, it is obvious that the LBP histogram is the best choice when building a real-time system. On the other hand, selection between the two different LBP representations depends mainly on the requirements set by the target platform. Our current real-time implementation coded in C relies on the basic LBP, which gives reasonable results with lower memory consumption.

When we take a closer look at the profiler results of the final system, the results indicate that most of the time is spent on the SVM classifier since LBP histogram computation takes only one third of the overall processing time. The total execution time for one QVGA frame is about 3 ms, which guarantees real-time performance even on more constrained platforms.

Test videos

In order to evaluate the performance of the LBP approach in real-time scenarios, we captured several video sequences with different cameras (Canon EOS 5D Mark II, Logitech QuickCam Fusion, Nokia N95). All the videos were resized to QVGA resolution, but no other pre-processing steps were applied.

The results of the sequences are shown in Fig. 24. To illustrate how the detection of landscape scenes works with the given frames, we use green boundaries for the landscape and red boundaries for the non-landscape frames. If the boundary is black, the decision value of the classifier is close to zero and therefore selection between the classes cannot be done reliably.

Nokia N900 and FCam

Based on the results presented above, we have made a real-time implementation on a Nokia N900 device. The implementation is built on Qt SDK and the FCam library (Adams et al. 2010) and is written in C/C++. The preliminary results indicate that landscape detection achieves real-time performance since it takes approximately only 30 ms to process one frame.
Fig 24. Video frames with the classification results. Green boundaries are used for the landscape and red boundaries for the non-landscape frames. If the boundary is black, selection between the classes cannot be done reliably. (Huttunen et al. 2011, ©Springer).

4.4 Discussion

In this chapter, we have introduced a special case of proactive computer vision. Usually a user has to select manually the most appropriate shooting mode, which can be distracting. Automatic scene mode selection can therefore be seen as an interesting opportunity and application for proactive computer vision. Compared to the automated distance education system presented earlier in this thesis (Chapter 3), the difference in scale is quite remarkable. The earlier application is also based on a fixed setting, whereas the application presented here is movable and built into a single device. Also communication between the sensor and the actuator is straightforward because, in fact, they are the same device.

A system capable of automatic landscape mode selection has to be able to distinguish landscape from non-landscape scenes. On that account, the focus of this chapter has been concentrating on different approaches that can be used in automatic landscape scene recognition. Due to the computational restrictions set by the target devices, the
primary goal of our work has been to find an accurate but still computationally light solution capable of real-time operation.

From the methodological point of view, it is shown with the extensive experiments that a global texture-based approach competes with or outperforms other more complex methods in the landscape image recognition problem. It appears that the local features are too distinctive for the given task. The results obtained clearly indicate that the computational cost of the method relying on the Local Binary Pattern (Ojala et al. 1996) representation is low enough for real-time systems. The computational cost of the local features basically rules them out independent of the length of the word histograms. It is worth noting that even the images used for training the classifier were collected from on-line databases, the method can still give reasonable classification results on a wide variety of cameras. It should be also noted that the LBP operates on gray scale images, which means that the use of color information is not needed.

When mobile devices and proactivity are considered on a general level, there are a number of application possibilities. Based on context, devices can adapt their services and functions in order to help people in their every-day life. In addition, the application presented in this chapter was using only visual information to guide decision making. However, there are also other sensors (Lane et al. 2010) that can provide us with useful data that can be analyzed and turned into implicit commands. For example, ambient light, location, and acceleration sensors are valuable when devising proactive mobile systems.
5 An open framework for distributed sensor networks

Researchers building proactive multi-camera systems using their particular computer vision algorithm face the challenge of not only implementing the actual algorithm, but also having to build an infrastructure for sensor access and node communication, which is often even more laborious. As Afrah et al. (2008) point out, even simple tasks such as obtaining images from a camera are not trivial, primarily due to the large variability of video data sources. A generic framework which includes these essential parts would allow the users to focus on their area of expertise and allow rapid prototyping and development of distributed sensor networks.

This chapter presents an open and expandable framework for development of distributed sensor networks with an emphasis on peer-to-peer networking (Saastamoinen et al. 2008). The user is provided with easy access to sensors and communication channels between distributed nodes, allowing the effort to be focused on the development of computer vision algorithms and their use in distributed proactive environments. Furthermore, customizations are made possible by releasing the source codes under an open source license\(^1\).

5.1 Introduction

Historically, most networked camera systems have been centralized client-server systems, usually employing a central control unit. For example, in traditional visual surveillance networks, cameras act as independent units that continuously send video streams to a central processing server, where the video is analyzed by a human operator or computer. In other words, the sensor nodes do not process the data they capture but simply pass it on to a central processing node. Fig. 25a shows a simple centralized network in which the nodes do not have connections to, or maybe even know of the existence of other nodes. Their only communication channel is with the central server, so the central control unit becomes the obvious bottleneck of the setup. The centralized approach does not scale well for networks of hundreds or thousands of nodes, each of which has its own level of performance and stability. The elimination of a central control unit leads to

\(^1\)http://www.cse.oulu.fi/MVG/Downloads/Scallop
more capable and flexible distributed sensor networks of the type shown in Fig. 25b.

As the processing power available to the nodes has increased with the emergence of smart cameras and fast wireless networks, the focus has shifted towards intelligent, decentralized nodes (Valera & Velastin 2005). For multi-camera applications, image processing migrates from central workstations to the distributed embedded sensors. This distributed computing approach helps to reduce the communication load within the network of cameras and to increase the reliability and scalability of the multi-camera application (Rinner & Wolf 2008).

Due to the requirements mentioned above, the framework presented in this chapter focuses on peer-to-peer (P2P) networks as a communication channel in proactive applications. In P2P systems, each network node (a peer) acts in both a server and a client role, as opposed to the traditional server-client architecture. The benefits of P2P systems include scalability, self-organization and tolerance to faults in individual nodes (Steinmetz & Wehrle 2005). In addition to the obvious choice of their intended application, P2P networks can be classified by their connection architecture and degree of centralization (Steinmetz & Wehrle 2005).

As mentioned earlier, the two main areas of interest for a distributed sensor framework are node communication and sensor access. One previous system focusing on the communication aspect is the Scalable Clustered Camera System (SCCS), a peer-to-peer multi-camera system for tracking objects (Velipasalar et al. 2006, 2008). Instead of transferring control of tracking jobs from one camera to another, each camera in the presented system performs its own tracking, keeping its own trajectories for each target object, which provides fault tolerance. The authors identify the weaknesses of previous
centralized systems and propose a network of peer nodes without any central servers. The nodes communicate with each other using a Message Passing Interface (MPI) for distributed processing. Messaging is non-blocking, allowing nodes to use their processing power efficiently and not waste processor cycles idly waiting for a reply. MPI is designed for computing clusters and is not the ideal solution for ad-hoc networks, where nodes can join and leave at will. For these situations, a true peer-to-peer network is a more suitable solution.

MONNET by Albu et al. (2006) is a system composed of intelligent nodes which form an ad-hoc network and communicate appearance models of detected objects through multicasting. The nodes use a wireless network and a discovery protocol for contacting other nodes, while actual messaging is conducted using multicasting. Nodes can join and leave the network in a plug-and-play manner without affecting the operation of other nodes.

Enficiaud et al. (2006) have developed CLOVIS, a framework for visual surveillance applications. It provides the application developer with flexible interfaces for machine vision processing, sensor access and event based communication with other nodes. Node communication is implemented using a web service framework and distributed remote procedure calls, with data represented in Extensible Markup Language (XML). User code runs inside a runtime container, allowing language and platform independence.

Another approach has been taken by Dunkels et al. (2007). Focusing on low-power radio protocols, they propose and implement Chameleon, an architecture containing a communication stack (Rime) and a set of packet transformation modules. Nodes can communicate with each other using the primitives provided by the stack without having any specific knowledge of the underlying communication channel or the protocols used.

Lin et al. (2010) describe a distributed, peer-to-peer gesture recognition system along with a software architecture modeling technique and authority control protocol for ubiquitous cameras. This system performs gesture recognition in real time by combining imagery from multiple cameras without using a central server. The system is built on Windows machines, and uses standard video cameras as sensors and the local network as a communication channel.

Open Source Computer Vision (OpenCV) is a set of libraries that provide a function based approach for tasks and algorithms in computer vision (Willow Garage 2011). Even though it provides easy methods for accessing image files and higher level algorithms, it does not support issues involved in building vision systems, that is, dealing with source details or communication issues.
The previous fully distributed sensor systems like (Albu et al. 2006) have mostly used some form of multicasting for inter-node communication. Multicasting uses bandwidth efficiently but IP-level multicasting requires support from the underlying infrastructure, mainly network routers. For these reasons, previous systems have usually opted for application level multicasting, but none have selected a totally distributed P2P infrastructure suitable for ad-hoc networks. The framework proposed in the following sections is designed to support both of these approaches, with a P2P approach currently implemented.

The rest of this chapter is organized as follows. Section 5.2 presents the framework and the programming interfaces used to interact with the modules. Details of the experimental system are given in Section 5.3. The currently implemented parts of the system are presented in Section 5.3.1, followed by a look at the demonstration system in Section 5.3.2. The results from experiments on this system are shown in Section 5.3.3, followed by discussion in Section 5.4.

5.2 Framework description

Briefly, the framework aims to provide the user with simple and configurable interfaces for accessing sensors and communicating with other nodes. Various sensor and network types can be used, and uniform event based interfaces are provided for both sensors and networks, enabling node heterogeneity. Node deployment and runtime configuration is simplified through the use of XML configuration files. In summary, the framework has been designed and implemented with the following guidelines in mind:

– Allow researchers to focus on machine vision algorithms by providing access to sensors and communication networks.
– Allow new sensor and network types to be implemented easily by defining an interface that implementations must conform to.
– Keep the interfaces simple and uniform over different types of sensors and networks.
– Separate sensors and communication from machine vision algorithms.
– Allow customizations by releasing/open sourcing the code.
5.2.1 Architecture and interfaces

The system architecture is shown in Fig. 26. It consists of interfaces for sensor and network access, and modules for specific types that are implementing the interface. The commands and events used to interact with the resources are the same for each type, allowing the same code to be used while changing the resource type. Only the runtime configuration needs to be changed.

To use the interfaces, the end user creates an object implementing the interface, passes the configuration options to the object, registers a set of event handler functions and starts the sensor. All data from the sensor or network is then passed to the user using a set of events. These include state changes, data and informational events. An example of the sequence of events between the user code and the framework is shown in Fig. 27.

Sensor and network modules are configured with XML files, as shown in Fig. 28, making the runtime configuration of nodes simple and flexible. XML schema are provided for easy editing and validation of the configuration files.

Network interface

The network interface contains method definitions for interacting with the network layer. The user is able to configure the network module, and join and leave the network. Individual nodes are identified by a unique node ID. Messages can be sent to either

![Diagram of the system architecture](image_url)
Fig 27. A typical sequence of events between parts of the framework. (Saastamoinen et al. 2008, ©IEEE).

```xml
<?xml version="1.0" encoding="utf-8" ?>
<AxisCameraConfigSet DefaultConfig="LobbyTS339a_1"
    xmlns="Scallop/AxisCameraSchema.xsd">
  <AxisCameraConfig ConfigName="Lab1">
    <Address>c1.example.com</Address>
    ...
  </AxisCameraConfig>
  <AxisCameraConfig ConfigName="Lobby">
    <Address>c2.example.com</Address>
    ...
  </AxisCameraConfig>
</AxisCameraConfigSet>
```

Fig 28. Example of sensor configuration. (Modified with permission from Saastamoinen et al. (2008), ©IEEE).

86
all other nodes (broadcast), to a set of named nodes, or to nodes within a number of hops. The network layer uses event callbacks to inform the user of specific events and incoming messages. The user can also query the network layer for the network state and diagnostic information. The interface is geared towards mesh-based peer systems but can also be used for multicasting or other types of P2P networks.

The network primitives and a set of read-only properties that can be queried are shown in Table 6. Event signatures are provided for errors, incoming data and node state changes.

Messages are delivered asynchronously to peer nodes as XML documents, which allows serialization of data and user customization of messages. Any byte-oriented data can be encoded to a form suitable for inclusion in an XML structure thus making the sending of images and other binary data possible. The user can provide schema for the validation of outgoing and filtering of incoming messages. These can be used to filter the messages the user sees so that nodes can differ in their accepted message types and functionality, while still contributing to the passing of network messages to other nodes. A sample XML message is shown in Fig. 29.

Table 6. Network primitives and events.

<table>
<thead>
<tr>
<th>Primitive</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Join()</td>
<td>Joins a network.</td>
</tr>
<tr>
<td>Leave()</td>
<td>Leaves the network.</td>
</tr>
<tr>
<td>SendMessage(message)</td>
<td>Sends a message to all nodes.</td>
</tr>
<tr>
<td>SendMessage(message, hops)</td>
<td>Sends a message to nodes within hops.</td>
</tr>
<tr>
<td>MsgCounts</td>
<td>Returns number of messages sent and received,</td>
</tr>
<tr>
<td></td>
<td>along with sizes.</td>
</tr>
<tr>
<td>Neighbors</td>
<td>Returns a list of the node’s neighbors, if</td>
</tr>
<tr>
<td></td>
<td>applicable.</td>
</tr>
<tr>
<td>NodeId</td>
<td>Returns the node identification string.</td>
</tr>
<tr>
<td>State</td>
<td>Returns the state of the network</td>
</tr>
<tr>
<td></td>
<td>(Offline, Online, Error, . . .).</td>
</tr>
<tr>
<td>Data event</td>
<td>Incoming data event.</td>
</tr>
<tr>
<td>Info event</td>
<td>Information event, for debugging purposes.</td>
</tr>
<tr>
<td>StateChanged event</td>
<td>Event indicating the network state changed.</td>
</tr>
</tbody>
</table>
Sensor interface

There have been efforts to standardize image acquisition, including Video4Linux and IIDC for IEEE 1394 cameras (FireWire). For example, Video4Linux provides standardized access to video devices on Linux machines, but this solution is unfortunately platform specific. Other approaches, such as IIDC, have been successful in standardizing a specific subset of camera devices that use a particular bus. However, it has yet to spread to the vast majority of other video sources such as network cameras, capture cards, USB cameras, and video files (Afrah et al. 2008). In our framework, we encapsulate the actual implementation behind an interface, so the end-users do not need to worry about the technology of the camera used.

Our sensor interface provides methods for configuring a sensor, and starting and stopping it. Once started, the sensor passes data to the user through event callbacks. The interface is generic enough to allow different types of sensors to be used, including video sources and audio and proximity sensors.

The sensor is accessed through the primitives shown in Table 7. Event signatures for data, information and state changes are also defined in the interface.
Table 7. Sensor primitives.

<table>
<thead>
<tr>
<th>Primitive</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register()</td>
<td>Configures the sensor to be used.</td>
</tr>
<tr>
<td>Start()</td>
<td>Starts the sensor.</td>
</tr>
<tr>
<td>Stop()</td>
<td>Stops the sensor.</td>
</tr>
<tr>
<td>State</td>
<td>Returns the state of the sensor (Idle, Active, Error, ...)</td>
</tr>
<tr>
<td>Data event</td>
<td>Incoming data event.</td>
</tr>
<tr>
<td>Info event</td>
<td>Information event, for debugging purposes.</td>
</tr>
<tr>
<td>StateChanged event</td>
<td>Event indicating the network state changed.</td>
</tr>
</tbody>
</table>

5.2.2 Example usage scenarios

In the following are presented some example scenarios where the framework could be employed.

Multicamera surveillance systems

A typical surveillance system consists of multiple smart camera based nodes and a smaller set of monitoring nodes. The surveillance nodes receive image frames from their sensors, extract features from the images and track detected objects. They then inform each other and the monitoring nodes of detected and tracked objects. As an object leaves the node’s field of view, it communicates with other nodes over the network, passes to them object data and hands over the responsibility for tracking the object, if possible. The monitoring nodes can present this information to the operator of the system. No central point of control is in control of overall tracking, the nodes distribute the responsibility among themselves. An example of such a system is shown in Fig. 30a.

Nodes with multiple sensors or networks

Using the framework, nodes can use several sensors (Fig. 30b). One possible scenario is a set of nodes, each tasked to monitor a room. A node can use two or more different cameras or other sensors, and communicate with other nodes or a central monitoring node based on the information it collects from the sensors.
Fig 30. Examples of different usages.

Sensorless nodes

Not all nodes need to have a sensor attached (Fig. 30c). They can be bare processing nodes, only processing raw data sent to them by sensor nodes. Another possible scenario is a database node that can save and be queried about object features. User interfaces can also be implemented in this way.

Smart environments

Smart environments, where sensors interact and assist people in their activities, are also a good example of where the framework could be used. The networking interface can be used to connect the sensors, displays and user interaction devices together, while the sensors can take advantage of the sensor interface.
5.3 Experimental system

5.3.1 Current interface implementation

The .NET Framework was chosen as the implementation platform due to its increasingly widespread use, availability of tools and built-in support for networking, databases and other required resources. The user code is expected to run on PC workstations (for ease of development), which is a platform that the .NET Framework is well suited for. This environment lends itself well to rapid prototyping and development.

Network

As Saha & Mukherjee (2003) state, in smart environment network-level devices will require automatic configuration since current manual techniques for configuring a device with addresses, subnet masks, default gateways, and so on are too cumbersome and time-consuming. Furthermore, automated techniques to dynamically reconfigure the network when required are also crucial. We have therefore implemented a network module utilizing the Microsoft P2P Framework (Smith 2006), using PeerChannel (Microsoft Corporation 2008), the managed implementation in .NET 3.5. The module creates an internal communication endpoint for messaging with other nodes and provides the user with the interface for communication with peers.

In the PeerChannel implementation, the nodes are connected by a P2P mesh, allowing ad-hoc network formation and node dispersal. The mesh is formed using the Peer Name Resolution Protocol (PNRP) (Yang et al. 2006a), which in turn utilizes the Simple Service Discovery Protocol (SSDP) to find and connect to neighboring peers.

Communication between nodes takes place over a many-to-many broadcast channel, constructed using the point-to-point connections of the underlying mesh. Messages are propagated by flooding them to neighboring nodes. The nature of the peer mesh also allows message propagation to be restricted by limiting the number of hops a message can traverse. Several channels can be overlaid on the mesh and messages can be encrypted with a password or certificate using Transport Level Security (TLS).

Our network implementation consists of code to expose the PeerChannel functionality through our network interface. The network only provides a best effort service, messages are not guaranteed to reach all nodes. In addition, the mesh nature of the network leads to a transmission delay, which might not be acceptable in some user scenarios.
Sensor types

We have implemented sensor source modules for Axis IP cameras and directories of JPEG frames. The Axis cameras are accessed through the Axis Application Programming Interface (VAPIX®, Axis Communications (2011)). The camera unit provides a Hypertext Transfer Protocol (HTTP) Motion JPEG stream, that is parsed into individual frames that get passed on to the user. The camera modules are fully configurable, as shown in Fig. 28. They can also automatically recover from network outages, which can be frequent in a wireless environment.

5.3.2 Demonstration system

A demonstration system was implemented to test the suitability of the framework for a distributed multi-camera surveillance system (5.2.2). The system is composed of a set of processing nodes running on PC workstations, with six Axis 210A/213 IP-cameras acting as sensors. These are accessed through the sensor interface and are located as shown in Fig. 31. Four of the cameras are located along a central corridor, one in a lab room and one outside the building. Each camera is configured to produce frames of 320×240 pixels at a rate of 15 fps. These are passed to the user code as bitmaps, and an object detector is then used to extract human shapes from them. Each object is given a unique ID and tracked between successive frames to produce individual trajectories. Furthermore, the objects detected are also saved in a local database for possible future use.

The object detector used in the experiment is built on the cascade system proposed by Viola & Jones (2001) and improved by Lienhart & Maydt (2002). The actual implementation of the human detector is based on the software found in the Intel Open Computer Vision Library (OpenCV) (Bradski & Kaehler 2008, Willow Garage 2011), and the training samples are taken from the DaimlerChrysler Pedestrian Classification Benchmark Dataset (Munder & Gavrila 2006). In this case, the size of the samples the classifier has been trained on is 18×36. According to Munder & Gavrila (2006), this approach is suitable for real-time systems as it gives decent results at lower computational cost. A screenshot of the application running on the sensor nodes is shown in Fig. 32a.

The node communication takes place over a hybrid wired and wireless network. Every time a node detects an old or new object, it informs the other nodes through the network interface. Object data and features from the detector are sent to the peer nodes.
using XML, with an encoded thumbnail image of the detected object. This provides a greater amount of test traffic on the network than simply broadcasting the first and last time an object is seen. A sensorless monitor node is used to collect and visualize the data sent by the sensor nodes (Fig. 32b). In summary, it shows all the objects currently active and their trajectories overlaid on top of the floor plan.
5.3.3 Experimental results

The framework was tested by running a series of experiments using different node implementations and network configurations. The Axis camera module functioned steadily, with only temporary network outages causing loss of frames. The network module was tested by sending the object data acquired from the detector to other nodes over the peer mesh. Even though the PeerChannel protocol does not guarantee message delivery, the channel was found to be very reliable. In addition, the CPU burden caused by the processing intensive detection algorithm did not cause messages to be dropped with moderate messaging frequencies.

Fig. 33 shows a typical mesh formation sequence. The node mesh is usually not fully connected, but every node still has a route to any other node in the system, so the graph flooding algorithm can be used to deliver messages to all nodes. Any connections lost by nodes leaving the mesh are quickly rerouted around by the remaining nodes.

![Image of mesh formation sequence](image_url)

Fig 33. Example mesh formation sequence. (Saastamoinen et al. 2008, ©IEEE).
5.4 Discussion

This chapter presents an open and expandable framework that provides the user with easy access to sensors and communication channels between distributed nodes. Consequently, the framework allows the effort to be focused on the development of computer vision algorithms and their use in distributed proactive environments.

The advantages of distributed sensor networks over centralized systems are apparent. A central control node always leads to problems with scaling as the number of nodes increases. The proposed framework leads to more flexible and robust networks. Abstracting the differences between different sensors with a common interface allows code reuse, as in a best-case scenario only runtime configuration changes are needed as the sensor type is changed.

The framework was successfully used as a basis for the experimental system, and was found to offer the intended functionality. The nodes were able to automatically discover their neighbors and could be connected and disconnected from the network with minimal configuration effort. The current network implementation is built on peer-to-peer networking. The P2P structure of the network is resistant to network problems, and allows efficient formation of ad-hoc sensor networks.

In the future, we intend to develop the framework further by adding modules for different network implementations. On the sensor side, a DirectShow based module would allow image data to be retrieved from video files and local camera sources. Wrappers for C/C++ code are also of great interest, due to the amount of computer vision code already written in these languages. We have also released the framework code with an open source license for others to use.
6 Multi-object tracking

Tracking is an essential part of current proactive computer vision applications. Many modern visual surveillance and human computer interaction systems rely especially on reliable multi-object tracking. In this chapter, a method for multi-object tracking is presented. It also describes two different ways of extracting measurements about the location of objects in images, and testing of the tracking method is carried out using both of the approaches. This method can be plugged into a module that provides location and trajectories of multiple objects in proactive applications. In a surveillance scenario, for example, the targets are usually people, whereas in interactive applications targets can be the hands or the faces of individuals.

6.1 Introduction

One difficulty in the application of multi-object tracking involves the problem of getting reliable observations and associating them with the appropriate objects. The association process would be simple if there were only one measurement for each object, but in order to get reliable information about the position, the number of observations has to be larger.

This chapter presents a novel method, which utilizes soft assignment to associate the measurements to the objects tracked. Due to the soft assignment, the method is able to cope with inaccurate observations and inter-object occlusions. The method includes a component which combines the Kalman filtering algorithm (Kalman 1960) and the expectation maximization (EM) algorithm (Dempster et al. 1977) to estimate the parameters of the objects tracked and to assign the measurements softly. One of the benefits of this approach is also that neither iterations nor long measurement history are needed. The basic idea of the Kalman-EM algorithm was originally presented by Hannukela et al. (2007), and later it was extended to multi-object tracking (Huttunen & Heikkilä 2008, Huttunen & Heikkilä 2010).

The Kalman filter (Kalman 1960) is widely used in the context of tracking with noisy measurements and data association. In the multiple hypothesis tracking (MHT) algorithm (Reid 1979), target states are estimated from data-association hypotheses using the Kalman filter. For each measurement, probabilities are calculated for hypotheses that
the measurement came either from previously known targets or from a new target. The MHT algorithm is computationally exponential, both in memory and time. Later Joo & Chellappa (2007) have proposed an improved algorithm based on MHT. Another classical approach for data association is Joint Probabilistic Data Association Filter (JPDAF) (Fortmann et al. 1983), in which joint posterior association probabilities are computed for multiple targets or multiple discrete interfering sources in Poisson clutter. The major limitation of the JPDAF algorithm is its inability to initialize new objects entering the scene and to deal with objects exiting the scene.

6.2 Tracking algorithm

To give a general description, an overview of the algorithm is shown in Fig. 34. Our approach, like traditional Kalman filtering, can be divided into two phases. In the first phase, called the prediction phase or time update, the information learned in the past is used to further refine what the next locations of the objects will be. In the second phase, the correction phase or measurement update, measurements are made and then reconciled with the predictions based on our previous measurements. For a good introduction to Kalman filtering, the reader is referred to (Welch & Bishop 1995).

In the following, this section gives a detailed description of the tracking algorithm developed. First, we explore the system and measurement models of the tracking filter. Secondly, the time and measurement update equations, including the soft assignment of measurements are studied.

Fig 34. Overview of the combined Kalman filter and EM algorithm.
6.2.1 System model

As discussed in Section 2.3.2 of this thesis, an appropriate representation of a target should be selected in order to track the object. Here, we assume that each object \( j = 1, \ldots, M \) is represented by a vector \( x_j = [s_j, u_j, t_j, v_j]^T \) of four state variables which contain information about the object’s position \((s_j, t_j)\) and velocity \((u_j, v_j)\) in the \( X \) and \( Y \) directions respectively. The state-space model of the object \( j \) can therefore be formulated as

\[
\begin{align*}
x_j(k+1) &= \Phi x_j(k) + \Gamma \varepsilon_j(k), \\
\Phi &= \begin{bmatrix}
1 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 1 \\
0 & 0 & 0 & 1
\end{bmatrix}, \\
\Gamma &= \begin{bmatrix}
0.5 & 0 \\
1 & 0 \\
0 & 0.5 \\
0 & 1
\end{bmatrix},
\end{align*}
\]

where \( x_j(k) \) denotes the state of the object \( j \) at time step \( k \), \( \Phi \) is the state transition matrix, and \( \Gamma \) is the disturbance matrix. Finally, \( \varepsilon_j(k) \) is the process noise term, which is assumed to be zero-mean white Gaussian noise with a \( 2 \times 2 \) covariance matrix \( Q_j = \sigma^2_\varepsilon I \).

6.2.2 Measurement model

The observation \( i \) of the position \( l_i \) is assumed to follow the measurement model

\[
l_i(k) = H \sum_{j=1}^{M} \lambda_{i,j} x_j(k) + \eta_i(k), \\
H = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0
\end{bmatrix},
\]

where \( M \) is the number of objects being tracked, \( H \) is the measurement matrix, and \( \lambda_{i,j} \) is a hidden binary assignment variable which indicates the object that generated the measurement. In other works, if the first measurement \((i = 1)\) is coming from the first object \((j = 1)\) then \( \lambda_{1,1} = 1 \) and \( \lambda_{i,j} = 0 \) for \( i, j \neq 1 \). \( \eta_i(k) \) is the observation noise, which is assumed to obey zero-mean Gaussian distribution with a covariance matrix \( R \).

6.2.3 Soft assignment

It follows from equations (1) and (2) that the observations \( \{l_i\}_{i=1}^{N} \) form a set of 2-D points that follow a dynamically evolving Gaussian mixture model where the mean values of the components change during the course of time. Were the values of the
binary assignment variables $\lambda_{i,j}$ known beforehand, it would be possible to accomplish state estimation simply by using $M$ ordinary Kalman filters independently. Since this information is not available in general, we are compelled to estimate the assignments as well. For this purpose, we use Algorithm 1 which has been shown experimentally to converge on the mean values of the mixture components.

When applying the algorithm, the basic assumption is that there are $M$ distributions corresponding to different objects, and the location measurements $\{l_i\}_{i=1}^{N}$ are originating from them. Having the previous estimate of distribution parameters, we can evaluate a posteriori probabilities of the measurements to obtain the soft assignments $w_{i,j} \in [0, 1]$. In this work, the predicted estimates of object state $x_j(k)$ and state estimation uncertainty $P_j(k)$ in conjunction with a priori probabilities $\pi_j(k)$ of associating observations to the objects are used to compute soft assignments $w_{i,j}$ using the Bayesian formulation. It can be seen that this part corresponds to the “E step” of the EM algorithm.

Soft assignments are then used in computation of the Kalman gains $K_j(k)$ which are needed to get the filtered estimates of $x_j(k)$. Traditionally, one way of thinking about the weighting by $K_j(k)$ is that as the measurement error covariance $R$ approaches zero, the actual measurement is trusted more, while the predicted measurement is trusted less. On the other hand, as the a priori estimate error covariance $P^{-1}_j(k)$ approaches zero, the actual measurement will have a smaller effect on the estimate, while the predicted measurement is given more weight. However, in our approach, we have multiple measurements instead of a single measurement, and therefore we have to estimate the uncertainties for each of them. The principle is that the covariance matrix $R_{i,j}$, which represents the uncertainty of one measurement, is inversely proportional to $w_{i,j}$, and directly proportional to $R$. The small constant $\delta$ has the effect that the measurements not belonging to any objects, in other words, outliers, get very small weights and large uncertainty values and are therefore discarded. This part corresponds to the “M step” of the EM algorithm.

### 6.2.4 Implementation details

When using the filter in real world scenarios, the system noise characteristics are not exactly known. If too much emphasis were given to the dynamical model, the estimation would ignore the information from new measurements. It is even possible that this can lead to filtering instability and divergence. This can happen, for example, when an object is in the same position for a very long time. In order to alleviate the aforementioned
Algorithm 1 The combined Kalman filter and EM algorithm to estimate state using the
given model.

**Step 1.** Predict estimate \( \hat{x}_j^- (k) \) by applying dynamics (1)
\[
\hat{x}_j^- (k) = \Phi \hat{x}_j^- (k-1)
\] (3)
and predict error covariance \( P_j^- (k) \)
\[
P_j^- (k) = \Phi P_j^- (k-1) \Phi^T + \Gamma Q_j \Gamma^T, \quad f > 1.
\] (4)

**Step 2.** Compute the weights \( w_{i,j} \) for each position estimate \( l_i \) using a Bayesian formulation.
Let \( \pi_j (k) > 0 \) be the a priori probability of associating a measurement with the object
\( j (\sum_j \pi_j (k) = 1) \). The weight \( w_{i,j} \) is the a posteriori probability given by
\[
w_{i,j} = p( l_i | \mu_j (k), C_j (k)) \pi_j (k-1),
\] (5)
where the likelihood function \( p(\cdot) \) is a Gaussian pdf, with mean
\[
\mu_j (k) = H \hat{x}_j^- (k),
\] (6)
and covariance
\[
C_j (k) = H P_j^- (k) H^T + R.
\] (7)

**Step 3.** Use the weights \( w_{i,j} \) to set the observation noise covariance matrices in (2)
according to
\[
R_{i,j} = \frac{R_{i,j}}{w_{i,j} + \delta},
\] (8)
where \( \delta \) is a small positive constant to prevent a division by zero. Compute the Kalman gain
\[
K_j (k) = P_j^- (k) O^T (OP_j^- (k) O^T + R_j)^{-1},
\] (9)
where \( R_j \) is a block diagonal matrix composed of \( R_{i,j} \), and \( O = [HH \cdots H]^T \) is the corresponding \( 2N \times 4 \) observation matrix. Note that if \( w_{i,j} \) has a small value, the corresponding measurement is effectively discarded by this formulation.

**Step 4.** Compute filtered estimates of the state
\[
\hat{x}_j^+ (k) = \hat{x}_j^- (k) + K_j (k) (z(k) - O \hat{x}_j^- (k))
\] (10)
and compute the associated error covariance matrix
\[
P_j^+ (k) = (I - K_j (k) O) P_j^- (k),
\] (11)
where \( z(k) = [l_1 (k)^T, l_2 (k)^T, \ldots, l_N (k)^T]^T \).

**Step 5.** Update a priori probabilities for assignments with a recursive filter
\[
\pi_j (k) = a \pi_j (k-1) + (1 - a) \frac{1}{N} \sum_{i=1}^N w_{i,j},
\] (12)
where \( a < 1 \) is a learning rate constant.
problem, it is wise to introduce a constant fading factor $f$ into the proposed filtering solution (4) to keep it stable.

In the actual implementation of the filter, the measurement noise covariance is usually measured prior to operation of the filter. Estimating the measurement error covariance is possible because we should usually be able to take some off-line sample measurements in order to determine the variance of the measurement noise.

6.3 Obtaining measurements from binary masks

Using Algorithm 1 for multi-object tracking requires that we have some image measurements of the object locations. This and the following section elaborate two different approaches to accomplish this goal.

In this section, we assume that binary masks of the objects of interest have been extracted using some image segmentation technique such as background subtraction. As stated in Section 2.3.1, usually the masks cannot be extracted very accurately due to, for instance, image noise or varying illumination conditions. Thus the masks obtained are often heavily fragmented or incorrectly merged with other masks. In order to deal with this problem, we propose the following procedure. Firstly, a binary image that contains all the relevant masks in their current positions (see Fig. 35a) is composed. Then, the image is filtered using a series of morphological operations, and finally, the centroids of the remaining connected components that form the measurements $l_i$ are computed.

Both erosion and dilation operations are used, and they are applied successively several times in different orders. The number of times and the corresponding orders could be selected randomly. As a result, we obtain a set of measurements for each mask as shown in Fig. 35d. Using this kind of shrinking and growing technique, we are likely to always have "bad measurements" as in Fig. 35b, and "good measurements" as in Fig. 35c. However, Algorithm 1 can deal with both situations by giving no assignment to "bad measurements", as indicated with the colors in Fig. 35d.

6.3.1 Implementation based on color features

In order to use the algorithm presented to track objects in a video scene, the objects have to be separated from the background. There are several background subtraction or segmentation methods that could be used for this task. However, in this section we use a color feature based approach (Collins et al. 2005b) as an example to illustrate the
characteristics of the algorithm. Once the initial locations of the objects are known by user interaction or detection in the first frame, the features used for discriminating the objects from the background can be selected.

Provided with a set of seed features, log likelihood ratios of class conditional sample densities from the object and background are computed. These densities are needed to form a new set of candidate features tailored to the local object/background discrimination task. Finally, a two-class variance ratio is used to rank these new features according to how well they separate sample distributions of object and background pixels, and only the best one is chosen.

The set of seed candidate features is composed of linear combinations of camera R, G, B pixel values. As in the original method, in our system the integer weights for different color channels are between -2 and 2. By leaving out the coefficient groups which are linear combinations of each other, a pool of 49 features is left (Collins et al. 2005b).

The binary images needed for our filter are obtained from the log-likelihood ratio images $\log_j(x, y)$, which are generated for every object using the features selected. In the log-likelihood ratio image, object pixels contain positive values, whereas background pixels contain negative values (Collins et al. 2005b). It is therefore straightforward to calculate the binary images $B_j(x, y)$ by setting a threshold $t$. It is worth noting that it would be also possible to utilize those log-likelihood values in tracking. After the binary images have been formed, morphological operations can be applied to extract measurements as described in the previous subsection.
6.3.2 Experimental results

The proposed algorithm has been tested on several sequences from the CAVIAR database (CAVIAR project 2011) using color features to segment the objects. The tests were run on a regular PC using MATLAB. To demonstrate the feasibility of the concept described in this thesis, one test sequence is explained thoroughly.

The results of our algorithm are compared against a publicly available baseline color-tracking algorithm (Collins et al. 2005b) which uses the same color features as the proposed method. The implementation of the Variance Ratio method can be found at the VIVID Tracking Evaluation Web Site (Collins et al. 2005a). Initialization of the trackers was done using the ground truth information available in the dataset.

Fig. 36 shows the tracking results for the ExitEnterCrossingPaths1cor sequence. The positions of the objects are illustrated using red and green bounding boxes, and tracking of the objects starts from frame 1, as Fig. 37 depicts. The proposed method can successfully track two objects, even if they are occluded by a third person between frames 75 and 120. On the other hand, the Variance Ratio method loses track of the other object due to occlusion, and starts to follow the wrong person. The objects have a similar color appearance, and this causes problems for the Variance Ratio, which is a Mean Shift based algorithm.

![Fig 36. The tracking results of the proposed system (a)-(f) against the Variance Ratio tracker (g)-(l) for an ExitEnterCrossingPaths1cor sequence. The results are shown for frames 70, 80, 100, 110, 120, and 140. (Huttunen & Heikkilä 2008, ©IEEE).](image-url)
6.4 Using detector responses as measurements

One approach to address the limitation of reliable measurements is to combine tracking with detection. Previously, detection has been used to initialize new objects, or detectors are applied only to selected frames. Between the detections, tracking has been carried out using, for example, methods based on color or texture features. Due to increased computational power, it is nowadays possible to use detectors for every frame instead of just some frames. The tracking-by-detection approaches using detector responses directly as observations for tracking are, therefore, gaining more and more attention (Breitenstein et al. 2010).

In this section, a new method using detector responses as measurements is introduced. Unlike the approach presented in Section 6.3, this makes it possible to initiate new objects, as well as terminate tracks that are no longer valid. In addition, the proposed solution is able to adjust the scale of the objects tracked.

Since a typical object detector (Viola & Jones 2001) is insensitive to small changes in translation and scale, multiple detection responses will usually occur around each object in a scanned image, and typically it often makes sense to return one final detection per object. To obtain this kind of result, it is therefore useful to post process the detected sub-windows in order to combine overlapping detections into a single detection. However, it is unclear how fusing multiple overlapping detections to yield the final object detections should be performed. Unfortunately, it is also difficult to distinguish the false positives using post processing, and in some cases a detector can also output inaccurate responses. Thus, the output of the detector can be thought of as a series of noisy measurements, and therefore our approach uses the original detector responses as a set of measurements and assigns them to the objects currently tracked. In that way we can leave out the problematic post processing entirely, and at the same time get a
number of measurements for tracking.

One of the benefits of detector based tracking is that it enables us to track only the objects of an interesting category, for instance, humans or their faces. Another advantage is that we are not required to use a static camera, as is the case with multi-object tracking methods relying on background subtraction.

One way of integrating detection and tracking is to link detection responses in consecutive frames. Huang et al. (2008) present a detection-based three-level hierarchical association approach. More recent work by Singh et al. (2008) introduces a two-stage multi-object tracking approach using a pedestrian detector and association of track segments. Leibe et al. (2007) have introduced an approach which considers object detection and space-time trajectory estimation as a coupled optimization problem.

When comparing the aforementioned methods (Huang et al. 2008, Singh et al. 2008, Leibe et al. 2007) with this work, the biggest difference is that we associate the detector responses to objects without utilizing trajectory history. This means we do not need iterations or a long measurement history in order to track the objects. In addition, the tracking algorithm presented here is not dependent on a specific object detector or object category. Later in Section 6.4.6, we demonstrate the applicability of the approach for tracking multiple pedestrians and faces using a basic cascade detector.

The multi-object tracking method proposed is based on soft assignment of detector responses. For every frame, first an object detector is applied and the resulting output is passed to the actual tracking algorithm. If there are responses that are not assigned to any object, possibly new objects are initialized. On the other hand, if an object does not have any measurements, it might be necessary to stop tracking it.

### 6.4.1 Object detection

As reported in Section 2.3.1, there does exist a wide variety of detection methods. It is worth noting that the tracking algorithm presented here is not bound to any specific detector. The only requirement the used object detector has to meet is that it must be able to output the bounding boxes of the objects in a single frame.

The object detector used in this work is built on the cascade system proposed by Viola & Jones (2001) and improved by Lienhart & Maydt (2002). To recap, first a cascade of boosted classifiers working with Haar-like features is trained with a collection of sample views of a particular object and arbitrary images of the same size, called positive and negative examples, respectively. After a classifier is trained, it can be
applied to a region of interest in an input image, which has to be of the same size as that used during the training. Like any other detectors based on a binary object/non-object classifier, the detector scans the image with a detection window at all positions and scales, running the classifier in each window and yielding multiple overlapping detections.

6.4.2 Object initialization

When there is a number of detections which are not assigned to any objects tracked, it is very likely that there is at least one new object in the image. Detector responses that are left unassigned, and are overlapping with each other, form the bounding box of a new object candidate. In the current implementation, detections are combined in a very simple fashion. The set of unassigned detections is first partitioned into disjoint subsets. Two detections are in the same subset if their bounding regions overlap. Each partition yields a single final detection. A new object is created only if number of detections in the subset is greater than a predetermined threshold.

Dimensions of the new object are selected as the average of each of the corners of all detections in the overlapping set. A new object cannot be entirely initiated from a single measurement since it does not provide velocity information, and also false detector responses may cause problems. We are, therefore, using several frames in order to initialize an entirely new object.

6.4.3 Object scale

Since the method presented in Section 6.3 is based on color features, it does not provide any means of updating the dimensions of the objects tracked. In this approach, we are using detector responses and are able to update the dimensions \( \{d_j^w, d_j^h\} \) using the formula

\[
d_j^{[w,h]}(k) = a \cdot d_j^{[w,h]}(k-1) + (1 - a) \sum_{i=1}^{N} w_{i,j} \cdot r_i^{[w,h]},
\]

where the weights \( w_{i,j} \) are given by (5), and \( a \) is the learning rate used in (12). A detection response is denoted by \( r_i = [r_i^x, r_i^y, r_i^w, r_i^h]^T \), in which \( (r_i^x, r_i^y) \) are the pixel coordinates of the center of the bounding box, and \( (r_i^w, r_i^h) \) are the dimensions.
6.4.4 Track termination

When an object under tracking goes out of the camera view, the tracking algorithm must have a way of detecting it and removing the object. In our method, the criteria for terminating a trajectory is as follows. If no measurements are assigned to an object $j$ within a certain time, i.e. $\forall i, w_{i,j} = 0$, the object is considered to be lost and is therefore deleted.

6.4.5 Occlusion handling

There are two kinds of occlusions that can take place. The first case is occlusion due to a static obstacle, and the second alternative is that two or more tracked objects occlude each other. The proposed algorithm can handle both of these cases. The first case is taken care of straightforwardly, since the measurements are assigned softly to the occluded objects. In other words, the same measurements are shared between several objects. When the objects finally split, all of the objects are assigned different measurements. When an object goes, for example, behind a static obstacle, there will be no detections which could be used to update the position of the occluded object. It is, therefore, necessary to update the state of the object, based on the dynamic model (1). When the target reappears from behind the obstacle, there are going to be measurements available again, and tracking can continue normally.

6.4.6 Experimental results

The object detectors used in the experiment are built on the cascade system proposed by Viola & Jones (2001) and improved by Lienhart & Maydt (2002). The actual implementation of the detector is based on the software found in the OpenCV library (Willow Garage 2011).

Human tracking

For the experiments on human tracking, the training samples needed for the cascade detector are taken from the DaimlerChrysler Pedestrian Classification Benchmark Dataset (Munder & Gavrila 2006). The proposed algorithm has been tested on several sequences from the CAVIAR database (CAVIAR project 2011), and some results of the
test sequence ExitEnterCrossingPaths1cor are shown in Fig. 38. The proposed method can successfully track two objects, even if they are occluded by a third person.

Our own Axis_Busstop sequence has been captured using a PTZ camera that pans and zooms in to a person when he is walking by a bus stop. During the course of the sequence, the size of the objects changes significantly and turning the camera causes occasionally a large number of false measurements. Since there are also several objects that are tracked, the results indicate that the method is also able to track objects with a moving camera (Fig. 38).

**Face tracking**

To evaluate the usefulness of our method for tracking several faces, the method was tested in conjunction with a basic face detector. For face detection we used the face detector included in the OpenCV library (Willow Garage 2011) directly.

The face sequence motinas_multi_face_frontal (Maggio et al. 2007) used for testing is part of the AVSS2007 dataset\(^2\). The sequence in question includes many situations where four targets repeatedly occlude each other while appearing and disappearing from the field of view of the camera. The results show that the method is able to track the objects after a total occlusion (Fig. 38).

All the tests were run on a regular Pentium 4 2.8GHz desktop PC using MATLAB. Based on the studies on all test sets, the most computationally intensive part of the method is usually detection. Also computation of the Kalman gain (9) may take some time, depending on the number of measurements. However, based on the performance study of the current MATLAB implementation, we are confident that the method is feasible for different applications when implemented in C/C++.

### 6.5 Discussion

We have presented a new algorithm for tracking multiple objects. The method embeds the Kalman filter and Expectation Maximization (EM) algorithms in order to update the state of the objects and assign measurements to them. The algorithm is basically a recursive filter that allows observations to follow a mixture of the Gaussian model. The method is not computationally demanding, and it maintains multiple object hypotheses by maximizing a posteriori probabilities of the objects given the observations.

\(^2\)http://www.elec.qmul.ac.uk/staffinfo/andrea/avss2007_d.html
Fig 38. The tracking results of the proposed system for the ExitEnterCrossing-Paths1cor (rows 1-2), Axis_Busstop (rows 3-4), and motinas_multi_face_frontal (rows 5-6) sequences. Detector responses (top), and the final tracking results (bottom). (Huttunen & Heikkilä 2010, ©VISAPP).
Based on the experiments, computation of the Kalman gain (9) depends on the number of measurements, and in most cases takes most of the total processing time. In order to alleviate the problem, it would be possible to utilize the information filter (Grewal & Andrews 2001) where the estimated covariance and estimated state are replaced by the information matrix and information vector, respectively. The information filter form has the advantage that the update equations are computationally simpler than the equations for the Kalman Filter, at the cost of increased complexity in prediction. It is independent of the observations made, so the computational cost depends largely on the number of objects, which is usually smaller.

This thesis also describes a novel way of extracting measurements from binary masks using basic morphological operations. In order to use the algorithm presented in this chapter, the objects have to be separated from the background. The current implementation uses color features to extract the objects tracked from the background. One problem with this approach is that a separate initialization step is required at the moment.

In addition to binary mask measurements, we have introduced a solution for tracking multiple objects based on detector responses. The method employs the EM-Kalman filter in order to update the state of the objects and assign detector responses to them. It is worth noting that the method presented is not bound up with any certain static detector. Currently, a well-known cascade classifier is applied to detect humans or their faces.

Experimental results conducted clearly indicate the usefulness of the multi-object tracking approach proposed in this chapter. The EM-Kalman filter is able to operate on measurements originating from different kinds of sources, which makes it independent of any specific measurement method. In conclusion, in this chapter, we have covered the problem of getting reliable observations and associating them with the appropriate objects.
Conclusions

As the number of technical devices around us is increasing, also the way of interacting between humans and computers is changing. The systems of the future should be proactive so that they can adapt and adjust to people’s movements and actions without requiring any conscious control. Visual information plays a vital role in this kind of implicit human-computer interaction due to its expressiveness and unobtrusiveness. Combined with modern computer vision methods, proactive systems can detect and track people or other objects, personalize services by recognizing people, recognize gestures, text and other objects or events. Regardless its many advantages, use of computer vision is not always straightforward due to the wide variety of different applications and environments. The number of cameras involved in the system may vary from one camera in a mobile device to all the way to hundreds of cameras connected in wide area visual surveillance systems.

The focus of this thesis has been on developing new methods and systems that can be utilized in vision-based proactive applications. The topics discussed in this thesis cover different aspects and levels of the research area, and especially the applications represent opposite ends of the spectrum. As a case study, this thesis covered two different applications related to the research field. Firstly, an automated system that takes care of both the selection and switching of the video source in a distance education situation was presented. The camera views from the classroom are used to observe the teacher’s movements and possible document camera usage. The system is further extended with a pan-tilt-zoom camera system that is designed to track the teacher when s/he walks at the front of the classroom. In the future work, integration of the electronic material and other external video sources to the existing system should be considered.

The second proactive application, automatic shooting mode selection, was targeted on mobile devices. A system capable of automatic landscape mode selection has to be able to distinguish landscape from non-landscape scenes. Different approaches that can be used in automatic landscape scene recognition were therefore studied. Based on the experiments, a global texture-based approach competes with or outperforms other more complex methods in the landscape image recognition problem. Furthermore, the computational cost of the method relying on Local Binary Pattern representation is low enough for real-time systems. The application presented uses only visual information to
guide decision making. However, there are also other sensors that can provide us with useful data that could be analyzed in order to give additional hints.

The simultaneous use of several camera based sensor modules in the same environment to support a proactive application gives even a broader range of possibilities. Hence, distributed smart cameras and sensors have been an active area of research in recent years. For multi-camera applications it means that image processing migrates from central workstations to the distributed embedded sensors. On this field, the thesis presents an open and expendable framework for development of distributed sensor networks with an emphasis on peer-to-peer networking. The framework provides a user with easy access to sensors and communication channels between distributed nodes. Consequently, the framework allows the effort to be focused on the development of computer vision algorithms and their use in distributed proactive environments. The framework was successfully used as a basis for an experimental multi-camera surveillance system, and was found to offer the intended functionality. In the course of time, we intend to develop the framework further by adding modules for different sensor and network implementations.

From the methodological point of view, this thesis makes its contribution to the field of multi-object tracking. As a result, we can have location and trajectories of multiple objects which can be utilized in proactive applications. The method presented utilizes soft assignment to associate the measurements to the objects tracked. In addition, the thesis also introduced two different ways of extracting location measurements from images. The first approach extracts measurements from binary masks using basic morphological operations. In addition to binary mask measurements, a solution for tracking multiple objects based on detector responses was described. Experimental results verify the usefulness of the multi-object tracking approach proposed in this thesis. The approach based on the EM-Kalman filter is able to operate on measurements originating from different kinds of sources, which makes it independent of any specific measurement method. In the future, the aim is to utilize the method in our multi-camera surveillance framework.

As devices are becoming cheaper and faster all the time, more complicated and sophisticated methods can be developed and taken into use in real applications. It is, therefore, hard to predict what kind of applications will eventually appear, and which of them will find their way into commercial products. However, it is highly probable that every application still sets specific requirements for the methods that can be applied, and selecting the most suitable approaches to be used in each proactive
computer vision solution is going to require thorough testing and studying. Furthermore, it will be possible to embed sensors and processing units into a small unit, which can be seamlessly integrated into the environment. Serpanos & Papalambrou (2008) remind us that since the sensors are often deployed in public space or our personal environment, they increasingly access and manipulate sensitive or private information, which should be taken into account.
References


Cooperstock JR (2001) The classroom of the future: Enhancing education through augmented

119


Conference on Image Processing, 2640–2643.


Rabiner LR (1989) A tutorial on hidden markov models and selected applications in speech
Stanford V, Garofolo J, Galibert O, Michel M & Laprun C (2003) The nist smart space and


385. Kortelainen, Jukka (2011) EEG-based depth of anesthesia measurement: separating the effects of propofol and remifentanil
386. Turunen, Helka (2011) CO2-balance in the atmosphere and CO2-utilization: an engineering approach
387. Juha, Karjalainen (2011) Broadband single carrier multi-antenna communications with frequency domain turbo equalization
394. Pazvola, Marko (2011) An efficient entropy estimation approach
396. Koskela, Timo (2011) Community-centric mobile peer-to-peer services: performance evaluation and user studies
398. Teirikangas, Merja (2011) Advanced 0–3 ceramic polymer composites for high frequency applications
400. Goratti, Leonardo (2011) Design, analysis and simulations of medium access control protocols for high and low data rate applications

Book orders:
Granum: Virtual book store
http://granum.uta.fi/granum/
Sami Huttunen

METHODS AND SYSTEMS FOR VISION-BASED PROACTIVE APPLICATIONS

UNIVERSITY OF OULU
FACULTY OF TECHNOLOGY, DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INFOTECH OULU