



Technical Report
CVL-TR-6

# Detecting Falls in a Supportive Home Environment

Florian Dorn

Computer Vision Lab
Institute of Computer Aided Automation
Vienna University of Technology

September 15, 2011

**Abstract**

This thesis discusses the use of computer vision for assisting elderly in the field of Ambient Assisted Living with *Aging-in-place*. Falling has been identified as a major health issue for elderly, especially for those that live independently. Therefore an abnormal activity recognition system for detecting falls is presented. Activity or event recognition has been gaining much interest in the computer vision community in recent years. The application area covers a wide range, from video surveillance and monitoring, human-computer interaction to augmented reality. The fundamental problem lies within the detection and modeling of *Video events*, the semantic concepts that humans perceive when observing a scene. When emulating this process with computer vision, the semantic content of the low-level input has to be abstracted with meaningful features. Finding reliable models for describing and recognizing events given these abstractions is the key part in event understanding. In order to detect falls reliably a multi-camera vision system is proposed. The image evidence is fused early and fall detection is performed in 3D space. This allows the computation of reliable, view invariant features. Fuzzy logic is used to estimate the membership of the currently observed features to different human motion models. Using a novel feature, which is presented in this work, the unexpectedness of a fall incident is modeled reliably. The evaluation shows, that the proposed approach is a reliable and computationally efficient fall detector.

# Contents

# Chapter 1

# Introduction

Humans can perceive and understand semantic concepts when observing video sequences. Recently much effort has been made to offer solutions to this problem using computer vision and machine learning approaches. The recognition of events or *video event understanding* has a wide range of applications, including video surveillance and monitoring, human-computer interaction and augmented reality, or content based video databases. Applications such as "smart" surveillance, content-based video databases, gesture driven human computer interaction or motion analysis have already become available [TCSU08]. An event understanding framework abstracts the input image sequence into meaningful units. These are processed by the event model, which determines, if an event of interest occurred. The output is usually a decision whether a particular event occurred or a summary of the input [LRR09]. The overall video understanding process can be separated into two problems of abstraction and modeling:

**Event Abstraction** The formulation and computation of meaningful abstractions of the input.

**Event Modeling** Finding suitable formalisms to model events of interest and allow recognition of these events.

Video event understanding is generally considered the highest level image processing task, since it is based on a variety of lower level algorithms and systems that facilitate the recognition process. Among the events, actions, activities or gestures that have been investigated by means of computer vision are hand washing [MCB04], tennis strokes [YKI92], airport apron activities [FVB+07], sign language gestures and drinking actions in movies [LP07]. In monitoring applications recognizing *unusual* or *unexpected* events is of interest. Since these are events that differ from the expected behavior, they should be reported for further examination [ZSV04]. However, they are rare, hard to predict and generally difficult to describe, so recognizing them is not straight forward. Given a large number of observations of normal or expected behavior, the verification can be tackled with machine learning approaches. In this thesis, the unusual events are falls and we investigate computer vision based recognition of falls in the field of supportive homes.

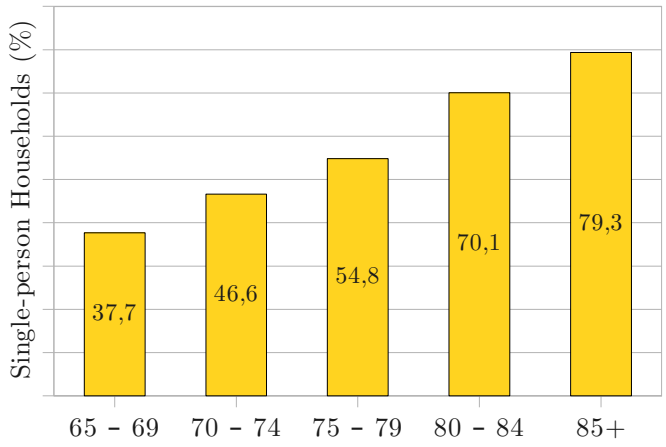



Figure 1.1: The percentage of Austrians aged 65 and above living in a single-person household (e.g.: unwed, divorced, married but living apart or widowed) as of 2006 [KE06].

## 1.1 The Need for Supportive Homes

Living and aging at home is the preferred lifestyle by many seniors [AF08]. In this context, the term *Aging-in-place* has been defined as "living where you have lived for many years, or to living in a non-healthcarehealth care environment, and using products, services and conveniences to enable you to not have to move as circumstances change" [AF08]. Smart home technology, so called "supportive homes", can be used to continuously monitor the well-being of "patients".

Currently, more than a third (35.81%) of the people living in a single-person household in Austria are aged 65 and above [KE06]. This makes up more than half (52.6%) of all the population aged 65 and above. As is illustrated in figure 1.1, the number of those living in a single-person household is rising with age: While 37.7% of the 65-79 years old live in a single-person household, this number steadily increases to 79.3% for those 85 years and above.

Falling has been identified as a health issue for the elderly, especially for those who live independently. Falls are the primary cause of the injuries for those 65 years and older, 63% of all injuries within this age group are related to falls, accounting for 15,802 deaths in the USA in the year 2005[1]. One third of the elderly fall once in a year, almost 50% of these falls are recurrent. Almost 10% of the falls result in serious injuries. Currently, an estimated 6% of the US health care expenses are related to elders recovering from a fall [RFW+98, CKN90, SCFM06]. Studies show that the risk of falling increases with age [GAA77, GLS+96]. It is reported, that the earlier a fall is reported and treated, the lower the mortality rate [GLS+96]. However, falls are not only a major threat for the physical health, but also reduce the independence of living even further due to the traumatic accident experience [NFR+07]. *"For elderly people who live alone, becoming incapacitated and unable to get help is a common event, which usually marks the end of*

[1]According to the Department of Health and Human Services, Centers for Disease Control and Prevention, Web-based Injury Statistics Query and Reporting System.

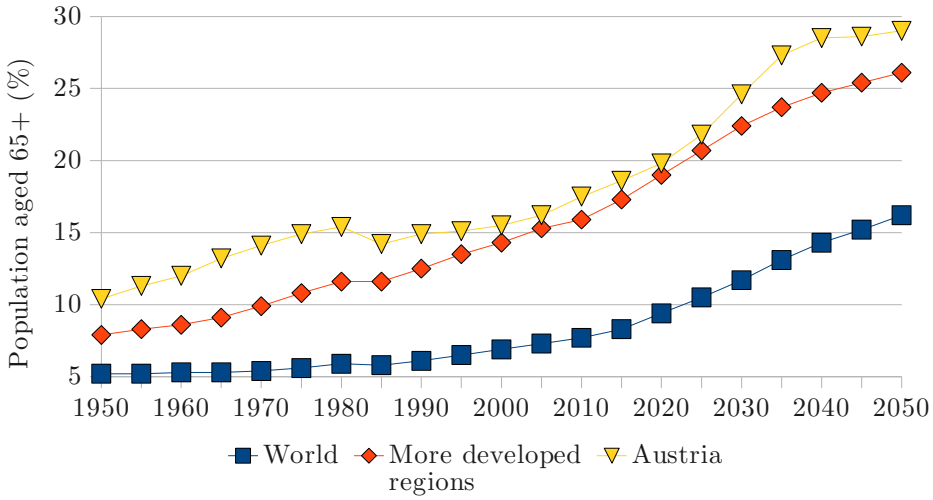



Figure 1.2: Growth of the population aged 65 and above from 1950 to 2050 [Zlo06].

*their ability to live independently* [GLS+96]."

### 1.1.1 Demographic Trends

Since 1950 the proportion of the elderly has risen steadily. The United Nations estimate, that the worldwide population aged 65 years and over will increase to 22% by 2050 [Zlo06]. For the more developed regions[2] a rise from 16% as of 2010 to 26% is estimated (Figure 1.2). In this period the potential support ratio[3] will decrease from to 4 to 2. While this indicates the steadily rising quality of the health care systems of the more developed countries, it implies more age-related diseases and raises questions on the long-term care for elderly. This development is seen as a great challenge for the health care systems in the developed countries [Com04, MER00].

A survey conducted in the UK in 1999 with 11,500 users involved, showed that the expected savings due to the introduction of smart homes for telecare would be around £7,100,000. Translating these results to the UK as a whole the savings would be around £7.7 billion over a 10-year period [BBB+99].

### 1.1.2 Acceptance and Perception

Ethical considerations are key issues that have to be taken into account in the design of smart homes, especially if computer vision is deployed. On the one hand, the users are monitored during their daily activities intruding their privacy; on the other hand this monitoring enables them to live an autonomous life in their own homes. In 2004 a study on the acceptance of smart home technology and sensoring devices has shown that elderly people accept vision based approaches for fall detection [DRA+04], however a major concern is that continuous monitoring is used "because technology can do it"

[2]All regions of Europe plus Northern America, Australia/New Zealand and Japan [Zlo06]
[3]The number of people age 15-64 per one person aged 65 or older

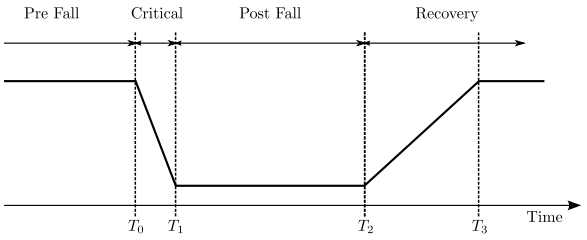



Figure 1.3: The four phases of a fall [NRB+08].

rather than out of a necessity. The questions on how to protect the patients privacy, how much information should be accessible and to whom this information should be accessible, are another major concern [CARP07].

The EU funded CONFIDENCE project[4] aims to build a care system for the detection of abnormal events. In the course of the project user requirements and acceptance were evaluated. While most of the participants engage in social and physical activities, 50% are afraid of falling and 20% are afraid of going out. A majority showed a positive attitude towards CONFIDENCE and would trust the system in a fall situation. However, privacy and dignity concerns are raised by 18 respondents.

## 1.2 Recognizing a Fall

Even though falls are experienced by everybody, is difficult to describe and thus detect a fall [NFR+07]. The World Health Organization defines a fall as "An event, which results in a person coming to rest inadvertently on the ground or other lower level" [PKS02].

In [NRB+08] it is proposed to partition the behavior into four categories when detecting falls: *prefall*, *critical*, *postfall* and *recovery* (see Figure 1.3). During the prefall phase, the person performs random activities of daily living (ADL). Sudden movements can occur, like when sitting down, getting up or lying down rapidly. The *critical* phase marks the actual fall event. It consists of a sudden movement towards the ground, and ends with a vertical shock. The duration of this phase is roughly 300–500ms. The time the person remains inactive, lying on the ground is the postfall phase. To reduce the sanitary consequences, this phase should be as short as possible. Last comes the recovery phase, where the victim gets up on his own or with the help from another person. Since the critical phase is relatively short and a highly dynamic process, time, speed and direction vary strongly [HHdW08]. Thus, fall detectors do not necessarily recognize the falls, but rather the postfall phase, where the patient is lying on the ground.

[4]CONFIDENCE: "Ubiquitous Care System to Support Independent Living". Project reference: FP7-ICT-214986. "The main objective of the CONFIDENCE project is the development and integration of innovative technologies to build a care system for the detection of abnormal events (such as falls) or unexpected behaviors that may be related to a health problem in elderly people." http://www.confidence-eu.org/

## 1.3 Thesis Objective and Contribution

This work presents an overview on video event understanding. The focus is on the recognition of unusual events, namely falls. Currently, wearable devices that have embedded accelerometers or gyroscopes are available on the market. However, it is commonly agreed upon, that they are insufficient for a number of reasons; they rely on user interaction and on the users capability and willingness to wear or use them. Those that restrain mobility are likely to be removed. Additionally, if alarms are not triggered automatically, especially severe falls, where the user is unconscious or might be unable to move, are not reported.

Vision based approaches have already shown promising results in laboratory setups. However, most use single cameras [TM06, RMSAR06, RMSAR07, FAP08, VMS07] and presume a strict camera placement [SJ04, FCLD08, HHdW08, YNC09], which is not possible under real world conditions. Anderson et al. [ALK+09] combine image evidence from multiple cameras, by performing a 3D reconstruction in voxel space.

Therefore a fall detection framework that uses multiple camera views is proposed. Multiple cameras increase the observation space and limit the effect of occlusions. To allow proper alignment of the views and fusion of the data, the cameras are calibrated. This gives better results than previously proposed approaches that rely only on a single uncalibrated camera. Further, in this setup the camera placement is not restricted, but cameras have to be mounted statically.

Detection of the critical and the postfall phases are attempted, to enhance the detection performance. Different features for event abstraction, as well as different event models are discussed and recognition results compared.

## 1.4 Structure of the thesis

A thorough investigation of the related fall recognition work is in Chapter 2 presented. First, user-activated and worn automatic devices, are presented in Section 2.1. Fall detectors based on acoustic information (Section 2.3) and floor vibration pattern (Section 2.4) are presented as well. In Section 2.5, computer vision fall detectors are presented in chronological order.

In Chapter 3 the proposed acquisition system and the laboratory setup are presented. The underlying camera model is introduced in Section 3.1. A refined model, which takes into account lens distortions, is presented in Section 3.2. Further an introduction to the state of the art in camera calibration is given in Section 3.3.

Chapter 4 discusses event abstraction methods. Silhouette extraction approaches are described in Section 4.1. The proposed early fusion of the image evidence in a global 3D voxel space is introduced in Section 4.2. Features, that describe shape and motion properties of the voxel representation are discussed in Section 4.3.

In Chapter 5, an introduction the Event Understanding terminology is given. This is followed by an introduction to the commonly employed modeling approaches: $k$-Nearest Neighbors (Section 5.1.1), Neural Networks (Section 5.1.2), Support Vector Machines (Section 5.1.3), Finite State Machines (Section 5.1.4), Hidden Markov Models (Section 5.1.5) and Fuzzy Inference (Section 5.1.6).

In Chapter 6 the proposed system and the experimental results are presented and discussed. Section 6.1 describes the system setup, the feature subset that was evaluated and the recognition pipeline. An evaluation of the proposed setup is given in 6.2 with a comparison to the state of the art.
Finally, Chapter 7 concludes the thesis, with a summarization of the contribution and the obtained results. Further an an outlook for future work in this area is given.

# Chapter 2

# Related Work

Different types of non computer vision related fall detectors have been commercially available or under active research for some time now. This section gives an introduction to these devices and presents the state of the art for computer vision based fall detection. These devices can be classified as one or a combination of the following four categories: User-Activated Alarms and Pendants (Section 2.1), Automatic Fall Detectors (Section 2.2), Acoustic Fall Detectors (Section 2.3) and Floor Vibration-Based Fall Detectors (Section 2.4). This is followed by a review of the relevant computer vision based approaches in Section 2.5.

## 2.1 User-Activated Alarms and Pendants

These devices generally require the user to press an alarm button in case of a fall. It is obvious that these systems are only suitable for cognitively intact persons and fail under certain situations, e.g.: if the person loses consciousness, or cannot reach the alarm button, due to trauma or pain. Such systems are available on the market [AF08]. An example is the Phillips Lifeline[1]: It consists of a stationary communicator device, that is connected to the telephone network, and a wearable personal help button (see Figure 2.1). On pressing the button a two way voice communication with the service center is established and the call will be handled immediately by an attendant of the service center. Based on the type of incident and the condition of the caller, a neighbor, a family member or emergency services are contacted.

## 2.2 Automatic Fall Detection Devices

A number of automatic wearable devices have been designed. They generally model the fall as an impact on the floor, followed by a near horizontal orientation of the faller [VBNL08, PFME06]. Accelerometers are employed to detect an impact and tilt sensors determine the orientation of the faller after an impact was detected. Fall detectors are

[1]Phillips Lifeline - the trusted medical alert service provider (accessed Dec. 2008) http://www.lifelinesys.com/


Figure 2.1: The Phillips Lifeline consists of a stationary device that is connected to the telephone network and a wearable personal help button.

usually worn around the chest, the waist or the thigh. A problem for the acceptance of this kind of fall detectors, if worn visibly, is the possible stigmatization of the users [AF08]. Furthermore, depending on the formalization of the fall, alarms my not be triggered if the fall does not occur on a horizontal floor [AF08].

Recently Perolle et. al. have presented a wearable wireless platform for fall detection that is additionally equipped with a GPS system to allow localization and enables bidirectional communication with a service center using GSM/GPRS [PFME06].

With SPEEDY, a first prototype of a fall detector integrated in a wrist watch is presented [DJRW03] (see Figure 2.2). The device is more comfortable to wear and discreet, which increases the acceptance of fall detectors among elderly. It can be conveniently worn around the clock in contrast to devices that are worn with a belt around the hip. A disadvantage is the greater complexity of the fall detection algorithm, as the arm has six degrees of freedom. This is visible in the overall detection performance of only 65%. However during the test no false alarm was reported.

A wearable airbag with an integrated automatic fall detector has been presented in [TYS+09]. In this study, acceleration as well as angular velocity features are used as input for a thresholding classifier.

In [ST09] a software, which utilizes GPS and accelerometers that are embedded in recent smart phones is presented. This allows a non-obtrusive device, which can provide location independent fall detection and alarm notification.

## 2.3 Acoustic Fall Detectors

An entire telemonitoring system based on unusual sound detection has been presented by [CVI+03]. The system analyzes the sound environment of an apartment in real time and detects abnormal sounds - falls of objects or the patient - that could indicate a distress situation in the living space, and calls for help. The system was designed, because

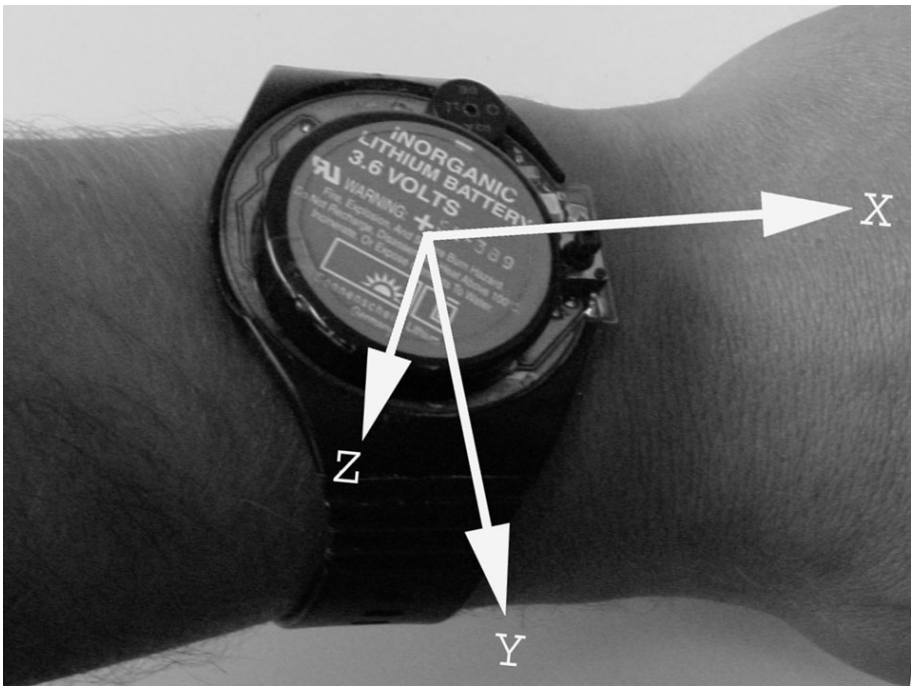



Figure 2.2: SPEEDY is an automatic fall detector embedded in a wrist watch [DJRW03].

"the elderly had difficulties in accepting the video camera monitoring, considering it a violation of their privacy" [CVI+03]. In an experimental setup, low cost, omni-directional microphones were installed in each room of the apartment. In a two step approach unusual sounds are recognized: in a first step relevant sound events are extracted and in the second step, sound events are classified in two groups, normal and abnormal. The event classification is transmitted to a master computer, which fuses the sound event data with data from medical sensors and sends alarm messages if necessary.

More recently an acoustic fall detection system that uses an array of acoustic sensors has been presented [PLSR08]. That way, the height of the sound-source is determined, thus reducing the false alarm rate.

## 2.4 Floor Vibration-Based Fall Detectors

A fall detector that directly measures the vibration has been presented in [ARK+06]. It is based on the observation, that human activities like walking, running cause measurable vibrations on the floor. The hypothesis is, that the vibration patterns resulting from falls are significantly different form those generated by normal daily activities or by objects falling on the floor. To measure and evaluate the floor vibration, a piezoelectric sensor coupled to the floor surface, by means of a mass and spring arrangement, combined with preprocessing electronics is used. The detector reports falls to a responder via a pager or a cellular phone. Controlled laboratory experiments with anthropomorphic dummies on mezzanine concrete floor and concrete slab floor showed a 100% detection rate. The detection range for the sensor was found to be 15 - 20 ft (4.5 to 6m). depending on the

floor type and the cover. Thus one device per room should be sufficient for most settings. To cover larger rooms, multiple devices can be installed. The installation procedure does not require significant customization by technical staff, as is generally the case with video monitoring-based fall detectors. Further experiments still have to be conducted, as different types of floors (e.g.: carpeted floors, parquet) alter the vibration-properties of floors drastically. Additionally the performance under realistic fall scenarios has to be evaluated as well.

## 2.5 Computer Vision

The traditional wearable devices presented above have various drawbacks. User activated alarms are not suitable in situations the faller becomes unconscious, immobilized or is mentally not in the shape to activate the alarm [RMSAR07]. Worn devices that use accelerometers can automatically detect falls, but are unintentionally often not worn, e.g. when returning home or if forgotten, because of dementia. If they are uncomfortable or cause alarms during housekeeping tasks, they are often removed. Moreover it has been pointed out, that these devices depend on the occupants willingness to wear such a device [Dou00, TM06]. Other systems based on acoustics or floor vibrations have shown promising results, but have not been thoroughly evaluated so far.

This led to the development of non-invasive vision based systems that operate automatically and do not constrain the occupant. A chronological presentation of the state of the art of vision based fall detection is given.

Nait-Charif and McKenna have shown that fall detection and activity summarization can be achieved with a single overhead-mounted camera per room [NCM04]. An adaptive background model with shadow detection is used to compute moving regions. Subsequently persons are modeled as ellipses and tracked with particle filtering. The authors claim that representing persons as ellipses yields a representation that is rich enough to allow detection relevant actions such as standing, sitting or falling, while it is coarse enough to allow tracking under various different body poses and clothing. The tracker provides trajectories in the 5D ellipse parameter space - $(x_t, y_t, \psi_t, s_t, e_t)$ with the center $(x_t, y_t)$ and orientation $\psi$, scale $s_t$ and eccentricity $e_t$ – which are used together with the persons speed $s_t$ (computed over a 40-frame temporal window) to provide a compact representation of the patients motion. It has been shown that context-specific spatial models can reduce the complexity of behavior interpretation greatly [MNC04, ZK10]. Two kinds of meaningful spatial regions are learnt from the motion trajectories: *Inactivity zones* such as a chair, a sofa, or a bed where it is usual, that little motion occurs for an extended amount of time. Entries and exits areas are labeled as *entry zones*. This was achieved using MAP estimation of Gaussian mixture models. Each Gaussian PDF, $p(\mathbf{x}_t \mid k)$ provides a model for the spatial extent of zone $k$. The activities in a room are semantically represented by temporally segmenting the sensor data into: time spent entering via an *entry zone*, inactivity in the *inactivity zones*, transition between the *inactivity zones*, and exiting the room. Figure 2.3 shows an observation with the motion trajectories and the inactivity zones marked. Fall detection works as follows: when the speed $s_t$ drops to an extent that indicates inactivity, the PDF's provide a way of checking whether the inactivity occurred

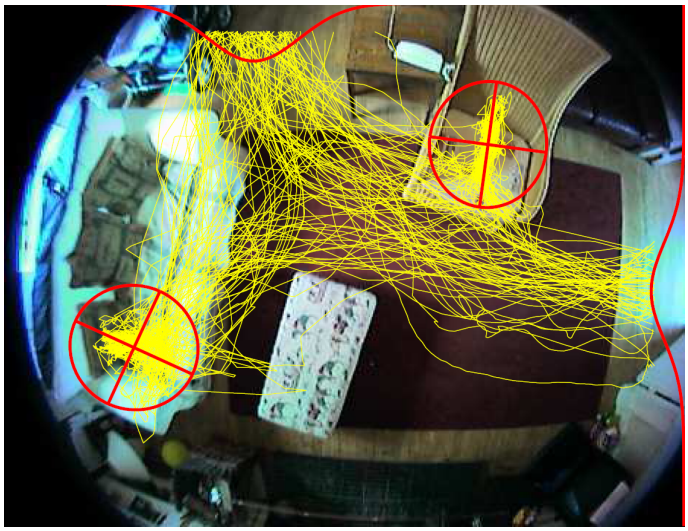


Figure 2.3: Top down view on a room, with motion trajectories (yellow) and inactivity zones marked as red circles. The entry zones are marked red on the left and top of the image. Image taken from [NCM04].

in a known inactivity zone or not.

An array of low-cost infrared detectors is used to detect falls in [SJ04]. Since interpreting the low-resolution infrared data can be implemented on small size, low-cost embedded hardware, the motion analysis occurs locally within the detector, in a single device. Object position, velocity shape and size are tracked with an elliptical-contour gradient-tracker. For fall detection, a neural network using vertical-velocity estimates provided by the tracked data as input is employed. Additional modules are employed to detect subtle motion and monitor inactivity. The output of the three modules is processed by a high-level reasoner, which triggers alarms based on long-term inactivity or falls. The performance of the fall detector was evaluated on laboratory data. The detector performed poorly with only 35.7% of the fall scenarios detected correctly. However, for the non-fall scenarios a 100% detection rate was achieved. The authors conclude that the vertical-velocity alone is not enough to discriminate falls from normal activity. However the postfall (i.e. the inactivity monitor) processing yields satisfying results.

In [TDc05] a combination of audio and video is used to detect falls with Hidden Markov models. In the image processing part, the width to height ratio $\rho(n) = \frac{w}{h}$ of the person's bounding box is transformed to the wavelet domain $\omega_i$. This 1-D signal is then used as the input for two three state Markov models for classification. They argue that using wavelet coefficients has 2 major advantages over directly using $\rho$: First, wavelets easily reveal the aperiodic characteristic of the fall. Second, setting thresholds is considered easier, since slow variations in the original signal lead to zero-mean wavelet signals. It is assumed, that while walking, $\omega$ is quasi periodic, while it rapidly converges to zero when falling and finally does show significant change. Two thresholds $T_1$, $T_2$ are introduced to formulate three states of Markov models, one for walking and one for falling:

$$\begin{array}{lcll} S_1 & : & |\omega_i| < T_1 & falling/after\,a\,fall \\ S_2 & : & T_1 < |\omega_i| < T_2 & in\,between\,(used\,for\,transitions) \\ S_3 & : & |\omega_i| > T_2 & walking,\,change\,in\,appearance \end{array} \tag{2.1}$$

In the walking model, since the signal is quasi periodic, the state transition probabilities are expected to be similar. While in the falling model, $S_1$ is expected to be dominant, while $S_2$ provides hysteresis and prevents sudden $S_1 \Leftrightarrow S_2$ changes.

In [TM06] the vertical angle of a person's principal axis $\theta$ is tracked as a 1-D signal over time. From the motion segmentation the minimal bounding rectangle is computed, which gives the orientation of the principal axis. Applying metric image rectification, the image is transformed such, that the horizontal image axis corresponds to the 3D Z-axis. Thus $\theta$ can be used as a reliable feature for fall detection. A two layer Hierarchical hidden Markov model (HHMM) is proposed to model and recognize activity. The first layer motion models are denoted as *elementary behavioral pattern*, and describe the corresponding observations. The following three pattern models are used: "Is Walking", "Is Falling" and "Is Lengthened". For the given observations, the most probable elementary behavioral pattern model is computed. On the second level the states correspond to the elementary patterns. Of the models WALK and FALL the one that best explains the sequence of elementary behavioral patterns detected is chosen. The usage of HHMM is motivated by their low computational cost and by additionally becoming tolerant to errors in the segmentation process.

The fall detection system presented in [RMSAR07] is designed as a low cost system, and works with an uncalibrated USB wide angle camera. It is based on three hierarchical verifications of the motion of the extracted persons blob and the ellipse approximating the blob. First a motion history image *MHI* ($H_\tau$) is used to quantify the blob motion $C_{motion}$. The MHI is a gray scale image, where the intensity of a pixel is a function of the temporal history of motion at this point. The more recent motion occurred, the brighter is the pixel:

$$C_{motion} = \frac{\sum_{Pixel(x,y)\in blob} H_\tau(x,y,t)}{\#pixels \in blob} \tag{2.2}$$

If large motion is detected ($C_{motion} > 65\%$) the approximated human shape is analyzed to distinguish normal motion from a fall. Two properties of the ellipse are examined: the angle $\theta$ between the ellipse's major axis and the horizontal axis $x$, as well as the ratio $\rho = \frac{a}{b}$ of the major axis $a$ and the minor axis $b$. Both are assumed to change significantly during a fall. This is measured by computing the standard deviation of $\theta$ and $\rho$ over a 1–second period. In the third step, a fall candidate is confirmed if a lack of motion of the ellipse is detected during a 5–second period. Non-motion is indicated by a low $C_{motion}$ and stable ellipse position and shape properties.

Lu¨trek et al. [LK09] have developed a fall detection prototype for the CONFIDENCE project, using a combination of 12 infrared body markers and multiple cameras to locate, and measure angles between body parts. From these locations three different sets of attributes are computed: the location in a reference world coordinate system, the location in a body coordinate system and the angles between adjacent body parts. Different machine learning algorithms and their performance on the different attribute sets have

been evaluated. Support Vector Machine (SVM) produced the most accurate results, yielding a classification accuracy of 96.5% on the combination of reference and angle attributes.

Hazelhoff et al. proposed an approach designed to handle real life situations [HHdW08]. Their method can handle inaccurate person segmentation caused by occlusions or due to additional objects such as walking aids. They achieve their goal using two uncalibrated perpendicular cameras. Using background subtraction and connected component analysis, persons are segmented in the image. Since two cameras are used, a tracker matches the detected objects and additionally discards non-human objects based on size constraints. The person's principal axis angle $\phi$ and the variance ratio $\rho$ are computed, and are used as feature input to determine a fall using a multi-frame Gaussian classifier. Since falling is a dynamic process involving varying time, speed and direction, the authors claim that it is more practical to identify lying persons. Manually defined inactivity zones are used to discriminate between intentionally lying poses and lying resulting from a fall. A Gaussian classifier is set up to discriminate between the two classes FALLS and NON-FALL. For each feature set (principal axis angle and variance ratio) the probability of a fall $p_{fall}(\rho, \phi)$ is computed and thresholded to obtain a binary classification result.

$$p_{fall}(\rho, \phi) = \frac{1}{\sqrt{2\pi}\,|\Sigma|} e^{-\frac{1}{2}\left(\begin{bmatrix} \phi - \mu_\phi \\ \rho - \mu_\rho \end{bmatrix}_{\Sigma^T} \begin{bmatrix} \phi - \mu_\phi \\ \rho - \mu_\rho \end{bmatrix}\right)} \tag{2.3}$$

To further reduce false positives, the position of the head is checked against the previous position and the fall rejected if the position is approximately the same.

Fu et al. [FCLD08] have presented an approach based on an Address-Event Temporal Contrast (ATC) Vision Sensor. An ATC sensor emulates the data–driven biological vision architecture [Lic06] as opposed to standard frame cameras. Instead of letting the receiver poll at a predefined frame rate, ATCs extracts changing pixels and reports temporal contrast as a sequence of events to a receiver [FCLD08]. The fall detection is based on centroid position and speed. Centroids are computed as the temporal averages of a series of event addresses. The centroid event address $(x_c, y_c)$ during a fixed period of $N$ frames is calculated as:

$$x_c = \left\lceil \frac{\sum_{i=1}^{N} x_i}{N} \right\rceil, y_c = \left\lceil \frac{\sum_{i=1}^{N} y_i}{N} \right\rceil \tag{2.4}$$

The event rate – the readout speed in the ATC – correlates with motion speed, size and light contrast. So when the lighting conditions are set, the motion in the scene can be derived from the event rate. Falls can cause 5120 events/s, while walking causes approximately 2100 events/s. The centroids vertical velocity during time period $T$ is given by:

$$V_y = \frac{\triangle y_c}{\triangle t} = \frac{(y_{c,i} - y_{c,,j})}{t_i - t_j} \tag{2.5}$$

with $\triangle t = t_i - t_j$. The self-contained non-intrusive detector is small in size and has low power consumption. Since no image data is obtained, the authors claim that privacy is protected. However the system relies strongly on the ATC sensors position mounted at a

height of $0.8m$. At that height, it is likely that objects such as chairs or tables obscure the sensors field of view. Since detection results have not been published, a detailed evaluation is not possible.

Anderson et al. [ALK+09] use four calibrated cameras for their proposed linguistic summarization approach. After motion detection they compute the volume the person is occupying in 3D voxel space – the "voxel person" – by back-projecting the silhouette images to the 3D space. For each frame, they compute the membership of the voxel person to a set of three predefined states (*upright*, *on-the-ground* and *in-between*) based on three features: centroid, eigen-based height and similarity of the major orientation with the ground plane. The state memberships are defined by fuzzy operators and have the following definitions:

**Upright** Voxel person has a large height, the centroid is at medium height and the primary orientation of the voxel person is similar to the ground-plane normal.

**On-the-ground** Voxel person has low height and low centroid. The primary orientation of the voxel person is dissimilar to the ground plane normal.

**In-between** Height and centroid are medium, and the primary orientation is non-identifiable or similar to the ground plane normal.

Activities are characterized according to the state duration, frequency of the state occurrence and the state transition behavior. Given an observation $\mathbf{o}$, a membership confidence value is assigned for each state at time $t$. Based on the fuzzy state memberships, a human readable linguistic summarization in the form of: $X_c$ is $S_i$ in $P_k$ for $T_j$ is generated. $X_c$ is the tracked person and $S_i$ is the observed state. The scene is partitioned into $K$ non-overlapping locations and $P_k$ denotes the location. $T_j$ is the duration, measured in $J$ fuzzy set definitions over the time domain. An example summarization would be "Person A is on-the-ground in the kitchen for a short time".

The authors argue, that while their proposed fuzzy logic based recognition requires domain expert knowledge for the formulation of the rules, fuzzy rules allow the recognition of activities as well as modeling of special cases. Their approach is flexible enough for rules to be added, removed or modified, which is extremely difficult with hidden Markov models.

Vishwakarma et al. [VMS07] employ a two state Finite state machine (FSM) in order to discriminate falls from normal activities. Three features are used as input: The width to height ratio of the object bounding-box, the object gradient and the fall angle.

In [FAP08] three different kinds of behavior are distinguished: normal, abnormal (stumbling and limping) and unusual (falling). This is achieved by a combination of five features that are computed from the motion segmented silhouette. The first two features are based on an approximated ellipse: standard deviation of the orientation (major axis) and standard deviation of major/minor axis length ratio. Normalized horizontal and vertical projection histograms as well as the frame to frame difference of the approximated head position are considered. A Multi-layer perceptron is used to differentiate the three behavior classes.

A simple thresholding approach is empowered by Yu et al. to detect falls [YNC09]. The vertical and horizontal head velocities are tracked with particle filtering and a first

order random walk model with additive Gaussian noise. Depending on the overall objects motion magnitude, the variance of the added noise is altered. Motion magnitude is obtained with MHI as in [RMSAR07].

Inspired by the work presented in [ALK+09], a comparison of early vs. late fusion has been presented in [ZMK10] as part of the MuBisA project[2]. As features for the fuzzy-based inference system, semantic driven features are chosen: bounding box aspect ratio, axis orientation, ellipse axis ratio and the motion speed. The evaluation showed that the early fusion of image evidence in a 3D space outperforms multiple independent fall detectors.

An overview of the presented acoustic and vision related literature as well as the vibration based approach by Alwan et al. is given in Table 2.2.

[2]MuBisA: "Computer Vision for an Independent Lifestyle of the Elderly and Disabled". Based on the idea of smart-homes, the aim of the MuBisA project is the development of a closed system for the automated event detection and the communication with mobile devices. In the project technical expertise of the state of the art computer vision is merged with the needs of well known consumer carriers in the field of assisted living. http://www.cogvis.at/mubisa/

| Author | Type | Detection | Abstraction | Model |
|---|---|---|---|---|
| Castelli [CVI+03] | audio | direct | sound events | GMM |
| Sixsmith [SJ04] | IR-vision | combined | vertical object velocity | Neural Network |
| Nai-Charif [NCM04] | vision | indirect | object centroid trajectory, centroid speed | GMM |
| Töreyin [TDc05] | vision + audio | indirect | wavelet transformed objects width to height ratio | HMM |
| Alwan [ARK+06] | vibration | direct | vibration pattern (frequency, amplitude, duration, succession) | integrated circuit |
| Thome [TM06] | vision (multi camera) | combined | object 3D principal angle | HHMM |
| Rougier [RMSAR07] | vision | combined | object motion quantification and principal angle | rules |
| Vishwakarma [VMS07] | vision | combined | object width to height ratio, object gradients, angle | FSM |
| Fu [FCLD08] | vision (ATC) | direct | vertical centroid velocity | NN |
| Popescu [PLSR08] | audio | direct | signal energy, sound height | K-NN |
| Hazelhoff [HHdW08] | vision | indirect | object principal angle, ratio of variance, head position | Gaussian classifier |
| Foroughi [FAP08] | vision | direct | standard deviation of orientation and angle, projection histogram, head position | MLP Neural Network |
| Anderson [ALK+09] | multi-camera vision | combined | centroid velocity, height and angle | fuzzy inference |
| Lustrek et al. [LK09] | IR-marker | combined | body angles | SVM |
| Yu [YNC09] | vision | direct | motion magnitude, 3D head velocities | threshold |
| Zambanini [ZMK10] | multi-camera vision | indirect | 3D bounding box ellipse aspect ratio, orientation, axis ratio and motion speed | fuzzy-based |

Table 2.2: An overview of automatic fall detectors, in chronological order.

# Chapter 3

# Acquisition System

In order to increase the observation volume and to reduce the effect of occlusions, image evidence from multiple cameras is used. In a laboratory setup, four cameras with partially overlapping views monitor a single room. At some point in the event recognition framework, the data derived from the multiple views has to be fused. Based on the results presented in [ZMK10] an approach is proposed, where the image evidence is fused early in the processing pipeline. In the suggested *early fusion* approach, the multiple views are combined to reconstruct a 3D voxel representation of the human. See Figure 3.1 for a comparison of early and late fusion. When working with multiple cameras, the data of the individual cameras has to be fused at some stage in the event recognition process. In late fusion approaches each camera performs data abstraction and event recognition individually. The recognition outputs of all cameras are fused late by a voting algorithm. With early fusion however, the camera data is fused early and data abstraction and event recognition are performed on the accumulated information provided by the different cameras.

Camera Calibration is the process of acquiring the internal and external parameters of the camera. With calibrated cameras a relation between the projection of the scene and the scene itself can be established. Starting with the pinhole camera model in Section 3.1 the mathematical fundamentals of the image acquisition process are described in this chapter. Since the pinhole camera model is an idealized camera model, this simplified model is extended in Section 3.2 to account for the distortions introduced when using lenses. In Section 3.3 an overview of calibration methods is presented, with references to further literature.

## 3.1 Camera model

The Pinhole Camera model describes the mapping of the coordinates of 3D points to the image plane of the camera through a perspective projection. In this model an ideal camera without lens is assumed. Instead the light enters the camera through an infinite small hole. Light rays reflected by objects pass through this pinhole and give an inverted projection of the object on the image plane $\pi$. The parameters of the camera can be categorized into *extrinsic* and *intrinsic* parameters.

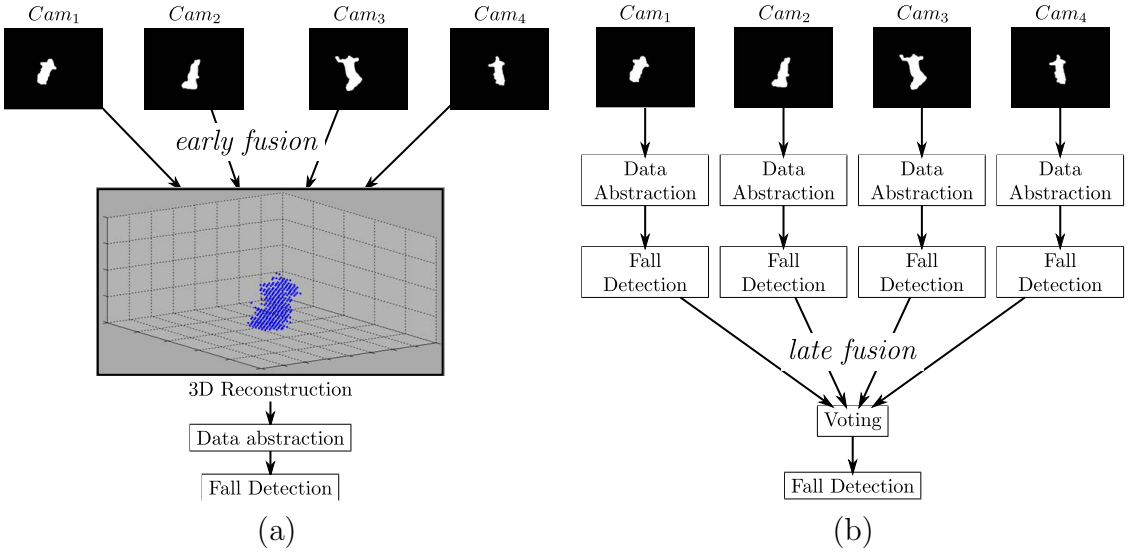



Figure 3.1: A comparison of early (a) and late fusion (b). [ZMK10].

The extrinsic camera parameters describe the location and orientation of the *camera coordinate system* in a known *world coordinate system* and are therefore required to transform from world coordinates to camera coordinates. To transform the world coordinate point $P_w$ at $(X_w, Y_w, Z_w)$ to $P_c$, $(X_c, Y_c, Z_c)$ in camera coordinates a translation followed by a rotation is applied:

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = R\,(P_w - t) \tag{3.1}$$

The vector $t = (t_x, t_y, t_z)$ translates the origin of the camera coordinate system to the origin of the world coordinate system. The $3 \times 3$ matrix $R$ expresses three elementary rotations – roll $\psi$, pitch $\varphi$ and yaw $\theta$ – of the coordinate axes along $x, y$ and $z$. This makes up six extrinsic camera parameters: three rotation and three translation parameters.

The intrinsic parameters define the projection of camera coordinates to pixel coordinates. Figure 3.2 depicts the pinhole camera model showing the *camera coordinate system* having origin $C$ and the axes $X, Y, Z$ and the *image coordinate system* with origin $c$ and the axes $x, y$. Capital letters denote points in the camera coordinate system, while small letters denote points in image coordinates. The pinhole $C$ is called *optical center* or *focal point*. The $Z$-axis is the *optical axis* and points away from the image plane $\pi$. The image plane is perpendicular to the optical axis, which intersects $\pi$ in the *principal point,* $c = (c_x, c_y)$. The distance $f$ of the focal point to the image plane is the *focal length*.

As is illustrated in Figure 3.2, the world coordinate point $M$ is projected along a ray through the optical center onto the image plane to the point $m$. As mentioned above the projection is inverted (rotated by 180°). Thus, a common simplification of the pinhole camera model is to introduce a virtual image plane in front of the optical center with $z = f$ [XZ96].

By similar triangles one can clearly see that the map of $M$ with coordinates $(X_c, Y_c, Z_c)$

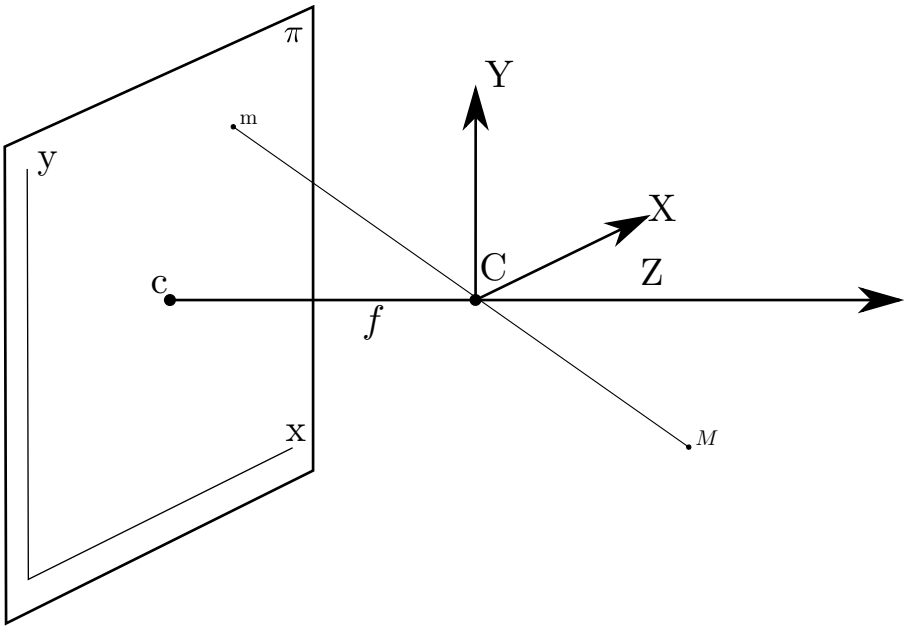



Figure 3.2: The pinhole camera model. The point $M$ in world coordinates is projected along a ray through the optical center $C$ onto the image plane $\pi$ as $m$.

to $m(x_i, y_i)$ in the *image coordinates* is given by:

$$x_i = \frac{f}{Z_c} X_c \text{ and } y_i = \frac{f}{Z_c} Y_c \tag{3.2}$$

Using homogeneous notation, affine transformations can be written as a matrix multiplication. The above transformation is written in homogeneous coordinates as:

$$\begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = \begin{bmatrix} \frac{f}{Z_c} & 0 & 0 & 0 \\ 0 & \frac{f}{Z_c} & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix} \tag{3.3}$$

Equation 3.2 assumes that the origin of the coordinates system in the image plane is at the principal point. Since we are dealing with digital images, we are using discrete positive coordinates, the *pixel coordinate*s. The associated projection from camera coordinates $(x_c, y_c)$ to pixel coordinates $(u, v)$ shifts the coordinate system to have it's origin at $(0, 0)$ and incorporates the size of sensoring elements $s_x, s_y$:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{bmatrix} \frac{1}{s_x} & 0 & c_x & 0 \\ 0 & \frac{1}{s_y} & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{pmatrix} x_c \\ y_c \\ 1 \end{pmatrix} \tag{3.4}$$

Combining the mappings (3.4) and (3.3) we have the *camera calibration matrix* or

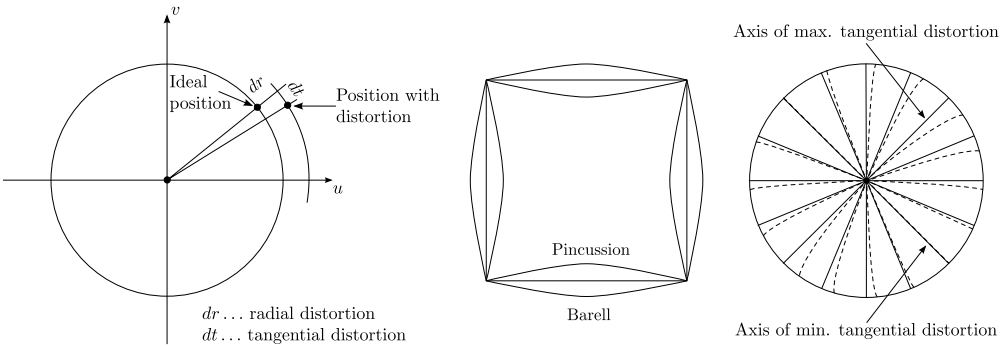



Figure 3.3: The general principle of radial and tangential distortions (left), the effect of radial distortion (center) and of tangential distortion (right).

*matrix of intrinsic parameters*:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \underbrace{\begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{camera\,calibration\,matrix} \cdot \begin{pmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{pmatrix} \tag{3.5}$$

where $f_x = \frac{f}{z_c s_x}$, $f_y = \frac{f}{z_c s_y}$.

## 3.2 Lens Distortions

The pinhole model is just an approximation of the projection when using a lens. It is useful, as it allows us to formulate the relationship between world and image coordinates in a simple way. However, it is practically not valid, since any lens introduces distortions. Most notably are radial and tangential distortions [HS97]. However depending on the quality of the lens they can be minimized. Figure 3.3 illustrates the effects of radial and tangential distortions.

With radial distortion, the location of pixels near the edge of the image are distorted more that at the center, where there is zero distortion. It is approximated by the first few terms of a Taylor series:

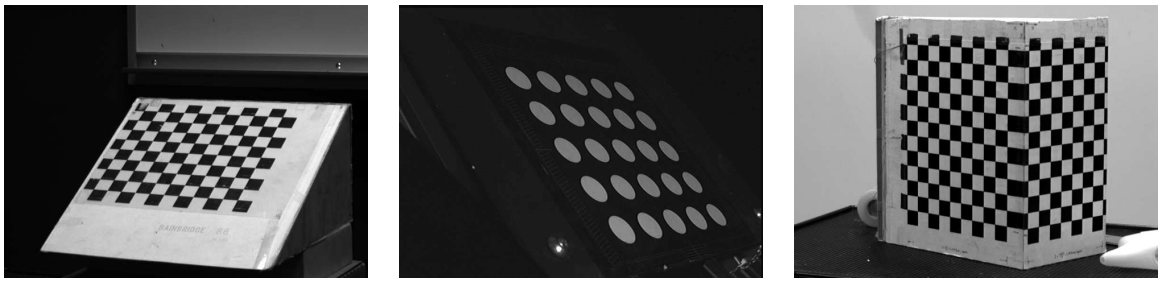
$$\begin{pmatrix} \delta x_c^r \\ \delta y_c^r \end{pmatrix} = \begin{pmatrix} x_c(\kappa_1 r^2 + \kappa_2 r^4 + \dots) \\ y_c(\kappa_1 r^2 + \kappa_2 r^4 + \dots) \end{pmatrix} \tag{3.6}$$

With $\kappa_1, \kappa_2, \dots$ the radial distortion coefficients and $r = \sqrt{x_c^2 + y_c^2}$.

The tangential distortion is formulated as:

$$\begin{pmatrix} \delta x_c^r \\ \delta y_c^r \end{pmatrix} = \begin{pmatrix} 2p_1 x_c y_c + p_2(r^2 + 2x_c^2) \\ p_1(r^2 + 2y_c^2) + 2p_2 x_c y_c \end{pmatrix} \tag{3.7}$$

where $p_1, p_2$ are the tangential distortion coefficients. In most cases it is sufficient to only estimate the first and second order radial distortions $\kappa_1$, $\kappa_2$. The tangential distortion is



(a) (b) (c)

Figure 3.4: Three examples of calibration images, two planar and one 3D target: (a) with a checkerboard pattern [Bou99], (b) with a circular pattern [ZS04] and (c) a 3D target [Bou99].

generally only applied for fisheye and wide angle lenses [Tsa86, Zha00]. Other types of distortion have been proposed in the literature, but depending on the physical properties of the lens, their effects are usually little, or can be approximated by radial and tangential distortions [WCH92].

## 3.3 Camera Calibration

Combining the pinhole camera model with the equations for radial and tangential distortion and the external parameters, the full camera model is specified. The rotation matrix can be expressed as three separate rotations, thus there are the following parameters:

- Extrinsic
  - $t_x$, $t_y$, $t_z$ - the components of the translation vector $t$
  - $\psi$, $\varphi$, $\theta$ - the three elementary rotations - roll, pitch and yaw
  - $f_x$, $f_y$ - focal length in $x$ and $y$ direction
- Intrinsic
  - $c_x$, $c_y$ - the image center
  - $\kappa_1$, $\kappa_2$ - the radial distortion coefficients
  - $p_1$, $p_2$ - the tangential distortion coefficients

The process of estimating the best set of model parameters is called camera calibration. Typically images of a calibration target – an object with known geometric properties – are acquired. The set of parameters, that best match the estimated projection with the observed projection are estimated [Bou99]. Heikkila [HS97] uses a cubic 3D calibration object with a dot pattern, while others use flat targets with checkerboard or circular patterns [WCH92, Bou99] that are easier to handle. Figure 3.4 shows examples of calibration targets: two planar and one 3D target.

In [ZS04] the quasi standard approaches of Tsai [Tsa86], Heikkila [HS97] and Zhang[Zha00] are compared, with the result, that all three approaches provide feasible results for close range photogrammetry, however practicability is determined by the modeled camera parameters. The selection of an appropriate approach depends on the desired accuracy and the quality of the lens system.

In [Tsa86] Tsai presents a method that estimates the parameters in a semi-linear way. This is achieved by reducing the distortion parameters to the first order radial distortion parameter $\kappa_1$. This simplification is sufficient to determine the extrinsic parameters (except for $t_x$) uniquely in a first step, without providing an initial guess. In the second step the $f$, $t_z$ and $\kappa_1$ are estimated by non-linear optimization.

Heikkila and Olli's [HS97] method is based on the direct linear transform (DLT). DLT is based on the pinhole camera model, ignoring the nonlinear distortion parameters. In the first step the linear transformation from object coordinates $(X_i, Y_i, Z_i)$ to image coordinates $(u_i, v_i)$ is solved. The non linear parameters are estimated using least squares method of the differences of the computed image coordinates and the measured coordinates. This requires initial guesses that are provided by the parameters of the DLT.

The approach proposed by Zhang [Zha00] requires at least two views on a planar target. Zhang assumes that $Z = 0$ for the calibration target and that the world coordinate system is aligned with the axes of the calibration target. With the points in image coordinates $p_i = (u_i, v_i)$ and the corresponding known points in 3D world coordinates $P_i = (X_i, Y_i, Z_i = 0)$ a homography can be derived. The homography is solved with a closed form solution and refined with a Maximum likelihood estimation inference. Distortion parameters are estimated by extending the maximum likelihood equation with the parameters for radial and tangential distortion.

Recently, Svoboda et al. [SMP05] have presented a fast multi-camera calibration procedure, which uses a virtual calibration target. This solves the two major problems of traditional approaches when calibrating multiple cameras: First, calibrating a multi camera system involves a lot of manual work: From placing the calibration target in each camera view, to registering all views to one common world coordinate system. Second, with the size of the working volume, the calibration target size has to grow as well. By moving a laser pointer in the working volume, a virtual 3D calibration target is created over time. Their approach is aimed at multi camera systems and requires at least three roughly synchronized cameras.

Grammatikopoulos et al. [GKP07] have presented an automatic approach for camera calibration from vanishing points of scenes that satisfy the Manhattan world assumption. This assumption states, that the image scene contains three orthogonal, dominant directions, which is usually satisfied in indoor and urban images [CY99]. While other approaches [DIM02] often only estimate the external camera parameters and focal length, their approach is able to estimate the camera constant, location of principal point, and the two radial distortion coefficients.

Zhang's calibration method efficiently supports the camera model described in section 3.1 with the distortions introduced in Section 3.2. While a short overview has been given in this section, a detailed description of the calibration procedures is out of the scope of this thesis. The reader may refer to [Zha00] for details on the closed form and maximum likelihood estimation equations.

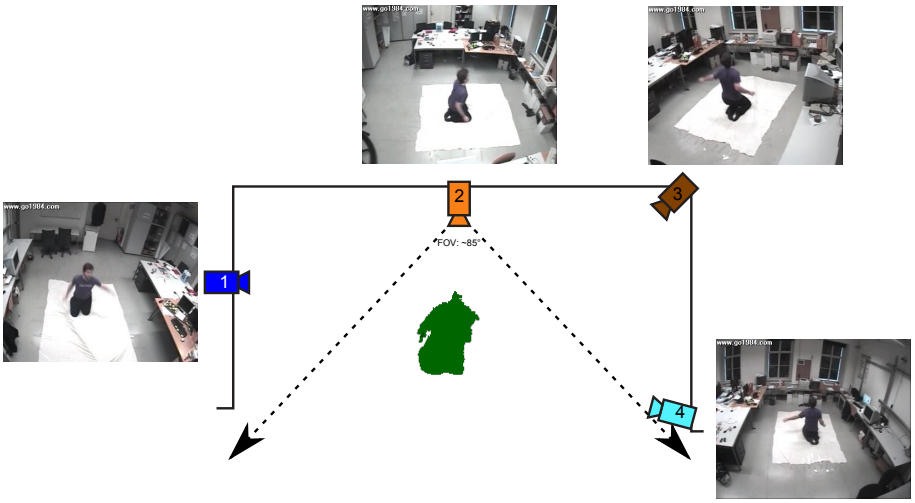



Figure 3.5: Camera Placement.

## 3.4 System Setup

Using 4 calibrated cameras, the working volume for the proposed early fusion approach covers approximately $6 \times 4.5 \times 2m$. The camera placement is shown in Figure 3.5. The cameras are placed at approximately $2m$ height. Calibration was performed using the approach suggested by Bouguet [Bou99], with respect to a common world coordinate system, having its origin roughly in the center of the observation area.

# Chapter 4

# Data Abstraction

Image sequences consist of massive amounts of raw information in the form of spatio-temporal pixel intensity variations [TCSU08]. Most of this information however is irrelevant for the recognition of motion, as was shown in an experiment by Johansson [Joh76]. Observers were able to identify human motion patterns just by observing light sources placed on 5-10 limb joints, without any other contextual information. In this chapter data abstraction methods are discussed. These abstractions are commonly referred to as *features*, the set of features at time $t$ makes up the *feature vector*. Pixel level and object level features are examined. Pixel level abstraction are those that rely on single pixel or group of pixel features such as color information, texture, edges, gradients and so forth. Object based abstraction on the other hand is founded on the assumption that a description of the objects participating in scene is a reasonable intermediate representation scheme [LRR09]. It builds on a meaningful grouping of pixels into objects and their properties, including size, shape, trajectory, speed, etc. Object based abstraction builds on previous object segmentation and tracking approaches.

In the following sections, various features, with an emphasis on object based features are presented. Since silhouettes are the foundation of the later proposed object based features, the state of the art is presented in Section 4.1. Shape from Silhouette, which is a method for reconstructing the 3D Volume of objects based on silhouette images from different viewpoints is presented in Section 4.2. In Section 4.3 various object based features as well as a novel unexpectedness feature are presented.

## 4.1 Silhouette Detection

To extract the objects of interest (the foreground) a *background subtraction* is usually applied. Background subtraction is based on the assumption, that there is a relatively static background and a moving foreground. The current image at time $t$ is subtracted from the background and thresholded to separate foreground $F$ and background $B$.

Another approach for estimating pixel based motion in image sequences are based on the estimation motion vectors. Horn [HS81] describes optical flow as the distribution of apparent velocities of movement of brightness patterns in an image. Velocities are assigned to each pixel in the frame, which describe the motion from the previous frame. This forms

a dense motion field, where the discontinuities resemble image segments from different objects. One major advantage of optical flow approaches over background subtraction is that they inherently handle camera motion.

Since we propose static cameras, background modeling can be applied. However background subtraction and optical flow approaches have to be able to handle a variety of common scenarios. These typical problems have been summarized in [TKBM99]:

**Unclean initialization** In uncontrolled scenes, a clean view on the background (with no foreground objects) is not possible.

**Varying illumination** During observation the illumination conditions can usually change. This is caused by the changes of the time of day (solar irradiation), cloudy weather conditions or when lights are switched on or off.

**Moved object** Background object are generally not static. Such moving background introduces an additional "ghost", located at the position of the object while it was part of the background.

**Waving trees** Background objects can show high frequent changes. Examples are waving trees or curtains.

**Shadows** Due to lightning conditions, foreground objects cast shadows, which move along with the object.

**Foreground Aperture** Uniformly colored objects show observable motion only at their boundaries.

### 4.1.1 Color Mean and Variance

The underlying assumption of the Color Mean and Variance (CMV) approach is that the background can be modeled by a single Gaussian distribution [WADP97]. Considering standard RGB color space, each pixel is modeled per channel by it's the mean $\mu_R, \mu_G, \mu_B$ and variance $\sigma_R$, $\sigma_G$, $\sigma_B$. A newly observed pixel $o$ is classified as foreground if:

$$|o_c - \mu_c| > \alpha\sigma_c \text{ for } c \in \{\text{R,G,B}\} \tag{4.1}$$

where $\alpha$ controls the sensitivity of the segmentation. Considering the Normal distribution, $99,73\%$ of the background is covered for an $\alpha = 3$. The background model is initialized from the first $N$-frames, which ideally only show the static background.

$$\mu_c = \frac{1}{N}\sum_{t=1}^{N} o_c(t) \tag{4.2}$$

$$\sigma_c = \sqrt{\frac{1}{N}\sum_{t=1}^{N} o_c(t) - \mu_c^2} \tag{4.3}$$

Using a simple adaptive filter the background model is updated after the classification:

$$\begin{aligned} \mu_t &= \lambda o_t + (1-\lambda)\mu_{t-1} \\ \sigma_t &= \lambda \left| o_{t-1} - \mu_{t-1} \right| + (1-\lambda)\sigma_{t-1} \end{aligned} \tag{4.4}$$

The learning rate $\lambda$ is different for foreground and background pixels. An extension to a multi-modal Gaussian mixture model is presented in the next section.

### 4.1.2 Gaussian Mixture Model

Grimson et al. have proposed an adaptive background modeling and maintenance algorithm that is widely used for long term[1] observation scenarios [GLRS98, SG99].

Based on the idea that backgrounds are dynamic as well, and thus at different times background pixels can represent multiple objects, possibly under different lightning conditions, the background is modeled by a mixture of $K$ Gaussian distributions for each pixel[2]. The normal distributions $\eta$ are specified by the mean $\mu$ and the covariance $\Sigma$. The probability of observing the value $X$ for a pixel at time $t$ is given by:

$$P(X_t) = \sum_{k=1}^{K} \omega_{k,t} * \eta(X_t, \mu_{k,t}, \Sigma_{k,t}) \tag{4.5}$$

where $\omega_{k,t}$ is an estimate of the weight (what portion of the data is accounted for by this Gaussian). To match a pixel value against the model, the $K$ distributions are first ordered by their *fitness:*

$$fitness = \omega_k / \sigma_k \tag{4.6}$$

which increases as the distribution gains more evidence and the variance decreases. This puts the most likely distributions on top and less probable ones on the bottom, where they are eventually replaced by new distributions. Of these, the first $B$ distributions constitute the current background model, where

$$B = argmin_b \left( \sum_{j=1}^{b} w_j > T \right). \tag{4.7}$$

$T$ represents the minimum proportion of pixel data that should be accounted for by the background. Every new pixel $X_t$ is tested for membership in the Gaussian distributions. A match $M_{i,t}$ of $X_t$ in $i^{th}$ distribution is defined as:

$$M_{i,t} = \begin{cases} 1 & if \ \left| X_t - \mu_{i,t} \right| < \alpha \sigma_{i,t} \\ 0 & \text{otherwise} \end{cases} \tag{4.8}$$

As before, $\alpha$ controls the sensitivity and is usually a value in the range of $2-3$ [SG99]. If a match is found within the first $B$ distributions, $X_t$ is part of the background. If none of

[1] In [SG99] continuous observation over a 16 month period has been reported.

[2] Typically the value of $K$ is 3 to 5

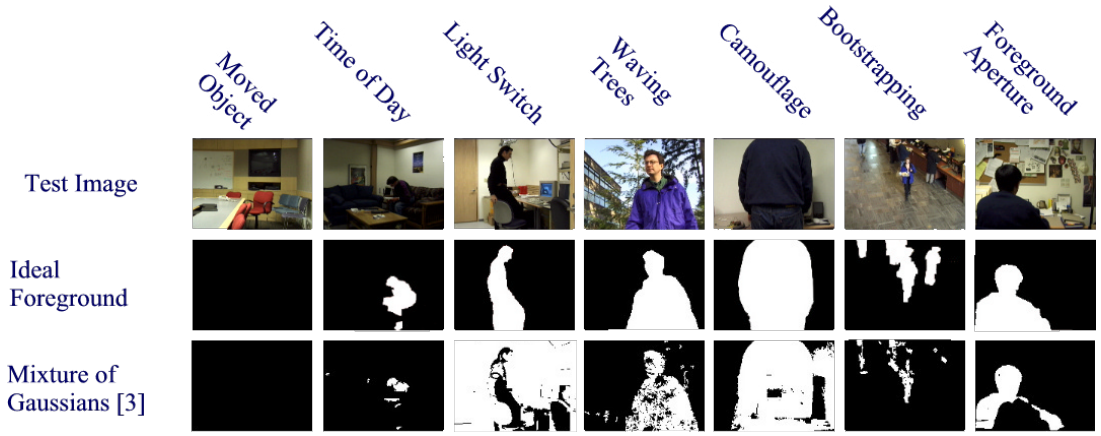



Figure 4.1: A comparison of the Gaussian Mixture Model and the ground truth[TKBM99].

the distributions match, the least probable distribution is replaced with $X_t$ as the mean, an initially high variance and low weight. After classification, the model is updated as follows:

$$\begin{aligned} \omega_{i,t} &= (1-\alpha)\omega_{i,t-1} + \alpha(M_{i,t}) \\ \mu_{i,t} &= (1-\rho)\mu_{i,t-1} + \rho X_t \\ \sigma^2_{i,t} &= (1-\rho)\sigma^2_{i,t-1} + \rho(X_t - \mu_{i,t})^T(X_t - \mu_{i,t})^T \end{aligned} \tag{4.9}$$

where $\rho = \alpha\eta(X_T\,|\mu_k, \sigma_k)$ is the learning rate for mean and variance and $\alpha$ is the learning rate for weights. For distributions that match the new observation, $\omega$,µ and $\sigma$ are adjusted, while only the weight is updated for unmatched distributions.

The Gaussian Mixture Model is robust enough to handle most of the problems mentioned above: Since it is adaptive and continuously maintains the background, unclean initialization and gradual illumination changes are handled. Sudden illumination changes are not handled properly. The waving tree problem is successfully addressed, since multiple background distributions are maintained. Foreground objects become motionless are integrated in the background, without destroying the original distributions. Thus, if this object moves again, distributions that describe the previous background are still valid and will quickly be re-incorporated. Since multiple distributions are independently maintained for each pixel, GMM is both memory demanding and computationally complex. Thus one has to either reduce the number of distributions, or decrease the image resolution. Figure 4.1 shows the results of the Gaussian Mixture Model in comparison to an ideal foreground.

### 4.1.3 Codebook model

With the codebook model, Kim et al. [KCHD05] recently presented a performant, non-statistical clustering technique for background modeling. It is designed for long term usage, and can model mixed backgrounds by using multiple codewords. For each pixel a codebook $C$, consisting of multiple codewords $c_i$, $i = 1 \ldots L$ is maintained. Each codeword consist of a color vector $\mathbf{v}_i = (R_i, G_i, B_i)$ and the 6-tuple $\mathbf{aux}_i = \left\langle \check{I}_i, \hat{I}_i, f, \lambda_i, p_i, q_i \right\rangle$ containing the brightness values and temporal variables of the codeword:

$I_i^{min}, I_i^{max}$ minimum and maximum brightness of all pixels assigns to this codeword

$f$ is the frequency with which the codeword has occurred

$\lambda$ the maximum negative run-length MNRL defined as the longest period this codeword has not occurred

$p, q$ the first and last access times, that the codeword has occurred

The brightness is defined as $I = \sqrt{R^2 + G^2 + B^2}$. Detecting the foreground follows a straight forward algorithm. The distance from the current observation $\mathbf{o}$ to the nearest codeword is computed and compared to a threshold. If no codeword matches the observation, the pixel is marked as foreground.

1. For all codewords $c_i$, find the first codeword that satisfies:
   (a) $colordist\left(\mathbf{o}, \mathbf{c}_i\right) < \epsilon_2$
   (b) $brightness\left(I, \left(I_i^{min}, I_i^{max}\right)\right) = true$

2. $BGS(\mathbf{o}) = \begin{cases} FG & \text{if there is no match} \\ BG & \text{otherwise} \end{cases}$
   A matched codeword $c_m$ is updated as follows:

$$\begin{aligned} \mathbf{v}_m &\leftarrow \left(\frac{f_m \bar{R}_m + R}{f_m + 1}, \frac{f_m \bar{G}_m + G}{f_m + 1}, \frac{f_m \bar{B}_m + B}{f_m + 1}\right) && (4.10) \\ \mathbf{aux}_m &\leftarrow \left\langle min\left(I, I^{min}\right), max\left(I, I^{max}\right), f_m + 1, \right. \\ &\quad \left. max\left(\lambda_m, t - q_m\right), p_m, t \right\rangle && (4.11) \end{aligned}$$

The two conditions (a) and (b) match the observation with a codeword based on color and brightness similarity, respectively. Observing that pixel colors change over time and under varying lighting conditions, and that this change is mostly distributed in an elongated shape along the axis towards $(0, 0, 0)$ a color model was developed. The principle idea is, that background pixel values lie along the axis of the codeword, with low and high bounds for brightness. Having an input pixel $\mathbf{o}_t = (R, G, B)$ and the codeword $c_i$ with $\mathbf{v}_i = \left(\bar{R}_i, \bar{G}_i, \bar{B}_i\right)$, the color distortion $colordist$ is measured as:

$$colordist\left(\mathbf{o}_t, \mathbf{v}_i\right) = \sqrt{\left\|\mathbf{o}_t\right\|^2 - \frac{\left\langle \mathbf{o}_t, \mathbf{v}_i \right\rangle^2}{\left\|\mathbf{v}_i\right\|^2}} \tag{4.12}$$

Where

$$\begin{aligned} \left\|\mathbf{o}_t\right\|^2 &= R^2 + G^2 + B^2 && (4.13) \\ \left\|\mathbf{v}_t\right\|^2 &= \bar{R}_i^2 + \bar{G}_i^2 + \bar{B}^2{}_i && (4.14) \\ \left\langle \mathbf{o}_t, \mathbf{v}_i \right\rangle^2 &= \left(\bar{R}_i R + \bar{G}_i G + \bar{B}_i B\right)^2 && (4.15) \end{aligned}$$

As will be discussed in greater detail in section 4.1.4, shadows cause changes in brightness, but leave the color relatively unchanged. To account for changes in brightness due to shadows, minimum and maximum brightness of the codeword is stored in the **aux** tuple and compared to the observed value:

$$brightness\left(I, \left(I_i^{min}, I_i^{max}\right)\right) = \begin{cases} true & \text{if} I_{low} <= \|\mathbf{o}_t\| <= I_{hi} \\ false & \text{otherwise} \end{cases} \tag{4.16}$$

with $I_{low} = \alpha I_{max}$ and $I_{hi} = \left\{\beta I_{max}, \frac{I_{min}}{\alpha}\right\}$.

Construction of the initial codebook for $N$ learning frames, for a single pixel $x$ follows a straight forward algorithm:

1. $\mathbf{x}_t = (R, G, B)\,,\ I = \sqrt{R^2 + G^2 + B^2}$
2. Find the first codeword $c_m$ matching $x_t$ based on
   - $colordist\left(\mathbf{o}, \mathbf{c}_i\right) < \epsilon_1$
   - $brightness\left(I, \langle I_i^{min}, I_i^{max}\rangle\right) = true$
3. If there is a match, then update the codeword as in 4.10
4. If there is no match, then create a new codeword $c_L$ with $L \leftarrow L + 1$:
   - $\mathbf{v}_L \leftarrow (R, G, B)$
   - $\mathbf{aux}_L \leftarrow \langle I, I, 1, t-1, t, t\rangle$

For each codeword $c_i$ wrap around $\lambda_i$ by setting

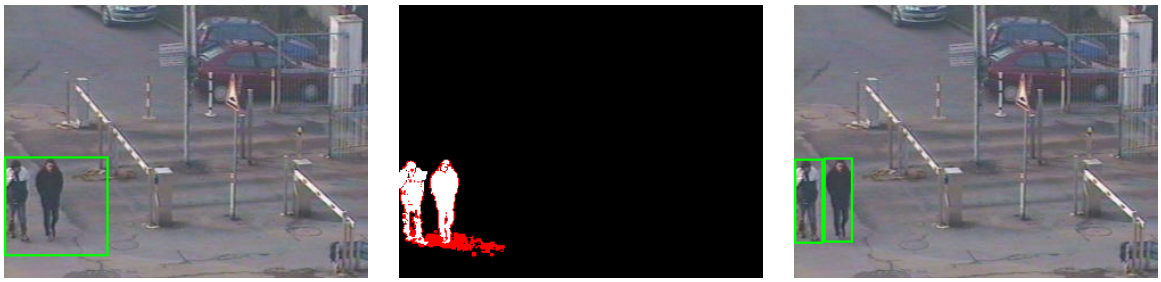
$$\lambda_i \leftarrow max\left\{\lambda_i, (N - q_i + p_i - 1)\right\} \tag{4.17}$$

To allow codebook generation with foreground objects present, only codewords present in at least half of the frames are preserved, thus stale entries in the codebook are removed.

### 4.1.4 Shadow Removal

Practically every scene, indoor or outdoor contains shadows. After background subtraction, we obtained an image, where the color values differ from the reference background. However this does not necessarily reflect a change of the foreground. Shadows cast by foreground objects also satisfy this restriction and are not of interest. The effect of shadows being detected as foreground that can lead from small shape distortions to merged objects are illustrated in Figure 4.2.

In [RE95] interpretation of shadows and their effects on the image pixels have been defined as: *"a semi-transparent region in which the scene reflectance undergoes a local attenuation"*. Thus it is assumed that a shadow reduces the luminance of pixel while the chromaticity is preserved. The effects of highlights can be described similar: the chromaticity is preserved, while the luminance is increased. Concise recognition of shadows



(a) (b) (c)

Figure 4.2: Improved segmentation results after shadow detection is applied. Two persons are tracked as one object since shadows are detected as foreground (a). The output of the segmentation with shadow detection applied is shown in (b). White pixels mark foreground, red shadows. The correct tracker output is shown in (c). Images from [**CGP+01**].

is a difficult task, which requires knowledge of the scene geometry, the materials and the properties of the light sources. This information however is generally not present and considering dynamic scenes is not obtainable. Salvador [Sal04] has presented several empirically deducted cues for the presence of shadows.

1. A material in shadow appears darker than the same material not in shadow.
2. The change of chroma due to a shadow is predictable.
3. Surface texture tend to continue across a shadow boundary
4. Shadows cast by objects moving with respect to a fixed light source move across the scene.
5. The motion of a shadow-casting object that moves relative to a fixed light source and that of its shadow are correlated.

Other cues have been described, however they require higher level processing and are out of scope.

These observations have been successfully utilized for shadow removal by other authors [WADP97, HHD99, CGP+01, BWHK06, NBT08]. Following the definitions given by Horprasert et al. [HHD99] each pixel can be classified as one of the four categories Background $B$, Shaded background $S$, Highlighted background $H$ and Foreground $F$. The following basic rules have been laid out to classify a pixel $P_x$ with the corresponding background model $BM_x$:

**Background** if it's brightness and chromaticity are similar to $BM_x$.

**Shaded background** if it has similar chromaticity but lower brightness than $BM_x$.

**Highlighted background** if it has similar chromaticity but higher brightness than $BM_x$.

**Foreground** if the chromaticity is different from $BM_x$.

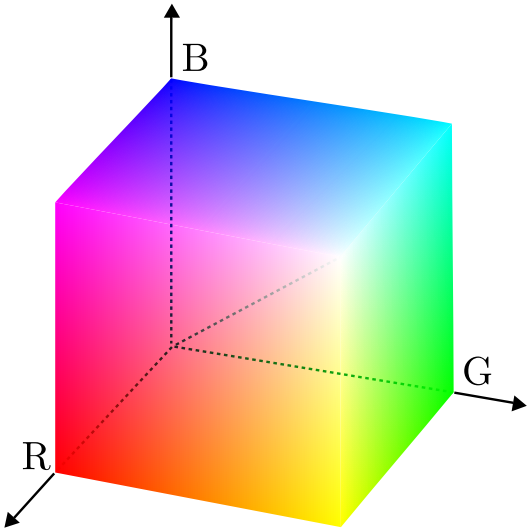



Figure 4.3: Sketch of the RGB color cube. Gray levels are on the main diagonal $(0,0,0) - (1,1,1)$

In [MJD+00] the first-order gradient is additionally evaluated to support the shadow classification. In [NBT08] texture information is exploited, based on the idea that within a shadow region, adjacent pixels show the same intensity reduction ratio.

With the above given rules, the question arises on how to ideally represent color in the background model. A color model or color space is a mathematical model, defining a way color is modeled as a vector of numbers, typically with 3 or 4 components. A multitude of different color spaces exists, each designed for specific domains. There is *RGB*, which is commonly used in display (CRT or LCD monitors) and retrieval units (cameras or scanners). It is an additive color model, that is composed of three the primaries red, green and blue. *CMYK* is a subtractive color model used for color printing. In analog video *YUV* or similar color spaces (*YCbCr*, *YPbPr*) are standard. They are not composed of primary colors, but have a separate luminance channel *Y* and two chromatic channels *UV*, *CbCr*, *PbPr*. This more closely corresponds to the way humans perceive color. The *CIELAB*, or just *Lab* is designed in a way, that the Euclidean distance corresponds to the perceived distance of colors. It is obtained from RGB data using *CIE XYZ* as an intermediate space. However for this conversion, the white point has to be estimated, which limits its practical use [Han08].

### RGB Color Space

Digital images are usually processed in the RGB color model as it is the natural representation of color for digital display and retrieval systems. The three primaries red, green and blue are in the range of $[0,1]$ and form a 3D Cartesian coordinate system. It is often represented as the RGB-color cube which is shown in Figure 4.3.

Because all three channels encode chromaticity and luminance, RGB is inadequate for shadow detection. This led to the development of the normalized RGB color space, that aims to separate the brightness from the chromaticity components [BWHK06]. It is

Figure 4.4: An Image from the PETS'2001 dataset in normalized RGB: The RGB image (a), luminance $l$ (b) and the chromatic components $r$ (c) and $g$ (d). The sensitivity of $r$ and $g$ to noise is visible in the dark areas top-left (bushes) and bottom-right (car shadow).

has been applied to background modeling and shadow detection [MJD+00, HW03]. The conversion of the components $R$, $G$, $B$ to their normalized counterparts $r$, $g$, $b$ and the luminance $l$ is defined as:

$$l = R + G + B,\ r = \frac{R}{l},\ g = \frac{G}{l},\ b = \frac{B}{l} \tag{4.18}$$

if $l \neq 0$ otherwise $r = g = b = 0$. Since $b = 1 - (r + g)$ a pixel is satisfactory described in nRGB by the brightness component $l$ and two chromatic components $r$ and $g$. The conversion from RGB to nRGB is computationally inexpensive [NBT08], what makes it a popular choice in real time applications. However, the chromatic components are frail to sensor noise or compression artifacts, in areas where the luminance is low [BWHK06]. Figure 4.4 illustrates this issue. Additionally, the dominant wavelength is only improperly represented in RGB and normalized RGB, thus two chromaticities, with different dominant wavelengths, could be considered [NBT08].

**Shadow Detection**

In normalized RGB, a pixel is considered foreground if:$|o_c - \mu_c| > \alpha\sigma_c$ for any channel $\in \{r, g, l\}$, with the observed value $o_c$, background model mean $\mu_c$and standard deviation $\sigma_c$, and foreground threshold $\alpha$. Foreground pixels are classified as shadow if:

$$o_l < \mu_l \quad \wedge \quad o_l > \beta\mu_l \tag{4.19}$$

$$|o_r - \mu_r| < \tau_c \quad \wedge \quad |o_g - \mu_g| < \tau_c \tag{4.20}$$

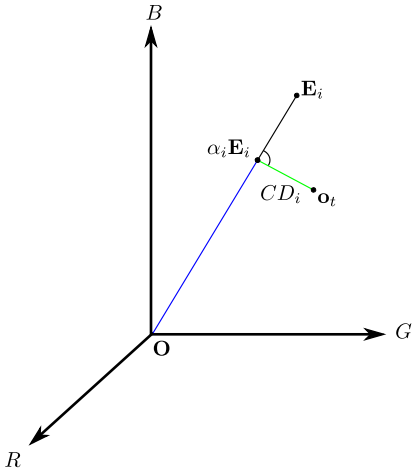



Figure 4.5: Illustration of the RGB color model presented in [HHD99].

Horprasert et al. [HHD99] have presented a CMV background model based in the RGB color space, attempting to separate the chromatic components from the brightness component. The codebook model introduced in section 4.1.3 is based on that model. Figure 4.5 illustrates this idea. Let $\mathbf{E}_i = \langle \mu_R(i), \mu_G(i), \mu_B(i) \rangle$ denote the $i^{\text{th}}$pixel's expected color in the background model, $\mathbf{o}_t$ is the current pixel and $\overline{\mathbf{OE}}_i$ is referred to as the expected chromaticity line. The variation from $\mathbf{o}_t$ and $\mathbf{E}_i$ is decomposed into the brightness distortion $\alpha_i$ and the chromatic distortion $CD_i$. Geometrically, $\alpha_i\mathbf{E}_i$ is the intersection of $\overline{\mathbf{OE}}_i$ and the plane defined by $\mathbf{o}_t$ and the normal $\overline{\mathbf{OE}}_i$. Thus $CD_i$ is the shortest distance from $\mathbf{o}_t$ to $\overline{\mathbf{OE}}_i$. In the background model a pixel is modeled as the four-tuple $\langle \mathbf{E}_i, \mathbf{s}_i, a_i, b_i \rangle$, $\mathbf{E}_i$ is the vector of RGB means, $\mathbf{s}_i = \langle \sigma_R(i), \sigma_G(i), \sigma_B(i) \rangle$ is the standard deviation vector, $a_i$ and $b_i$ are the quadratic means of the brightness distortion and chromatic distortion respectively:

$$a_i = RMS(\alpha_i) = \sqrt{\frac{\sum_{i=1}^{N} (\alpha_i - 1)^2}{N}} \tag{4.21}$$

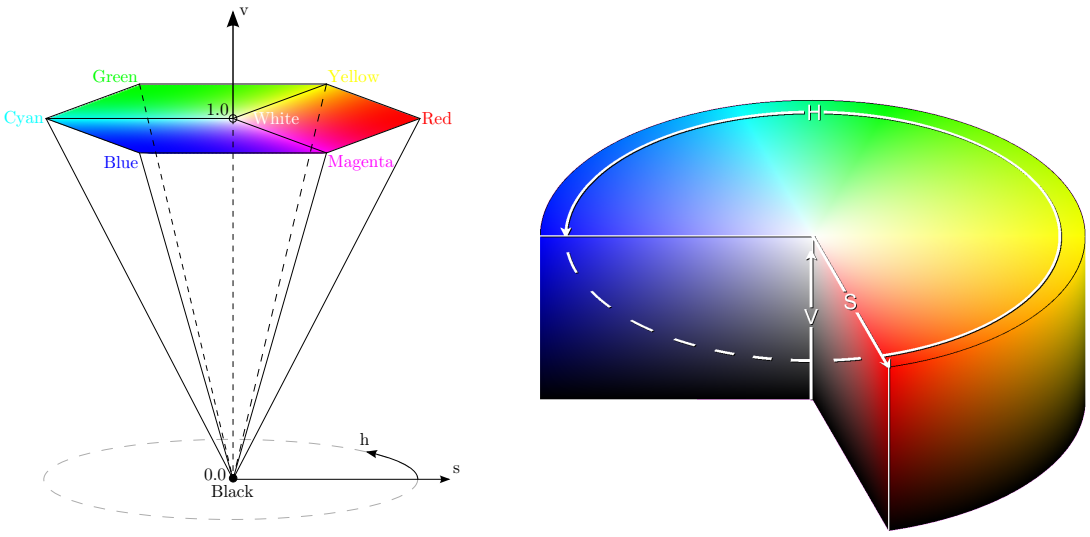
$$b = RMS(CD_i) = \sqrt{\frac{\sum_{i=1}^{N} (CD_i)^2}{N}} \tag{4.22}$$

In order to use a single threshold for subtraction, $\alpha_i$ and $CD_i$ are normalized by the respective $a_i$ and $b_i$: $\widehat{\alpha}_i = \frac{\alpha_i - 1}{a_i}$, $\widehat{CD}_i = \frac{CD_i}{b_i}$. A pixel is classified according to the following rules:

$$\begin{array}{ll} F: & \widehat{CD}_i > \tau_{CD} \text{ , else} \\ B: & \tau_{\alpha 1} < \widehat{\alpha}_i < \tau_{\alpha 2} \text{ , else} \\ S: & \widehat{\alpha}_i ¡ 0 \text{ , else} \\ H: & \text{otherwise} \end{array} \tag{4.23}$$

### HSV Color Space

HSV is not a hardware oriented model as RGB, but a perceptual model, developed to facilitate intuitive mixing of colors for artists [Smi78]. It is just one of many representations




(a) HSV Cone (b) HSV Cylinder

Figure 4.6: Representations of the HSV color space.

of the RGB color space in terms of 3D-polar coordinates [Han03]. The transformation of RGB to a 3D-polar color space is done with the help of an *opponent color space (OCS)* introduced in RGB space. The OCS is built as follows: An *achromatic axis,* that is aligned with the main axis of the RGB color cube, is placed in the RGB space. A *chromatic plane* perpendicular to the achromatic axis and the origin at the intersection with the achromatic axis is introduced and all RGB values are projected on that plane. The coordinates on the achromatic axis give a measure of the lightness and the position on the plane a measure for the chromaticity of a pixel [Han08].

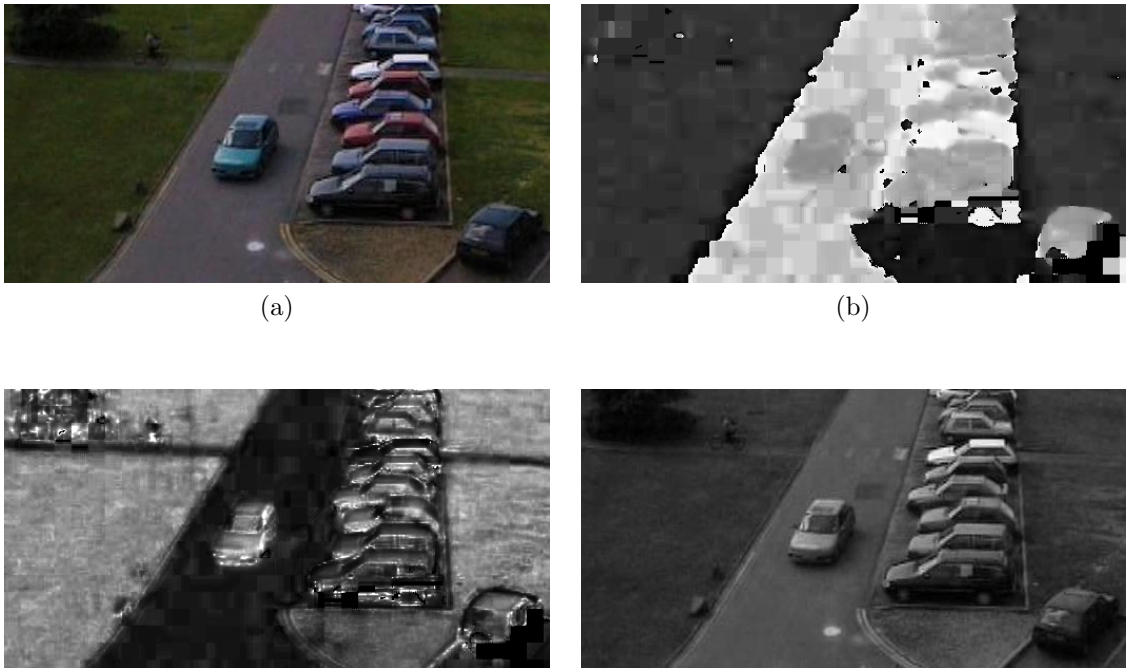
In HSV chromaticities are represented by the two values *hue* $h$ and *saturation* $s$. Brightness is represented as *value* $v$. Under the condition that $R, G, B \in [0, 1]$ and assigning $max$ to the greatest of value $R, G, B$ and $min$ to the lowest, RGB values are converted to $h, s, v$ with:

$$h = \begin{cases} undefined, & \text{if } \max = \min \\ \frac{G-B}{\max - \min} \times 60°, & \text{if } \max = r \\ 2 + \frac{B-R}{\max - \min} \times 60°, & \text{if } \max = g \\ 4 + \frac{R-G}{\max - \min} \times 60°, & \text{if } \max = b \end{cases} \tag{4.24}$$

$$s = \begin{cases} 0, & \text{if } \max = 0 \\ \frac{\max - \min}{\max} = \frac{max-min}{v}, & \text{otherwise} \end{cases} \tag{4.25}$$

$$v = max \tag{4.26}$$

Hue represents the dominant color, or more precisely the dominant wavelength in degrees $[0°, 360°)$, where red is at 0°, green at 120° and blue at 240°. Saturation refers to the proportion of pure light of the dominant wavelength, and is given in the range $[0, 1]$. Value is in the range of $[0, 1]$ and is a measure for the brightness. The HSV color space is typically visualized as a "hexcone" as illustrated in Figure 4.6a. The $v$-axis corresponds



(a) (b)

(c) (d)

Figure 4.7: Sample Image in HSV format: The original image (a), the chromatic channels hue $h$ (b) and saturation $s$ (c) and value $v$ (d).

to the main diagonal of the RGB cube. The chromatic plane is at $v = 1$, and contains the colors planes of the RGB cube where $R = 1$, $G = 1$ and $B = 1$. Due to the normalization of the saturation that was introduced by Smit [Smi78], it is actually shaped as a cylinder (see Figure 4.6b). This normalization is the root of the drawbacks of applying HSV for shadow detection. First, the saturation is dependent on the brightness. Second, it is not necessarily low in achromatic regions [Han03]. The effects are illustrated in Figure 4.7c: The dark areas around the bush in the upper-left corner and the window of the car in the center appear fully saturated. Additionally image noise is amplified in these regions.

In contrast to the RGB space, where all three channels are independent, $h$ and $s$ have a strong relation. For example, if $s$ is approximately 0, a big difference in $h$ does not denote a big difference in observed chromaticities, as there is no dominant wave length. This observation has implications on the exact formulation of the shadow detection algorithm, as thresholds for these values are to be dependent as well [NBT08].

**Shadow Detection**

In [CGP+01] a traffic surveillance system, where the segmentation results are enhanced by applying shadow removal in HSV, has been presented. Only pixels that are marked as foreground by the motion detection are evaluated. Cues 1 and 2 are exploited for their shadow classification. A pixel $I$ is marked as Shadow if:

$$
\begin{aligned}
&if\ \tau_1 \leq \tfrac{I_v}{B_v} \leq \tau_2 \\
&\wedge\ (I_s - B_s) \leq \tau_s \\
&\wedge\ |(I_h - B_h)| \leq \tau_h
\end{aligned}
\tag{4.27}
$$

where B is the corresponding background model value, subscripts $v, s, h$ denote the channels in HSV. $\tau_1$and $\tau_2$ name the lower and upper brightness thresholds, $\tau_s$ the saturation difference threshold and $\tau_h$ the hue difference threshold.

### IHLS Color Space

The Improved Hue Luminance and Saturation (IHLS) color space was introduced by Hanbury and Serra [Han03, HS03]. Compared to similar color spaces (GHLS, HSV, etc.), the normalization of the saturation by the brightness is removed. Referring to the OCS introduced for the construction of the HSV space, for IHLS we have a luminance $y$ giving the position on the achromatic axis, the saturation $s$ is defined by the distance from the axis and hue $\theta^H$ by the angle with respect pure red.

The conversion from RGB to IHLS as given in [BWHK06], which differs from the formulations in the original paper is presented here:

$$
\begin{aligned}
y &= 0.2125R + 0.7154G + 0.0721B && (4.28)\\
s &= max(R,G,B) - min(R,G,B) && (4.29)\\
cr &= \sqrt{cr_1^2 + cr_2^2} && (4.30)\\
& cr_1 = R - \frac{G+B}{2} && (4.31)\\
& cr_2 = \frac{\sqrt{3}}{2}(B-G) && (4.32)\\
\theta^H &= \begin{cases} undefined & if\ cr = 0 \\ arccos(\frac{cr_1}{cr}) & if\ cr \neq 0 \wedge cr_2 \leq 0 \\ 360^\circ - arccos(\frac{cr_1}{cr}) & if\ cr \neq 0 \wedge cr_2 > 0 \end{cases} && (4.33)
\end{aligned}
$$

where $cr_1$ and $cr_2$ denote chrominance coordinates and $cr \in [0,1]$ the chroma. Note that hue is undefined if $s = 0$ and that the entropy of hue decreases for values of s close to the achromatic axis. Figure 4.8 illustrates the chromatic plane of the IHLS color space

### Shadow Detection

Before going into the shadow detection using the IHLS background model, some foundations have to be laid out. While standard linear statistics can be used for brightness and saturation, hue is an angular value. Thus circular statistics have to be applied. Consider the following example: Given two hue observations $\theta_A = 1^\circ$ and $\theta_B = 359^\circ$ the linear mean gives 180°. Additionally the tight relationship of the chromatic components hue and saturation should be considered. In [BWHK06] this is called *Saturation weighted hue statistics*. Let $(\theta_i, s_i)\, i = 1, \ldots, n$ be $n$ pairs of hue and saturation values. The vector on

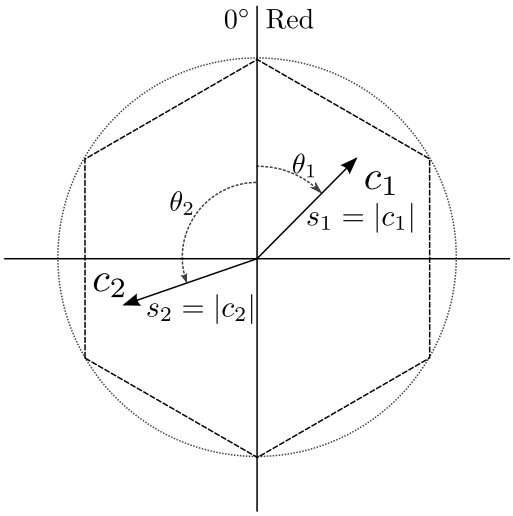



Figure 4.8: Diagram of the chromatic plane of the IHLS color space.

the chromatic plane from $(0, 0)$ to the point $(\theta_i, s_i)$ is given by $(s_i cos\theta_i, s_i sin\theta_i)$. For the mean chrominance $\overline{\mathbf{c}}_n$ we have:

$$\overline{\mathbf{c}}_n = \left(\frac{\mathcal{C}}{n}, \frac{\mathcal{S}}{n}\right)^{\mathrm{T}} \tag{4.34}$$

where

$$\mathcal{C} = \sum_{i=1}^{n} cos\theta_i,\ \mathcal{S} = \sum_{i=1}^{n} sin\theta_i \tag{4.35}$$

Testing similarity of $\overline{\mathbf{c}}_n$ and a newly observed chrominance vector $\mathbf{c}_o$ the Euclidean distance is used:

$$D = \sqrt{\left(\overline{\mathbf{c}}_n - \mathbf{c}_o\right)^{\mathrm{T}} \left(\overline{\mathbf{c}}_n - \mathbf{c}_o\right)} \tag{4.36}$$

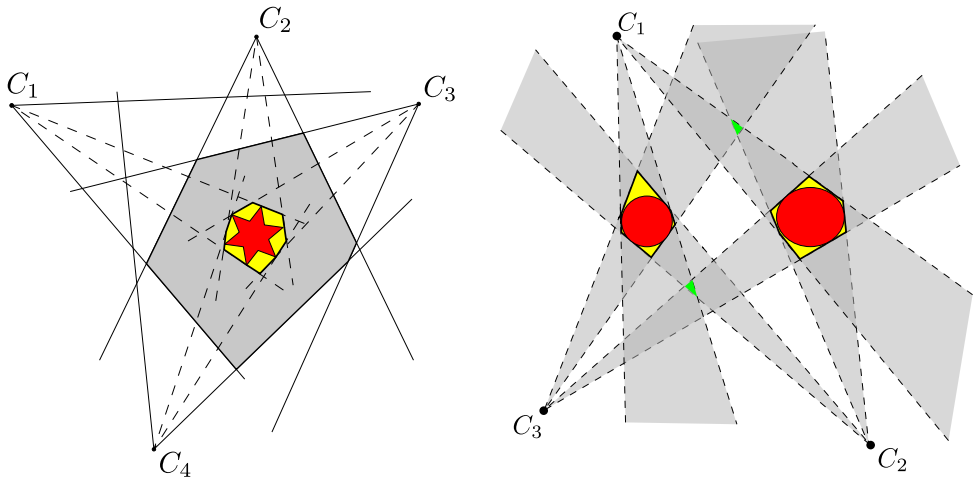
Considering a uni-modal background model as presented in Section 4.1.1 the background is modeled pixel-wise by the mean luminance $\mu_y$ and standard deviation $\sigma_y$, the mean chrominance vector $\overline{\mathbf{c}}_n$ and the mean Euclidean distance $\sigma_D$. A pixel with luminance $y_o$, chromatic vector $\mathbf{c}_o$ is classified as foreground if:

$$\left|y_o - \mu_y\right| > \alpha\sigma_y \ \vee\ \left\|\overline{\mathbf{c}}_n - \mathbf{c}_o\right\| > \alpha\sigma_D \tag{4.37}$$

with $\alpha$, the foreground threshold. Similar to shadow classification in HSV, a pixel is classified as shaded background if:

$$\begin{array}{rrcl} y_o < & {}_y \wedge y_o - \mu_y & < & \beta\mu_y \\ \wedge & s_o - \left\|\overline{\mathcal{R}}_n\right\| & < & \tau_{ds} \\ \wedge & \left\|\mathbf{h}_o\overline{\mathcal{R}}_n - \overline{\mathbf{c}}_n\right\| & < & \tau_h \end{array} \tag{4.38}$$

where $s_o$ and $h_o$ are the observed saturation and hue, $\overline{\mathcal{R}}_n = \|\overline{\mathbf{c}}_n\|$. The first equation checks if the observation is darker than the background, with the upper threshold $\beta$ taking into account the strength of the predominant light source [BWHK06]. The test of reduced of




(a) (b)

Figure 4.9: Shape from Silhouette with four cameras $(C_1, \ldots, C_4)$ for one object (a): The gray area marks the intersection of all camera views; the reconstructed shape is yellow and the original shape red. Shape from shading with two objects (b): When multiple objects are in the scene, ambiguities can be reconstructed as "ghosts" (green).

saturation is performed in the second equation. The third check is performed using the statistical model introduced earlier in this section, as opposed to the linear comparison in the HSV shadow detection. The observed hue vector $\mathbf{h}_o = (cos\theta_o, sin\theta_o)^{\mathrm{T}}$ is scaled to the same length as the mean chrominance vector and tested against it using the Euclidean distance.

## 4.2 Shape from Silhouette

Shape from Silhouette *(SFS),* or voxel carving, aims to reconstruct the 3D shape of an object form binary silhouette images of the object observed from different viewpoints [Bau74]. The closest approximation to the object that can be obtained with SFS is the object's visual hull. The visual hull $\mathcal{S}$ is defined as the maximal object that gives the same silhouette as $\mathcal{S}$ from any possible viewpoint [Lau91]. Therefore only objects that conform to their visual hull are exactly reconstructable. It is obvious that this does not include concave objects. The human body however is for the most part convex and smooth, which makes SFS a popular and robust method for 3D human body modeling and motion tracking algorithms [CH04, HLS04, KG06, MGBB07]. In Figure 4.9a, shape from silhouette, with four cameras $(C_1, \ldots, C_4)$ is illustrated.

The standard algorithm for computing the visual hull works as follows [CH04]:

1. Subdivide the observation space into voxels
2. Project each voxel onto the image plane of each view
3. Keep the voxels that lie within the silhouette of each view

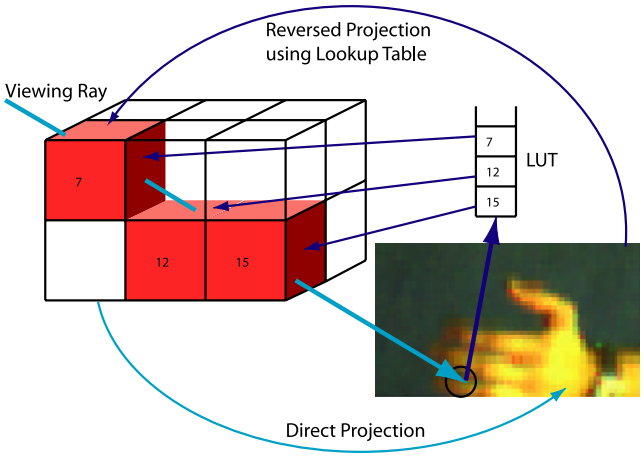



Figure 4.10: Illustration of the Shape from Silhouette approach from [KG06]. The look-up table that maps pixels to voxels is computed offline. This way the volume can be very efficiently updated at run-time, since the – computationally expensive – voxel projection is preprocessed.

With a high resolution volume and with an increasing number of views, this is computationally and memory intensive. A working volume of $4000 \times 4000 \times 2000$mm and voxel size of 25mm, spans a $160 \times 160 \times 80$ voxel space. Computing the volume in a coarse to finer resolution instead of the fixed sized voxels is a common speedup technique [CH04]. Kehl et al. [KG06] address the problem the other way round by mapping of pixels to voxels. In a preprocessing step, a look-up table for each view is built, where for each pixel a list of voxels that project on this particular pixel is generated. This approach is illustrated in Figure 4.10. This representation has several advantages: first of all, the image coordinates of voxels do not have to be computed during run-time nor are they stored in memory. The representation of a voxel can be simplified to a bit mask, where each bit $b_i$ is 1 if it is in the foreground of camera $i$, which can be evaluated fast. Additionally, the volume be can updated instead of being recomputed from scratch in each frame: The bit mask of a voxel only changes, if one of the pixels, the voxel is projected onto, changes it's foreground-background membership. Thus only pixels set in the difference of two consecutive foreground segmentations have to be evaluated. This again reduces the number of voxel look-ups. However there is additional memory required to store the look-up table for each view.

With the advent of programmable graphics hardware, GPU based approaches have been presented that drastically increase the reconstruction speed [HLS04, MGBB07].

There are two major problems, which shape from silhouette suffers from [MGBB08]. First, while adding additional views improves the reconstruction quality, the working volume is reduced simultaneously (see Figure 4.9a). Moreover SFS can possibly reconstruct visual ambiguities as "Ghost" objects, where empty regions are extracted as illustrated in Figure 4.9b. Solutions have been proposed in [MGBB08].

The reconstructed volume lies in a global 3D world coordinate system. This simplifies further processing, as it is independent of the different cameras. Further, noise in the silhouette images is automatically filtered in the volume estimation process, since it is not

coherent across views.

## 4.3 Feature Extraction

A number of features that describe elementary properties such as shape, color, texture or motion of images or image regions have been presented in the literature. We aim for real-time detection of falls in multiple cameras, thus descriptors have to be computationally inexpensive. After noise is removed from the object segmentation and connected components are grouped into blobs, these features are extracted. In the following sections a range of features that has been utilized widely in the fall detection literature is presented.

### 4.3.1 Bounding-Box Aspect Ratio

A common way to model and track objects is using their bounding box [Sen02, Sal04, TM05, KD06]. The bounding-box or minimum bounding rectangle of a set of pixels is defined by $(x_o, y_0, w, h)$, the top left point, width and height. It is the smallest axis aligned rectangle that completely contains the region. Previously, the bounding box aspect ratio has been used extensively [TDc05, AKS+06, RMSAR07, VMS07] as a feature for fall recognition:

$$B^{lf} = \frac{h}{w} \tag{4.39}$$

In the early fusion case, the bounding box aspect ratio is defined as:

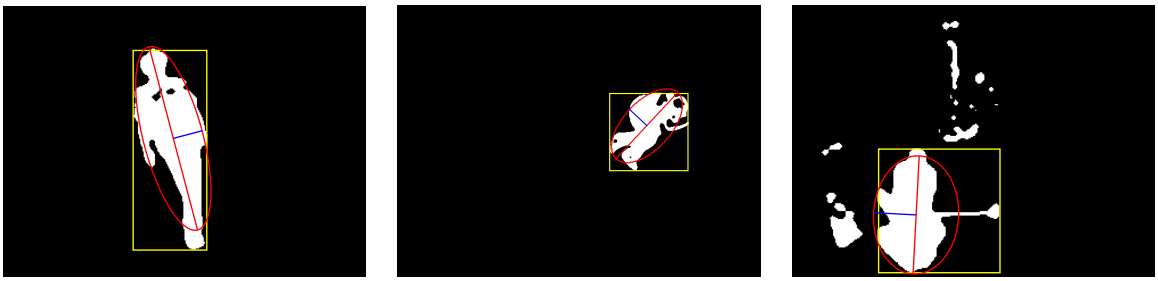
$$B^{ef} = \frac{h}{mean(w_x, w_y)}. \tag{4.40}$$

Since the bounding box is aligned with the image axis, the view point and camera angle have a strong effect on its descriptiveness of 2D bounding-boxes. Another drawback is that it is highly sensitive to shape changes, possibly due to segmentation errors. This restriction applies to the 3D bounding box as well. Figure 4.11 illustrates these issues.

### 4.3.2 Ellipse

Instead of bounding-boxes, ellipses too have been widely used for modeling and tracking [NCM04, TM06, RMSAR07, HHdW08]. They offer various advantages compared to a bounding-box approach. Since they are not aligned to the image axes, they can more accurately model objects. Another advantage is their inherent stability against segmentation errors. Since the covariance matrix is used to estimate the properties of the ellipse, outliers due to inaccurate silhouette segmentation are removed. In Figure 4.11 bounding box and ellipse features are compared.

An ellipse is defined as a quintuple $\langle c_x, c_y, \theta, a, b \rangle$ with the center $c_x, c_y$, the orientation $\theta$, the angle between the major–axis and the x–axis, and the length of the major and minor semi axes $a$ and $b$. The ellipse can be estimated by computing the covariance matrix of the binary segmentation.



(a) (b) (c)

Figure 4.11: The descriptiveness of the bounding-box (yellow) is influenced by the camera placement (a) and (b). It is further frail to segmentation errors (c). Modeling blobs using the ellipse (red), is more accurate and stable under segmentation errors.

The moment of order $(i, j)$ for a binary image $I(x, y)$ is defined as:

$$m_{ij} = \frac{\sum_x \sum_y x^i y^j I(x, y)}{\sum_x \sum_y I(x, y)} \tag{4.41}$$

It is obvious that zeroth moment $m_{00} = 1$. The mean (centroid) of $f(x, y)$ is given by the first moments $\bar{x} = m_{10}, \bar{y} = m_{01}$. With the centroid, we can define the central moment of order $(i, j)$ as:

$$\mu_{ij} = \frac{\sum_x \sum_y (x - \bar{x})^i (y - \bar{y})^j I(x, y)}{\sum_x \sum_y I(x, y)} \tag{4.42}$$

The central moments $\mu_{11}, \mu_{20}, \mu_{02}$ correspond to the covariance between $x$ and $y$, between $x$ and itself and between $y$ and itself. Thus the covariance matrix $\sum$ can be expressed with first and second order central moments as:

$$\sum = \begin{bmatrix} \mu_{20} & \mu_{11} \\ \mu_{11} & \mu_{02} \end{bmatrix}. \tag{4.43}$$

Since the covariance matrix is symmetric and positive semidefinite the eigenvalues $\lambda_{max}$, $\lambda_{min}$ and the associated eigen-vectors can be easily computed.

$$\lambda_{min,max} = \frac{(\mu_{20} + \mu_{02}) \pm \sqrt{4\mu_{11}^2 + (\mu_{20} - \mu_{02})^2}}{2} \tag{4.44}$$

The eigen-vectors are perpendicular and the vector associated with $\lambda_{max}$ defines the major orientation $\theta$ of the ellipse with the $x$–axis. The length of the semi-major and semi-minor ellipse axes $a$ and $b$ equal $\sqrt{\lambda_{max}}$ and $\sqrt{\lambda_{min}}$ respectively [THS99]. In voxel space, ellipsoids can be computed via moments of the volumetric data in the same way.

### Orientation

For detecting falls, having the orientation of the ellipse, which corresponds to the main orientation of the human body, is an important feature. A Person standing upright has a main orientation almost perpendicular to the ground plane, while it is parallel to the

ground plane, when the person is lying. However, the main axis can only be measured properly if it is perpendicular to the optical axis of the camera [HHdW08]. In 3D however this does not apply and the orientation of the major axis can be directly used as a meaningful feature.

#### Axis Ratio

In [ZMK10] the axis ratio of the ellipse in image and voxel space has been suggested.

$$A = \frac{a}{b} \tag{4.45}$$

#### Centroid Height

In [ALK+09], the height of the person's center is approximated by the centroid height.

### 4.3.3 Motion

It was shown, that falls can be distinguished from normal activities by using vertical and horizontal velocities characteristics in a 3D world coordinate system [Wu00]. They are a major cue for the critical falling phase as well as the postfall phase. The objects frame to frame velocity vector at time $t$ is computed from the center $\mathbf{C}$ as

$$\mathbf{v}_t = \mathbf{C}_{t-1} - \mathbf{C}_t \tag{4.46}$$

where $\mathbf{C}$ is the centroid of a bounding box or an ellipse. Since the direction of the velocity is typically not meaningful, we are only interested in the speed

$$v_t = |\mathbf{v}_t| . \tag{4.47}$$

Additionally the acceleration is measured as the rate in which speed changes:

$$a_t = v_{t-1} - v_t \tag{4.48}$$

#### Motion History Images

Motion History Images (MHI) introduced in [BD01] are another way of representing how and where motion occurs. Each pixel in the MHI $H_\tau$ is a function of the temporal history of motion at that point. Let $F(x, y, t)$ be the segmented foreground at time $t$ and $\tau$ the duration for which the MHI is computed, then the MHI at time $t$ is

$$H_\tau(x, y, t) = \begin{cases} \tau & if\ F(x, y, t) = 1 \\ max(0, H_\tau(x, y, t-1) - 1) & \text{otherwise.} \end{cases} \tag{4.49}$$

Rougier et al. propose a motion coefficient $C_{motion}$ based on the MHI [RMSAR07].

$$C_{motion}(t) = \frac{\sum_{Pixel\,(x,y) \in blob} H_\tau(x, y, t)}{\#pixels \in blob} \tag{4.50}$$

$C_{motion}$ is within $[0, 1]$ indicating no motion 0 and full motion 1. In [BD01] and [RMSAR07] the importance of defining $\tau$ is laid out. A duration of $500ms$ is proposed in [RMSAR07].

Figure 4.12: Illustration of the head detection approach proposed by [HHdW08]

#### Motion speed

A similar approach that is inspired by MHI, which just computes the inter-frame motion speed is used in [ZMK10]. Here the relative number of new motion pixels in the current frame compared to the previous frame is estimated:

$$M^{ef} = \frac{|F(t) \setminus F(t-1)|}{|F(t)|} \tag{4.51}$$

where $F(t)$ denotes the set of foreground pixels at time $t$.

### 4.3.4 Head Position

For the verification of falls, it has been proposed to use the head position as an additional feature [RMSAR06, YNC09]. In [RMSAR06] a single calibrated camera provides the image evidence, while the head position is computed with the POSIT algorithm [DD95]. Input arguments for POSIT are the known 3D dimensions of the head (which are based on anthropometric data), the corresponding 2D points of the ellipse modeling the head, and the camera calibration matrix. The algorithm calculates the relative position of the head in the camera coordinate system, from which the world coordinate position of the head can be easily computed. Similarly, in [YNC09] the position is computed from two calibrated cameras.

Hazelhoff et al. [HHdW08] propose to estimate the head position in order to increase the robustness of the fall detector and for identifying objects as humans. The proposed head detection is straight forward: The head is considered as the skin colored blob farthest away from the center along the main axis on the border of the silhouette (see Figure 4.12). A head candidate is matched against size constraints. The head is tracked in successive frames by searching for skin-colored blobs nearby the head position. The head position is used just as an additional cue for their system: if a fall is detected, it is checked if the head position has remained stationary. If this is the case, the fall is rejected.

Figure 4.13: Input image (left) and the corresponding trained accumulated hitmap (right) [ZK10].

### 4.3.5 Accumulated Hitmap

Approaches that detect the postfall phase, where the person is on the ground, typically model the spatial unexpectedness of the fall. In [NCM04] the motion trajectories are used to split the observation space into inactivity regions like beds, sofas, or chairs, where it is commonly expected that little or no motion occurs and activity zones, where little or no motion is not expected. When a fall event is recognized, it is checked, if the spatial location of the fall is within an inactivity zone and can thus be rejected.

To model the unexpectedness of an event, the *accumulated hitmap*, which operates on the pixel level, has been proposed in [ZK10]. The hitmap is a counter for the consecutive appearance of foreground at a given location, and is decreased if the location is background for $n$-consecutive frames, where $n$ controls the robustness of the hitmap. In Figure 4.13 a sample input image and the corresponding accumulated hitmap are shown. The unexpectedness $H_U(x, y)$ is essentially the deviation of the observed hitmap $H_O$ and the trained hitmap $H_T$. In the original paper the following equation is proposed to compute $H_U(x, y)$:

$$H_U(x,y) = \begin{cases} \left(1 + \frac{\alpha}{1+max(H_T)-H_O(x,y)}\right)^{\frac{H_{diff}(x,y)}{2}} & if\ H_O > H_T \\ 0 & otherwise \end{cases}$$

where $H_{diff} = H_O - H_T$. In [ZZK10] the accumulated hitmap has been used for fall verification in 2D. Therefore, the unexpectedness is summed up in the area contained in the bounding box and compared to a threshold. The fall is verified if the threshold is exceeded in a four seconds verification window (two seconds before and two seconds after the fall incident is suggested.

Building on this approach, we suggest to compute the accumulated hitmap for the early fusion approach directly in voxel space. A visualization of the trained accumulated hitmap is show in Figure 4.14.

## 4.4 Selected Approach

The approaches presented in this chapter represent some of the commonly used methods to abstract the raw input into meaningful features vectors for the subsequent event understanding. Leaving the choice of cameras aside, the fundamental part in the presented fall detection framework is the person's silhouette extraction. These approaches represent

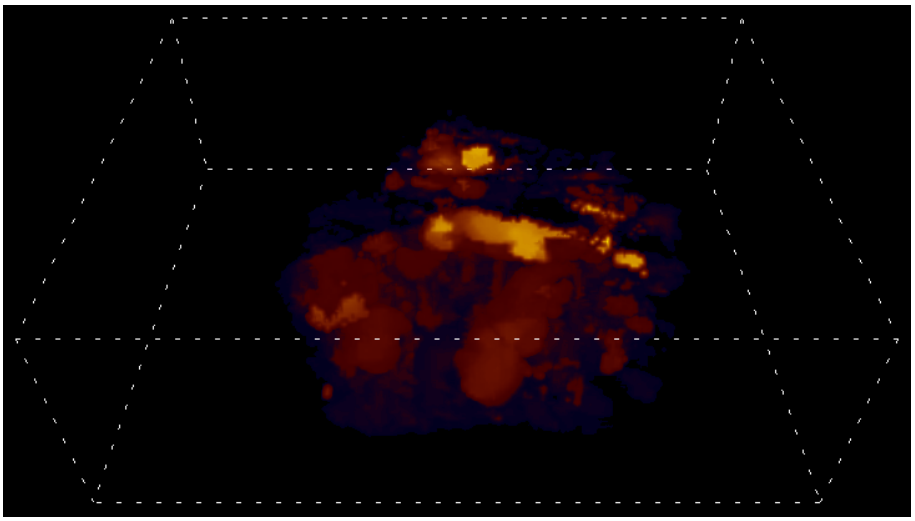


Figure 4.14: Visualization of the trained accumulated hitmap in Voxel space. Blue areas indicate low, red medium and yellow a high hit-count.

the state of the art. Due to the low computational cost and low memory requirements, a Color Mean and Variance approach with a Shadow removal in normalized RGB space is suggested. While the other presented methods provide higher quality results, they are computationally more demanding and not sufficiently efficient to be applied in the suggested setup.

With the Accumulated Hitmap, a feature for modelling the unexpectedness has been presented. The other features shown in Section 4.3 have previously been applied for fall detection. As part of the experiments these features and feature sets are evaluated.

# Chapter 5

# Event Understanding

In an event understanding framework the input is abstracted into meaningful units. These features are processed by the event model, which determines if an event of interest occurred. The output is usually a binary decision whether a particular event occurred, or an activity summary [LRR09]. Features have been discussed in the previous chapter. In this chapter the terminology will be defined and an overview of approaches is given. Finally, these are examined for their suitability for the proposed fall detection approach.

The vast application space in which event understanding has been applied has lead to a variety of different terms for essentially same concepts. From hand washing [MCB04] and sign language gestures [LP07], to tennis strokes [YKI92] or airport apron activities [FVB+07], different kinds of events have been tried to model and recognize. Hence, terms like "behavior", "activity", "scenario", "gesture" or "event" are used in the literature. While they carry some context about the particular application and domain and event complexity, they essentially describe the same concepts [TCSU08]. Recently, the generic, unifying terminology that is presented in this section has been proposed in [LRR09]. It is formulated around the concept of an "event", which has the following characteristics:

1. An event occupies a period of time.
2. An event is built of smaller semantic unit building blocks.
3. An event is described using the salient aspects of video input.
4. An event is an occurrence of interest.

When describing the various properties of these events, such as the hierarchical, temporal or content composition, instead of introducing vague distinctions, prefixes are attached to name these properties. That way, an event may be recursively composed of multiple "sub-events" and the same concept is used to describe "simple" as well as complex activities. Analogous a "super-event" is composed of sub-events. An "atomic event" has no sub-event composition. When referring to the abstraction primitives that are used to describe an event, content prefixes are inserted. Thus an "object based event" is modeled by means of object properties and tracking (size, shape, trajectory) while "pixel based event" refers to events modeled using pixel features such as color, texture or gradient. The temporal composition of events is addressed with the terms "single-threaded-event" and

Composition Prefixes

| | |
|---|---|
| Atomic | Has no sub-event composition |
| Composite | Has sub-event composition |

Content Prefixes

| | |
|---|---|
| Pixel-Based | Described by pixel-level features (color, texture, gradient) |
| Object-Based | Described by object-level feature (size, shape, trajectory) |

Temporal Prefixes

| | |
|---|---|
| Single-Thread | Has sequential temporal relationships between sub-events |
| Multi-Thread | Has non-sequential temporal relationships between sub-events |

Relation to Event of Interest Prefixes

| | |
|---|---|
| Sub | Component of an event |
| Super | Composed of events |

Table 5.1: An overview of the event terminology as proposed in [LRR09].

"multi-threaded-event", describing linear ("single-threaded-") or the non-linear ("multi-threaded-") composition.

With the introduction of the "event" term, much of the context of previously used terms as "activity", "gesture" or "behavior" is lost. To re-establish context, the "event domain", a possibly natural language description of precisely what kind of events are tried to recognize, is introduced. In Table 5.1 the proposed terminology is summarized.

## 5.1 Event models

Event models aim to describe and classify the events of interest in a particular event domain. They build upon features that have been identified as meaningful in the data abstraction process. The classification task can be seen as a labeling task: Given an observation (the feature vector) $\mathbf{x} = (f_1, \ldots, f_n)$ the aim is to choose the appropriate event label $l_x$ from a set of $m$ class-labels $\mathcal{L} = \langle l_1, \ldots, l_m \rangle$. As of today, a variety of event models has been presented [LRR09, TCSU08].

Finite State Machines (FSM), Grammars and Petri Nets are examples of deterministic event models, while Bayesian Networks, Hidden Markov Models (HMM), Conditional Random Fields (CRF) and Stochastic grammars associate a probability score with the occurrence of an event [LRR09]. Discriminative models directly model the posterior probability $p(l \mid x)$. However generative models first model and learn the joint probability $p(x, l)$. Each approach has its advantages as well as limitations [UB05]: while discrim-

inative models have a lower error rate when using a large number of training samples, generative models converge faster even with a small training set. Additionally generative models can handle incomplete data and additional classes more flexible and are thus considered to be more suited for complex patterns [UB05].

In [LRR09] event models are grouped in three categories based on the utilization of semantic knowledge for the recognition task: "pattern-recognition methods", "state models", and "semantic models". Pattern Recognition Methods do not address the representational aspect of event modeling. Event recognition is rather treated as a classic pattern recognition or classification problem. Hence algorithms such as $k$-Nearest Neighbor ($k$-NN), Neural Networks (NN), Support vector machines (SVMs) or Boosting fall in this category. These methods require minimal semantic knowledge of the application domain and are in most cases fully specified from training data. Pattern recognition methods thus are generally straight forward to implement and have moderate processing costs. Since these models do not incorporate the high level semantic knowledge about the event domain, they are mostly applied for the recognition of atomic events [LRR09].

State Event models improve the pattern recognition methods as they intrinsically model the structure of the state space of the event domain. This allows these models to capture both the temporal evolution of states as well as their hierarchical nature. Finite State Machines, Hidden Markov Models or Bayesian Networks fall into this category. Given previously labeled training data, these allow the estimation of optimal model parameters. However, with increasing complexity of events, the state space increases and training as well as evaluating becomes unmanageable.

Semantic models do not try to define the entire state space as state event models do, however semantic rules, constraints and relations of events are defined, which is more like humans define events and connections between events. They allow the capturing of high-level semantics like long-term temporal dependence, hierarchy, partial, ordering concurrency as well as complex relations among sub-events and abstraction features [LRR09]. As a result, the models have to be manually specified by domain experts. Learning or training of the model structure and parameters is either ill defined or impossible for these models.

Choosing the appropriate event model is crucial to achieve a meaningful labeling of the feature input. It is important that the model is capable of handling the various possible event prefixes of the specific domain. Further, when applying an approach that is based on training samples, the availability of a large enough training dataset is essential. Therefore, in the following sections modeling approaches are reviewed with an emphasis on those that have been proposed to tackle fall detection in the literature.

### 5.1.1 $k$-Nearest Neighbor

The Nearest Neighbor classifier assigns to an unlabeled observation $\mathbf{x}$ the class-label of the closest training point. Typically, the $k$-Nearest Neighbors ($k$-NN) are evaluated for classification and via a majority vote the label for $\mathbf{x}$ is determined. $k$-NN is illustrated in Figure 5.1. A proper choice of distant measure or previously normalizing the feature space is crucial for K-NN to perform well [Bis06]. The quality of the model is additionally dependent on the coverage of the feature space with labeled points. The more dense it

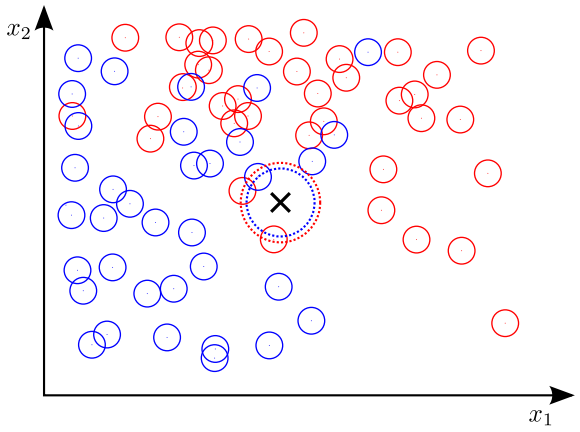



Figure 5.1: With the $K$-Nearest Neighbor classifier, a newly observed point (shown by the black cross) is labeled according to the majority class membership of the $K$ closest training data points.

is covered, the better the classification results. However, with the size of the dataset increasing, so do the computational and memory costs during recognition. It should be noted that it was shown that as the dimensionality of the feature space increases, the distance of the nearest point approaches the distance of the farthest point [BGRS99].

In [PLSR08] and [FCLD08] $k$-NN has been applied to the recognition of falls.

### 5.1.2 Neural Networks

Artificial Neural Networks have their origin in attempts to find mathematical representations of biological information processing systems [Roj96]. They simulate biological neural systems with neurons connected by communication channels. In a feed-forward network, the connections are organized in layers and the information flow is unidirectional from one layer to the next. Nodes within a layer are not connected. This is an important simplification, which makes Neural Networks of practical use. The basic neural network model is described as a composition of functional transformations. In the first network layer, $M$ linear combinations $a_0, \ldots, a_M$ are computed of the input variables $x_1, \ldots, x_D$

$$a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)} \tag{5.1}$$

where $j = 1, \ldots, M$ the superscript (1) indicates that the weights and the biases are in the first layer. The $a_j$ are then transformed using a differentiable, nonlinear *activation function* $h$:

$$z_j = h\left(a_j\right) \tag{5.2}$$

These are referred to as *hidden units* and they are again linearly combined to yield the *output unit activations*, the second layer of the network

$$a_k = \sum_{j=1}^{M} w_{ki}^{(2)} z_i + w_{k0}^{(2)} \tag{5.3}$$

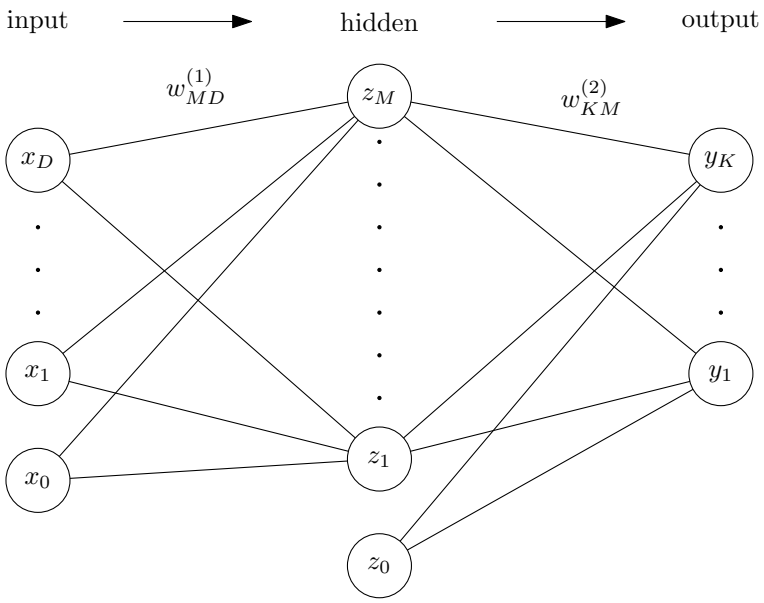



Figure 5.2: Illustration of a feed-forward neural network. The nodes represent the input, hidden variables and output, while the connections illustrate the weights [Bis06].

where $K$ is the total number of outputs, and $k = 1, \ldots, K$. Again there are weights $w^{(2)}_{ki}$ and bias parameters $w^{(2)}_{k0}$. The output unit activations are transformed using an appropriate activation function to give the network outputs $y_k$

$$y_k = \sigma\left(a_k\right). \tag{5.4}$$

Activation functions that are generally used are hyperbolic tangent **tanh** or the logistic sigmoid function $\sigma\left(a\right) = \frac{1}{1+e^{-a}}$. This choice is directly related to the problem of training the network [Bis06]. To simplify notation, the bias parameters in (5.1) (and in all other layers) can be integrated into the weighted parameters by adding the additional input variable $x_0 = 1$:

$$a_j = \sum_{i=0}^{D} w^{(1)}_{ji} x_i$$

Thus the overall neural network function $y_k\left(\mathbf{x}, \mathbf{w}\right)$ with one hidden layer is given by:

$$y_k\left(\mathbf{x}, \mathbf{w}\right) = \sigma\left(\sum_{j=0}^{M} w^{(2)}_{ki} h\left(\sum_{i=0}^{D} w^{(1)}_{ji} x_i\right)\right) \tag{5.5}$$

An example of a two layer feed-forward neural network is illustrated in Figure 5.2. The nodes represent the input, hidden and output variables, while the edges represent the weight parameters of the network. The two layer network – one hidden layer and the output layer – presented here can be easily extended with additional hidden layer units in the form of (5.3) and activation functions (5.4). Neural Networks have been widely used and studied as universal approximators [Bis06]. The key problem is training the network parameters given a set of training data.

Given a training set $\{\mathbf{x}_n\}$ of size $N$, and a set of target values $\{\mathbf{t}_n\}$, the optimal networks parameters $\mathbf{w}$ can be estimated by minimizing a cost function e.g.: the sum-of-squares error function:

$$E(\mathbf{w}) = \sum_{n=1}^{N} \{y(\mathbf{x}_n, \mathbf{w}) - t_n\}^2 \tag{5.6}$$

In practice this is complicated due to the non-linearity of $y(\mathbf{x}_n, \mathbf{w})$. The backpropagation algorithm has been widely used as an efficient technique for minimizing the error function using gradient descent [Bis06]. Gradient descent is a method for finding the minimum of a function by iteratively moving from the current point in the direction of the negative gradient. In each step the weights are updated as follows

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla E\left(\mathbf{w}^{(\tau)}\right) \tag{5.7}$$

for small enough $\eta > 0$, $E\left(\mathbf{w}^{(\tau+1)}\right) < E\left(\mathbf{w}^{(\tau)}\right)$. Since $E$ is a function of the entire training set $\{\mathbf{x}_n\}$, it is required to process the entire set in order to calculate $\nabla E$.

An efficient solution for computing the gradient is given by the previously mentioned backpropagation algorithm [Roj96]. In the first step, the output activations are computed and the derivatives of the error function with respect to the weights are evaluated and propagated backwards through the network. In the second stage, the weights are adjusted using the calculated derivatives. In summary, it works as follows:

1. The weighted sum of the inputs of each unit is computed

$$a_j = \sum_{i=1} w_{ji} z_i \tag{5.8}$$

   and the nonlinear activation function $h$ is applied to give the activation $z_j$ of unit $j$:

$$z_j = h\left(a_j\right). \tag{5.9}$$

2. The error in the output units $\delta_k$ is computed by:

$$\delta_k = y_k - t_k \tag{5.10}$$

3. A particular $\delta$ of a hidden unit is linked to the $\delta_k' s$ of units higher up in the network by the backpropagation formula:

$$\delta_j = h'\left(a_j\right) \sum_k w_{kj} \delta_k \tag{5.11}$$

4. Finally the derivatives are calculated, by simply multiplying the delta at the output end of the weight with the activation at the input end:

$$\frac{\partial E_n}{\partial w_{ji}} = \delta_j z_i$$

Neural networks are used in [SJ04],[FAP08] to recognize falls.

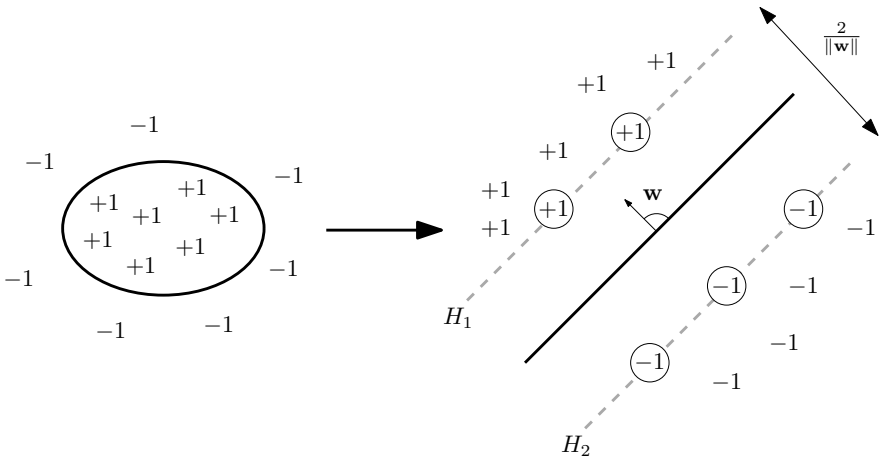



Figure 5.3: Illustration of a Support vector machine separating the data in a higher dimensional space. The two classes are +1 and −1, and not linearly separable (a). After transformation to a higher dimensional space, the data is separable (b). The maximized margin is illustrated as two dashed lines. Its location is determined only by a subset of the data-points, the support vectors (indicated by circles) [Bis06].

### 5.1.3 Support vector machines

Support vector machines (SVMs) are a supervised learning technique, for two class classification, where the two classes are separated by an optimal hyperplane. Optimal in the sense, that the margin – the smallest distance of the hyperplane and the closest data-points – is maximized. However, since the dataset may not be linearly separable in the feature space, it is mapped to a higher dimensional space, where it is linearly separable. This is achieved by applying a kernel function and is referred to as the *kernel trick* [ABR64]. In Figure 5.3 the principle of SVM is illustrated. In the example, the data is not directly linearly separable. However mapped to a higher dimensional space, the data becomes separable. For now, the case where the data is directly linearly separable shall be considered, and this idea extended later to use kernel functions.

Given is a training set $\{\mathbf{x}_n\}$ of size $N$, $\mathbf{x}_i \in \mathbb{R}$ each assigned a label $t_i \in \{-1, +1\}$. The goal is to find the hyperplane separating the two classes, which maximizes the class margin. The hyperplane is defined as:

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \tag{5.12}$$

where $\mathbf{w}$ is the normal and $b$ is the bias of the hyperplane. Points of class +1 lie in the region by $H_1 : \mathbf{w} \cdot \mathbf{x}_i + b \geq 1$ and those of class −1 at $H_2 : \mathbf{w} \cdot \mathbf{x}_i + b \leq -1$. Since data-points have to lie on or outside of the margin, the following constraint has to be satisfied

$$t_n \left(\mathbf{w} \cdot \mathbf{x}_i + b\right) \geq 1,\ n = 1, \ldots, N \tag{5.13}$$

Consider the hyperplanes $H_1$ and $H_2$ in Figure 5.3. Their perpendicular distance from the origin is $\frac{|1-b|}{\|\mathbf{w}\|}$ and $\frac{|-1-b|}{\|\mathbf{w}\|}$ respectively. From this it is obvious, that the margin is $\frac{2}{\|\mathbf{w}\|}$.

Thus the pair of optimal hyperplanes $H_1$ and $H_2$ can be obtained by maximizing $\|\mathbf{w}\|^{-1}$, which is equivalent to minimizing $\|\mathbf{w}\|^2$ [LRR09]:

$$arg\,min_{\mathbf{w}.b}\frac{1}{2}\|\mathbf{w}\|^2 \tag{5.14}$$

subject to the constraints in (5.13).

In order to solve this minimization problem, we will use Lagrangian multipliers. Later this will allow the extension to the nonlinear case [Bur98]. For each constraint in (5.13) a Lagrangian multiplier $a_n \geq 0$ is introduced, what gives:

$$L_P(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} a_i \left(t_i\left(\mathbf{w}\cdot\mathbf{x}_i + b\right) - 1\right) \tag{5.15}$$

Now $L_P$ has to be minimized in respect to $\mathbf{w}$ and $b$ requiring that the derivatives of $L_P$ with respect to $\mathbf{a}$ vanish, subject to the constraint that $a_n \geq 0$. This is a quadratic programming (QP) optimization problem. Equivalently one can maximize $L_P$ subject to the constraints that the gradient of $L_P$ with respect to $\mathbf{w}$, $b$ vanishes, subject to $a_n \geq 0$ [Bur98]. By the requirement, that the gradient of $L_P(\mathbf{w}, b, \mathbf{a})$ vanishes, the following conditions are obtained:

$$\mathbf{w} = \sum_{n=1}^{N} a_n t_n x_n \tag{5.16}$$

$$0 = \sum_{n=1}^{N} a_n t_n \tag{5.17}$$

With (5.16) and (5.17), $\mathbf{w}$ and $b$ can be eliminated from $L_P(\mathbf{w}, b, \mathbf{a})$ , which gives the dual representation of the maximum margin problem, where

$$L_D(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n a_m t_n t_m k\left(\mathbf{x}_n, \mathbf{x}_m\right) \tag{5.18}$$

is maximized subject to (5.17) and $a_n \geq 0$. $k(\mathbf{x}_n, \mathbf{x}_m) = \mathbf{x}_n \cdot \mathbf{x}_m$ is the *kernel function*. Moving to the dual representation, makes it possible to reformulate the model using a kernel function, and thus it can be efficiently applied to high dimensional feature spaces [LRR09]. The general idea of the kernel trick is that if an algorithm is formulated such that the input $\mathbf{x}$ is only present in the form of a dot product, then this can be replaced by any other choice of kernel [LRR09]. The simplest kernel $k(\mathbf{x}_n, \mathbf{x}_m)$ is the identity mapping, linear kernel function in the form of $\mathbf{x}_n \cdot \mathbf{x}_m$. Depending on the feature space, polynomial and radial basis function kernels are applied as well.

Training of the support vector machine amounts to the aforementioned maximization problem of $L_D$. Thus for every training point, a Lagrangian $a_n$ exists, however only points for which $a_n > 0$ contribute to the result, and lie on the hyperplanes. These are the *support vectors*. This is a central property of SVM, since after training, only the support vectors have to be kept, while the other training data can be discarded.

For classification, the sign of (5.12) is evaluated. Therefore the parameters $\{a_n\}$ and the kernel function substitute for $\mathbf{w}$ using (5.16):

$$f(x) = sign\left(\sum_{n=1}^{N} a_n t_n k(\mathbf{x}, \mathbf{x}_m) + b\right) \tag{5.19}$$

What has not been discussed yet, is the case where the data is non-separable. Therefore the hard margin constraint is relaxed by introducing a *slack variables* $\xi_n \geq 0$ for each training data point [CV95]. The relaxed form of the classification constraints (5.13) is:

$$t_n y(\mathbf{x}_n) \geq 1 - \xi_n. \tag{5.20}$$

Since misclassified points will have $\xi_n > 1$, $\sum_n \xi_n$ is an upper bound for the number of training errors. The previous minimization problem in (5.14) is reformulated as

$$C\sum_{n=1}^{N} \xi_n + \frac{1}{2}\|\mathbf{w}\|^2 \tag{5.21}$$

with the parameter $C > 0$ controlling the trade-off between minimizing the training errors and allowing misclassification. Higher values of $C$ correspond to a higher penalty for errors. The corresponding dual Lagrangian is the same as in the separable case, only the constraints have changed:

$$0 \leq a_n \leq C \tag{5.22}$$

$$\sum_{i=1}^{N} a_i t_i = 0 \tag{5.23}$$

Unseen data is again predicted with (5.19). What remains to be calculated is the bias $b$. Support vectors where (5.22) holds, satisfy

$$t_n y(\mathbf{x}_n) \geq 1 - \xi_n \tag{5.24}$$

and have $\xi_n = 0$ [Bis06], so that they lie on the margin and will satisfy

$$t_n\left(\sum_{m\in\mathcal{S}} a_m t_m k(\mathbf{x}_n, \mathbf{x}_m) + b\right) = 1 \tag{5.25}$$

where $\mathcal{S}$ is the set of support vectors. A numerically stable solution is found, by averaging

$$b = \frac{1}{N_{\mathcal{M}}}\sum_{n\in\mathcal{M}}\left(t_n - \sum_{m\in\mathcal{S}} a_m t_m k(\mathbf{x}_n, \mathbf{x}_m)\right) \tag{5.26}$$

where $\mathcal{M}$ denotes the set of points where $0 \leq a_n \leq C$ [Bis06].

The two class SVM can be generalized to a multi-class classification. In the commonly employed *one-versus-the-rest* approach one SVM for each of the $K$ classes is built. The model is trained with data from class $\mathcal{L}_k$ as positive examples and the rest of the data as negatives. In the *one-versus-one* approach, $K(K-1)/2$ different two-class SVMs are trained on all possible class-pairs. Data is classified according to the class with the most "votes".

Lu¨trek et al. [LK09] have used several two class SVMs to distinguish between normal activities and falling.

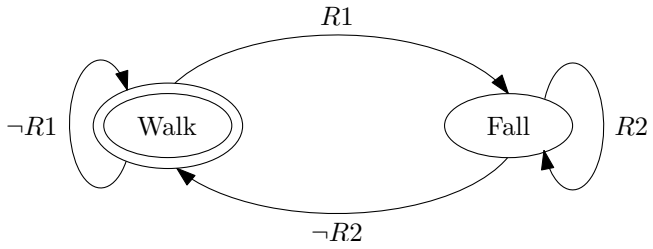



Figure 5.4: A simple 2 state fall detection FSM with the states *Walk* and *Fall*, and $Walk$ being the initial state [VMS07].

### 5.1.4 Finite state machines

A finite state machine (FSM) is composed of a finite number of states, transitions between these states, and assigned to the transitions conditions that trigger state changes [LRR09]. A FSM is typically represented by using a state transition diagram as shown in Figure 5.4. The quintuple $(\Sigma, \mathcal{S}, s_0, \delta, f)$ describes the FSM, with:

- $\Sigma$, the input alphabet
- $\mathcal{S}$, a finite set of possible states
- $s_0 \in \mathcal{S}$, the initial state
- $\delta(\mathbf{x}, q)$, $\mathbf{x} \in \Sigma$, $q \in \mathcal{S}$, the state transition function: $\delta(\mathbf{x}, q) : \mathcal{S} \times \Sigma \to \mathcal{S}$
- $\mathcal{F} \subseteq \mathcal{S}$ is the (possibly empty) set of final states

FSM are based on a set of fully observable states, input symbols and state transitions, and can thus be learned from training data are deterministic and computationally efficient [LRR09]. FSM fail to capture the hierarchical property of events and do not capture uncertainty. While extensions have been presented that address these issues of FSM, other models that are well adapted to such aspects offer more general solutions – such as Hidden Markov Models [LRR09].

In [VMS07] a two state FSM with "Walk" and "Fall" states is implemented.

### 5.1.5 Hidden Markov Models (HMM)

Hidden Markov models (HMM) are directed tree structured graph models, where it is assumed that future predictions are independent of all but the most recent observations. This is in contrast to FSM, where observations are considered as independent and identically distributed [Bis06]. HMMs however exploit the fact that previous states, give a strong clue on what the next state might be.

The foundation of HMM are Markov chains. A first-order Markov chain is a random process with the property, that the next state only depends on the current state, it is thus independent on all, but the most recent state. This is also called the Markov property. If we denote the $N$ possible states $\mathcal{S} = \{S_1, \ldots, S_N\}$ and the state at time $t$ is as $q_t$, the conditional distribution of observing state $S_j$ at time $t$ is given as:

$$p(q_t = S_j | q_{t-1} = S_i, q_{t-1} = S_k, \ldots) = p(q_t = S_j | q_{t-1} = S_i). \tag{5.27}$$

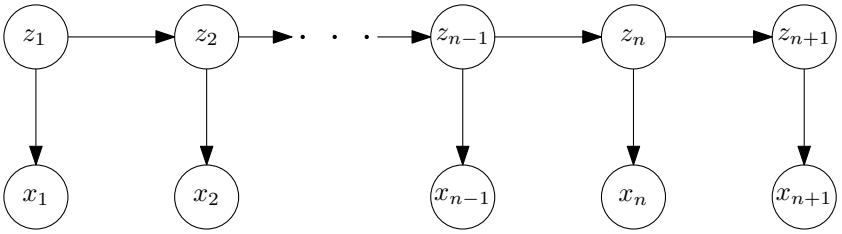



Figure 5.5: Sequential data is represented via latent variables $\mathbf{z}_n$, that form a Markov chain. Each $\langle \mathbf{z}_n, \mathbf{x}_n \rangle$-pair represents a single "time-slice" of the model.

Hence, the overall probability distribution of the states depends on the initial state probabilities

$$\pi_i = p(q_1 = S_i) \tag{5.28}$$

and the set of state transition probabilities:

$$a_{ij} = p\left(q_{t+1} = S_j | q_t = S_i\right), \;\; 1 \leq i, j \leq N \tag{5.29}$$

for which the following stochastic constraints apply:

$$a_{ij} \geq 0 \tag{5.30}$$

$$\sum_{j=1}^{N} a_{ij} = 1 \tag{5.31}$$

Moving to higher-order Markov chains, where the current state depends on more than the most recent observation are not an option, since the model parameters grow exponentially with $M$, and thus applying the model becomes unfeasible for large values of $M$. By introducing additional latent variables, a rich class of models can be constructed out of simple components [Bis06]. So for each observation $\mathbf{x}_n$ a corresponding latent variable $\mathbf{z}_n$, with possibly different dimensionality and type, is introduced. It is assumed that the latent variables $\mathbf{z}_n$ form a Markov chain. The resulting model is the so called *state space model*, and is illustrated in Figure 5.5. This models a double stochastic embedded process with an underlying stochastic process that is not observable, but can be observed through another set of stochastic processes that produce the observation sequence.

A HMM is described by the triple $\lambda = \langle A, B, \pi \rangle$, where

- $A$ is the state transition matrix $A = \{a_{ij}\}$. The $a_{ij}$ are the state transition probabilities, defined as in (5.29).
- $B = \{b_{jk}\}$ is the observation symbol probability distribution in state $j$, for the observed variable $\mathbf{x}_k$:
$$b_{jk} = p(\mathbf{x}_k | q_t = S_j) \tag{5.32}$$
- $\pi = \{\pi_i\}$ is the initial state distribution as defined in 5.28.

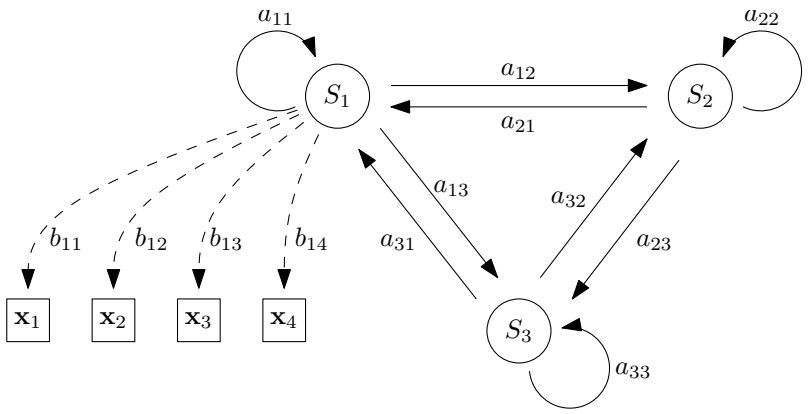



Figure 5.6: Transition diagram for a three state hidden Markov model with four observations. The circles correspond to the hidden states, the lines denote the transition probabilities between the states, the dashed-lines the observation symbol probabilities $b_{jk}$. In order to facilitate reading, only $b_{1k}$ for $S_1$ are illustrated.

With the number of states $N$ and the number of observations $M$ given, HMMs can be used as a model of how a given observation sequence $\mathcal{O} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ was generated by an appropriate HMM [Rab89].

Figure 5.6 shows the transition diagram for a three state model, with four possible observations.

In [Rab89], the three basic problems for HMMs are described:

1. **Evaluation:** Given the model $\lambda$ and an observation sequence $\mathcal{O}$, how can the probability that the observation sequence was produced by that model $p(\mathcal{O}|\lambda)$ be computed? In other words: how well does a given model match a given sequence of observations? A solution to this is given by the forward-algorithm. The forward variable $\alpha_t(i)$ is defined as:

$$\alpha_t(i) = p\left(\mathbf{x}_{1\ldots t}, q_t = S_i|\lambda\right) \tag{5.33}$$

where $\mathbf{x}_{1\ldots t}$ is the observation sequence until time $t$. This can be iteratively solved for $\alpha_t(i)$ as follows:

(a) Initialization

$$\alpha_1(i) = \pi_i b_i(\mathbf{x}_1) \quad 1 \leq i \leq N \tag{5.34}$$

(b) Iteration

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i)\, a_{ij}\right] b_j(\mathbf{x}_{t+1}) \quad 1 \leq t \leq T-1 \tag{5.35}$$

(c) Termination

$$p(\mathcal{O}|\lambda) = \sum_{j=1}^{N} \alpha_T(i) \tag{5.36}$$

2. **Optimization:** Given an observation sequence $\mathcal{O}$ and model $\lambda$, how can the corresponding optimal state sequence $Q = \{q_1, q_2, \dots\}$ be estimated? This requires some kind of optimality criterion. A solution to this problem is offered by the Viterbi algorithm [Rab89].

3. **Training:** Given the observation sequence $\mathcal{O}$, how can the model parameters $\lambda = \langle A, B, \pi \rangle$ be adjusted to maximize $p(\mathcal{O}|\lambda)$? In some cases [TM06] the state transition matrix is set up manually. The Baum-Welch algorithm, an application of expectation-maximization, offers a solution for estimating $A$ and $B$ [Rab89].

In [TM06] a Hierarchical Hidden Markov Model (HHMM) is used to model motion. The HMMs in the first layer model the elementary behavioral pattern. For each of the three motion patterns "Is walking", "Is falling", "Is Lengthened" a HMM is trained and, given the input sequence, evaluated. The first layer output is used as input to the two second layer HMMs for the "WALK" and "FALL" case.

While several extensions [LRR09], like HHMM, have been suggested to allow modeling of complex event compositions, Hidden Markov Models tend to become unmanageable [LRR09]. The most critical issue with HMMs is that they require a large amount of training data, which is in most cases – as in the example of fall detection – not available. However, in case of less complex events, the model parameters can also be estimated empirically as in [TM06].

### 5.1.6 Fuzzy inference

Fuzzy logic [Zad65] allows reasoning with imprecise concepts, much like the way humans do [MP05]. It is based around the concept of fuzzy sets. In contrast to classical set theory, where an element is either member of the set or not, in fuzzy set theory for each element a membership grade is given. Formally a fuzzy subset $A$ in space $\mathcal{X} = \{x\}$ is characterized by a *membership function* $f_A(x)$, which associates a number in the interval $[0, 1]$ with each point in $\mathcal{X}$:

$$f_A(x) \to [0, 1], \quad \forall x \in \mathcal{X}. \tag{5.37}$$

where a value close to 1 indicates a high grade of membership. While this resembles a probability function to some degree, there are essential differences between these concepts. Probability theory describes the uncertainty of the occurrence of a particular event, while fuzzy set theory describes the degree in which the event occurs [Kos90]. Various operators are defined for fuzzy sets [Zad65]:

**Complement** The complement of a fuzzy set $A$, denoted by $A'$ is

$$f_{A'} = 1 - f_A. \tag{5.38}$$

**Containment** Fuzzy set $A$ is contained in set $B$ if and only if $f_A \leqq f_B$

**Union** The union $C$ of sets $A$ and $B$, written as $C = A \cup B$ is given as

$$f_c(x) = \max(f_A(x),\, f_B(x)), \quad \forall x \in \mathcal{X} \tag{5.39}$$

**Intersection** The intersection $C = A \cap B$ of fuzzy sets $A$ and $B$ is given as

$$f_c(x) = \min\left(f_A(x),\, f_B(x)\right), \quad \forall x \in \mathcal{X} \tag{5.40}$$

Complement, union and intersection are defined such that they correspond to the Boolean logic operators negation $\neg$ , disjunction $\vee$ and conjunction $\wedge$, respectively. Input membership functions generally describe linguistic variables such as very low, low, medium, and high.

A Mamdani type fuzzy inference system is composed of the following steps [MA95]:

1. Determination of a set of fuzzy rules in the form of: IF $A$ THEN $B$, or short $A \Rightarrow B$, where both $A$ and $B$ are linguistic terms defined by fuzzy sets. $A$ is called the antecedent and $B$ the consequent.

2. "Fuzzyfication" of the input by applying the membership functions for each input.

3. Implication of the antecedent to the consequent, where the consequent membership function is reshaped using a function associated with the output of the antecedent.

4. Aggregation of all consequents. When all consequents are computed, the result of the fuzzy inference is in the form of one fuzzy set per output variable, by application of the union operator on all consequents [PGK+09].

5. Defuzzyfication of the aggregated outputs. This step yields a single valued output, for example by computing the centroid of the aggregated fuzzy set.

To illustrate the inference process, the fuzzy inference system employed by Anderson et al. [ALK+09] may be considered. A 24 rules inference system, with the antecedent variables *centroid*, *eigen-based height*, *similarity to ground plane normal* (as described in chapter 4) and three consequents *upright*, *in-between* and *on-the-ground* has been built. The fuzzy sets for the antecedents are mappings to the values: $\{L, M, H\}$, low, medium and high, and for the consequents the values $\{V, L, M, H\}$ (very low, low, medium and high). The rules are formulated like:

$$\begin{aligned} \text{IF } \textit{centroid} = H \text{ AND } \textit{height} = H \text{ AND } \textit{similiarity} = H \text{ THEN} \\ \textit{upright} = L, \textit{in-between} = V, \textit{on-the-ground} = V \end{aligned} \tag{5.41}$$

## 5.2 Evaluation

Choosing an event model is crucial to successfully classify events in a specific domain. The $k$-Nearest Neighbor approach is straightforward to apply however, it requires the whole training dataset for evaluation. Furthermore it is not possible to model temporal event composition. Though Neural Networks can be used as arbitrary approximators, their application is cumbersome. They require a large training dataset as well as appropriate preprocessing of the training samples to overcome the problem of overfitting to the training data. In order to successfully apply support vector machines, the choice of the kernel is

crucial. As for Neural Networks, the size and distribution of the training dataset are limiting factors. The problem with Hidden Markov Models lies in the Markov property, namely that the next state only depends on the current state. Additionally, HMMs require a large amount of parameters to be learned and thus a large training dataset is crucial. It is claimed [ALKS08, ALK+09], that fuzzy inference is superior to more complex probabilistic models such as HMMs and Neural Networks. The first problem noted earlier, is the requirement of those models for a large amount of labeled training data (which are not available in the case of fall detection). Fuzzy systems on the other hand can easily incorporate domain expert knowledge in the design of the membership function. Additionally modifying the fuzzy rules is a straightforward process. HMMs, SVMs and Neural Networks have to be retrained to incorporate new knowledge. Further Anderson et al. argue that the confidence output in the fuzzy system can be understood and reliably used to reject a wide range of unknown activities [ALK+09]. With the small-sized dataset that is available for fall detection it is proposed to apply a fuzzy inference system for classifying fall incidents.

# Chapter 6

# Experiments

The proposed system is designed to be capable of monitoring the well being of persons in real time. Thus the focus of the proposed system is on simplicity, computational efficiency, reliability and extensibility. This chapter starts with an introduction of the system in Section 6.1. Evaluation results and a comparison with previous work is presented in Section 6.2.

## 6.1 Proposed approach

The workflow of the proposed early fusion system is illustrated in Figure 6.1. In previous research, early fusion clearly outperformed late fusion [ZMK10]. For each camera, the person silhouette is extracted. These silhouettes are used to obtain a voxel space representation of the object of interest. In this global 3D world coordinate system, view-invariant features are extracted that estimate the human posture. Finally, fall confidence scores are estimated by a fuzzy logic inference system.

In order to estimate the 3D silhouette, the cameras are calibrated using the method by Bouguet [Bou99]. It efficiently supports the enhanced camera model presented in Section 3.

The 2D silhouettes are extracted using the color mean and variance approach (Section 4.1.1) with an additional shadow and highlight detection step in normalized RGB space (Section 4.1.4). The Gaussian mixture model and the Codebook model have been evaluated as well. However, they have been rejected in favor of the computationally less expensive CMV–approach. To remove noise, morphological operations are applied.

As described in Section 4.2, the 3D reconstruction of the person is based on a high performance Shape from Silhouette approach. The $6 \times 4.5 \times 2m$ observation volume is sampled in a regular $50 \times 50 \times 50mm$ grid. In a previous work [ZMK10] the voxel size was twice as large. The grid size was chosen as small as possible without having a notable impact on the run-time.

Selecting features is a crucial part in the event recognition process [RSV05]. In Section 4.3, features that are commonly used in the fall detection and activity recognition literature have been presented. Of these, the following features have been chosen for evaluation:

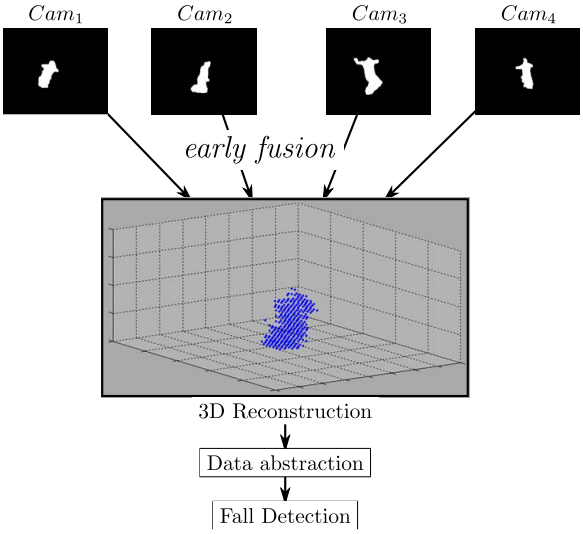



Figure 6.1: The work-flow of the proposed early fusion approach.

1. Bounding-box aspect ratio
2. Orientation of the ellipse major axis in respect to the ground plane normal
3. Ellipse axis ratio
4. Motion speed
5. Centroid speed
6. Centroid height
7. Unexpectedness based on the accumulated hitmap

Features are extracted from the voxel reconstruction in each frame, and fed into a fuzzy inference system. To achieve real-time capable performance, the proposed fall detector is implemented in C++. The current system runs at a frame rate of $\sim$ 22 fps on an Intel Core2 Duo (E7300@2.66GHz) processor. Previously, a MATLAB implementation has been presented [ZMK10], working at 5 fps, but with a half the 3D resolution.

The fuzzy classification procedure assigns a membership degree for each feature for each class. Contrary to previous work, Gaussian membership functions, which can be learned from training data, are proposed instead of the empirically established trapezoid functions membership functions.

$$gauss_{mf}(x) = e^{-\frac{(x-\mu)^2}{\sigma}}$$

Since the normal distribution, does not take into account the nature of the features, a simple extension, an asymmetric Gaussian membership function is suggested. To motivate this idea, consider for example feature 2, the axis orientation. For an angle of 90° the trained membership function $gauss_{mf}$ yields a membership confidence of 0.8881 in the postfall class.

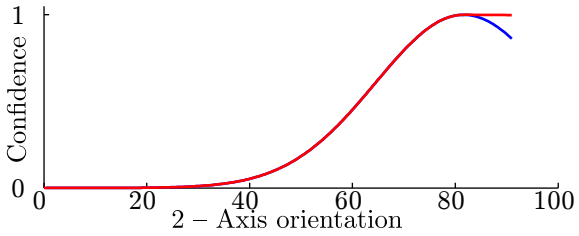



Figure 6.2: Plot of trained Gaussian membership functions $gauss_{mf}$ (red) and $gauss^2_{mf}$ (blue) of the axis orientation feature in the postfall case.

The asymmetric Gaussian membership function $gauss^2_{mf}$ is defined by $(\sigma_{low}, \mu, \sigma_{high})$:

$$gauss^2_{mf} = \begin{cases} e^{-\frac{(x-\mu)}{\sigma_{low}}^2} & \text{if } x < \mu \\ e^{-\frac{(x-\mu)}{\sigma_{high}}^2} & \text{otherwise} \end{cases}$$

Based on the trained parameters for $gauss_{mf}$, the parameters have been manually estimated. Plots of the adjusted membership functions are shown in Figure 6.3. The final class membership degree is computed as the mean of the membership degree of all features.

## 6.2 Evaluation

A dataset consisting of 73 sequences, which follows the scenarios described in [NFR+07], that have already been used for evaluation in previous works [ZMK10, ZZK10], was used for the evaluation. Five actors have simulated a total of 49 falling and 24 non-falling sequences in a laboratory setup. There are 17 forward and 15 backward falls, 13 lateral falls, 6 syncope sequences, 8 involving a chair and 4 falls where the subject could recover. A summary of the sequences is given in Table 6.1. The sequences were shot with four un-synchronized cameras with a $288 \times 352$ resolution at 25 frames per second. The dataset ground truth is labeled according to the four phases proposed by Noury et al. [NRB+08]:

**1 – prefall** normal activity

**2 – critical** the actual fall

**3 – postfall** on-the-ground after a fall

**4 – recovery** getting up, with or without help

Providing accurate per-frame labels for the dataset is difficult, since there is no distinct point at which normal behavior ends, and the critical phase starts. The dataset was labeled such that the critical phase begins whenever a change in acceleration and angle is visible, and ends with the persons head touching the ground. The recovery phase is starting with the first push off the ground.

As suggested in [NFR+07], the classification performance is measured in terms of the classifier being able to positively recognize fall and non-fall events. The event recognition output can thus be one of:

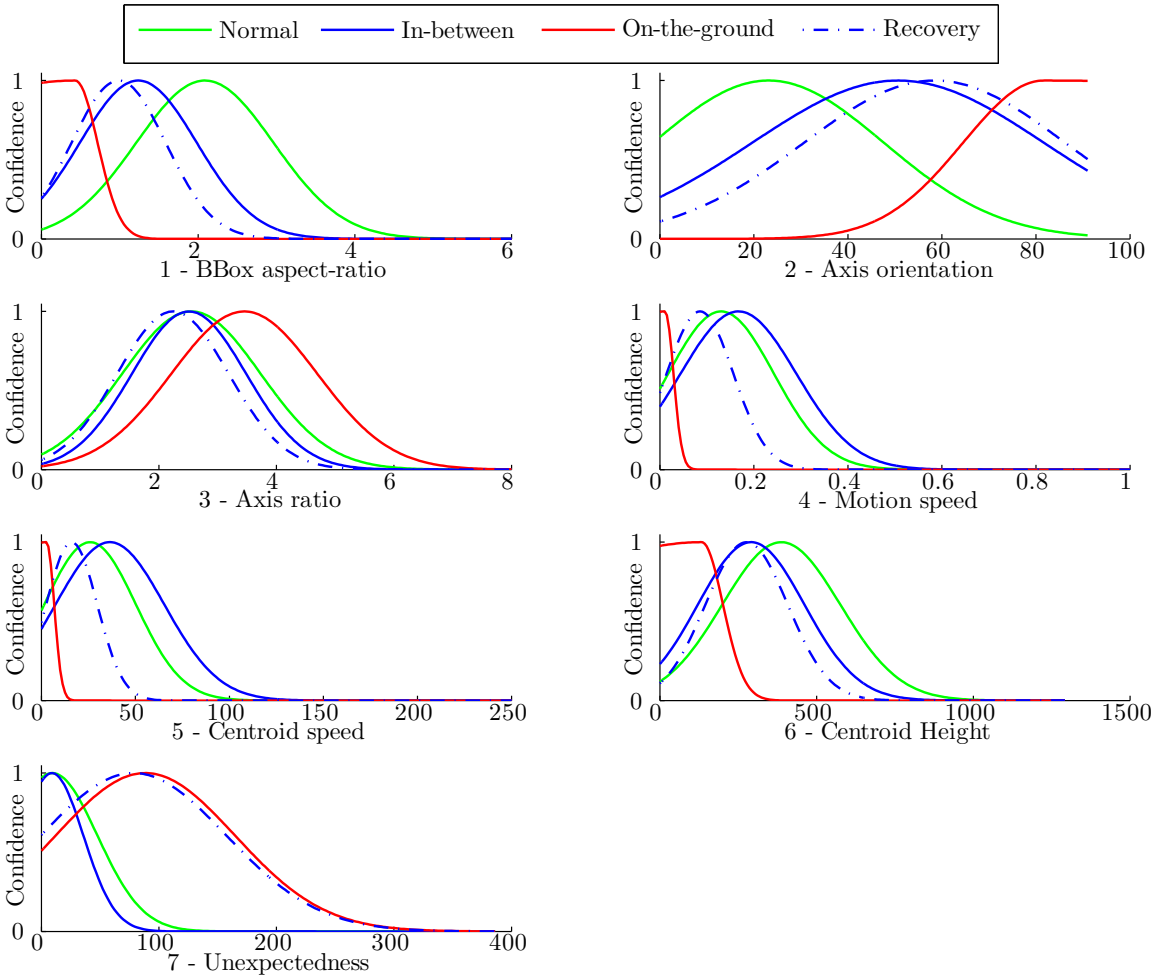



Figure 6.3: Plot of trained Gaussian membership functions for the four phases plotted for each feature.

| Category | Name | Outcome | # |
|---|---|---|---|
| **Backward fall** | Ending sitting | Positive | 4 |
| | Ending lying | Positive | 4 |
| | Ending in lateral position | Positive | 3 |
| | With recovery | Negative | 4 |
| **Forward fall** | On the knees | Negative | 6 |
| | Ending lying flat | Positive | 11 |
| | With recovery | Negative | 5 |
| **Lateral fall** | Ending lying flat | Positive | 13 |
| | With recovery | Negative | 1 |
| **Fall from a chair** | Ending lying flat | Positive | 8 |
| **Syncope** | Vertical slipping against a wall, finishing in sitting position | Negative | 2 |
| **Neutral** | Sit down on a chair then to stand up | Negative | 4 |
| | Lie down then to rise | Negative | 2 |
| | Walk around | Negative | 1 |
| | Bend down, pick something up on the floor and rise again | Negative | 2 |
| | Cough or sneeze | Negative | 3 |

Table 6.1: Overview of the evaluation dataset scenarios with the number of videos [ZMK10].

**True Positive – TP** a fall occurs and is detected

**True Negative – TN** a normal activity is performed and recognized

**False Positive – FP** a fall is wrongly detected

**False Negative – FN** a fall occurred, but is not detected

The output of the fuzzy detector is the membership degree in the range of $[0, 1]$ of the current feature vector in the postfall class, which is thresholded to give the discrete **fall** or **normal** classification results. The receiver operating characteristics (ROC) curve is obtained by plotting *sensitivity* vs. *1-specificity* computed for each threshold of the fall membership. ROC curves have become a popular method for visualizing and comparing classifier performance in machine learning [Faw06].

The *sensitivity* or true positive rate *tpr* of a classifier is

$$sensitivity = \frac{\mathbf{TP}}{\mathbf{TP} + \mathbf{FN}}. \tag{6.1}$$

And the false positive rate $fpr$ is

$$tpr = \frac{\mathbf{FP}}{\mathbf{TN} + \mathbf{FP}} \tag{6.2}$$

Other measures that are associated with ROC curves are $specificity$

$$specificity = \frac{\mathbf{TN}}{\mathbf{FN} + \mathbf{TP}} = 1 - fpr, \tag{6.3}$$

the classifier *accuracy*

$$accuraccy = \frac{\mathbf{TP} + \mathbf{TN}}{\mathbf{FP} + \mathbf{FP} + \mathbf{TN} + \mathbf{FN}}, \tag{6.4}$$

the positive prediction value or *precision*

$$precision = \frac{\mathbf{TP}}{\mathbf{TP} + \mathbf{FP}} \tag{6.5}$$

the $F_1$-score

$$F_1 = 2 \cdot \frac{precision \cdot tpr}{precision + tpr} \tag{6.6}$$

and the area under the ROC curve $AUC$.

In Figure 6.4 the ROC curves for the evaluated feature sets are plotted. As can be seen, the ROC curves are similar. For comparison, in Figure 6.5, the ROC of the results comparing early and late fusion presented in [ZMK10] are given. As you can see, considerably better results have been achieved, as in [ZMK10], regardless of the feature set.

Table 6.2 compares the performance of the feature sets for the optimal threshold – the one that maximizes accuracy. The best score for each performance measure is emphasized.

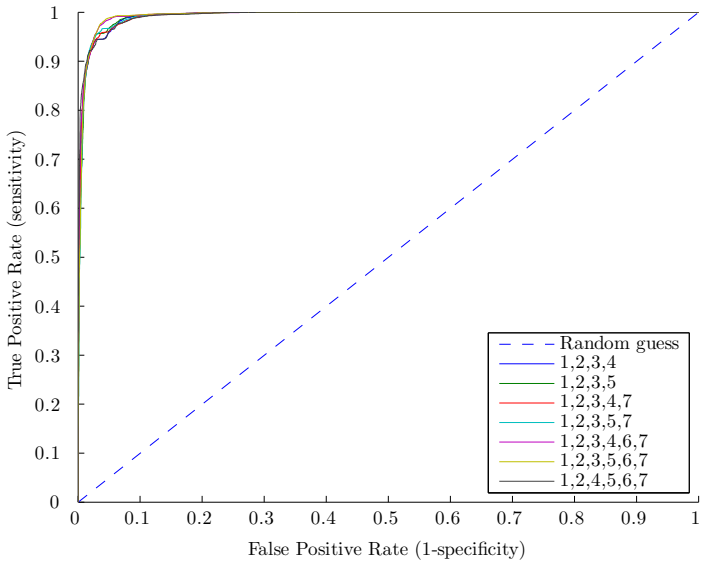



Figure 6.4: The computed fall confidence (blue), with accuracy optimized threshold (green), plotted against the ground truth (red).

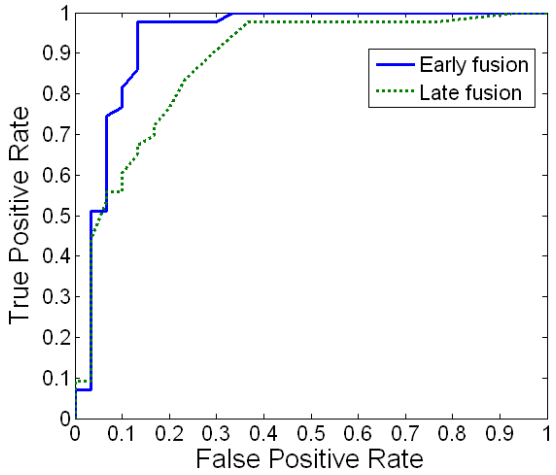



Figure 6.5: ROC of the previously proposed early and late fusion approaches [ZMK10].

| Adjusted Gaussian Membership function | | | | | | |
|---|---|---|---|---|---|---|
| Features | accuracy | precision | sensitivity | specificity | $F_1$-score | AUC |
| 1,2,3,5 | 0.858 | 0.902 | 0.916 | 0.982 | 0.909 | 0.992 |
| 1,2,3,4,7 | 0.828 | **0.916** | 0.898 | **0.985** | 0.907 | 0.993 |
| 1,2,3,5,7 | **0.861** | 0.907 | **0.918** | 0.983 | **0.913** | 0.992 |
| 1,2,3,4,6,7 | 0.825 | 0.915 | 0.896 | **0.985** | 0.905 | **0.994** |
| 1,2,3,5,6,7 | 0.844 | 0.908 | 0.908 | 0.983 | 0.908 | 0.993 |
| 1,2,4,5,6,7 | 0.849 | 0.915 | 0.912 | **0.985** | **0.913** | 0.993 |

Table 6.2: Evaluation results using the adjusted Gaussian membership functions. The best values for each measure are emphasized.

For each measure, the optimal value is 1. As in the ROC plot, it can be seen that the feature sets result in similar detection performance. The best results however are achieved with the proposed unexpectedness measure (feature 7). The highest $AUC$, $precision$ and $specificity$ are obtained when incorporating the unexpectedness. The quality of the previously proposed motion speed feature (4) in comparison to the easier to compute centroid speed (5) has been evaluated as well. Similar performance was achieved using both features. However the centroid speed can be computed more easily and shows slightly better results in combination with other features.

Using the same feature set $(1, 2, 3, 4)$ as in previous experiments [ZMK10] the recognition quality could be increased. An $AUC$ of up to 0.994, compared to the 0.935 in [ZMK10], is achieved.

A direct comparison of the ground truth and the computed fall confidence values for the first three sequences is shown in Figure 6.6. The computed fall confidence is plotted in blue and the ground truth data in red. A ground truth value of 1 indicates a fall in the ground truth data. For illustration of the ground truth, the critical phase is plotted at 0.5 and the recovery phase at 0.75. As noted above, a crisp distinction between *critical*, *postfall* and *recovery* is not easily definable, and this is where classification errors occur. As can be seen in the plots, all fall scenarios are detected, however, the fall is detected with a delay of a few frames.

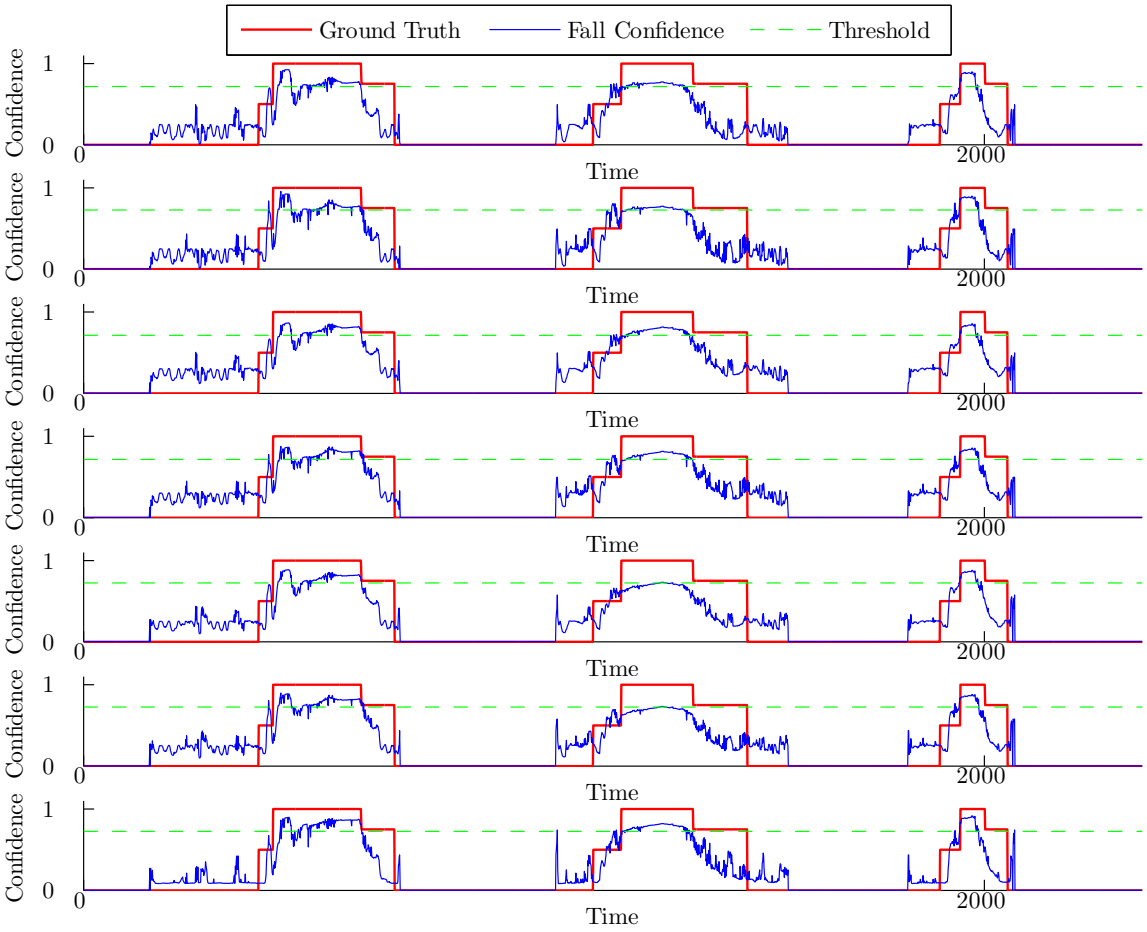



Figure 6.6: The computed fall confidence (blue), with accuracy optimized threshold (green), plotted against the ground truth (red). Feature sets from top to bottom: $(1, 2, 3, 4)$, $(1, 2, 3, 4, 7)$, $(1, 2, 3, 5, 7)$, $(1, 2, 3, 4, 6, 7)$, $(1, 2, 3, 5, 6, 7)$

Nevertheless, all fall incidents in the test-set were successfully classified as such, regardless of the feature set.

As mentioned earlier, the results presented here outperform previous ones in [ZMK10] with an $AUC$ of 0.994. As other authors use different datasets and report per incident as opposed to the per frame evaluation given here, results cannot be compared straightforward. Using a similar setup, Anderson et al. [ALK+09] obtain higher sensitivity 0.976 but lower specificity 0.932. Based on the evaluation of four test sequences Hazelhoff et al. [HHdW08] report the accuracy of fall detection per incident with 100% under "normal" and "realistic" activity, and 55% under "occluding" activity. In a strictly controlled environment, Foroughi et al. [FAP08] achieve similar results (sensitivity: 0.928, specificity: 0.976), as are presented here in an unconstrained environment. Detecting falls with a single camera and a combination of motion quantification and shape features, Rougier et al. [RMSAR07] achieve a sensitivity of 0.882 a specificity of 0.875. Thome and Miguet [TM06] use two perpendicular cameras and the person's principal angle for fall detection.

They report a sensitivity of 0.82 a specificity of $0,983$.

# Chapter 7

# Conclusion

In this thesis, a video understanding framework that allows the reliable detection of falls has been presented. The key considerations are applicability in real world scenarios while achieving a real-time performance. A thorough investigation of the related vision and non-vision based fall detection is the starting point of the here presented research. The main drawback of existing solutions is that they are not independent of camera placement. Further, they cannot deal with occlusions. Therefore they are only applicable in laboratory setups.

With the proposed system, a fall detector that extracts view invariant features in a global 3D coordinate system, based on a voxel space representation of persons is presented. It is shown that the proposed approach automatically and reliably detects falls in real time using a straight forward fuzzy logic approach. The evaluation results show that previous works that uses more complex event understanding schemes such as Hidden Markov models are outperformed by the proposed approach. Fuzzy inference offers a flexible event understanding approach, which makes adding features and rules simple. HMMs in contrast would have to be retrained when new features are added.

Using the accumulated hitmap as an unexpectedness measure, the performance of the feature set is considerably increased. It was shown, that the unexpectedness is an important clue when detecting falls as it allows false positives to be rejected. Such a feature has been ignored in the related work so far.

Since event understanding is just the final step in the processing pipeline, more advanced lower level approaches benefit the classification task. Ranging from fully automatic camera calibration [GKP07], to object segmentation with person identification [CHK+06] and robust tracking approaches [HHdW08]. Occlusions, shadows, different rates of execution and multiple actors are the main problems when working with real world scenarios. The combination with non-vision features like in [TDc05] is another way to tackle these problems.

# Bibliography


[ABR64] A. Aizerman, E. M. Braverman, and L.I. Rozoner. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

[AF08] M. Alwan and R. A. Felder, editors. *Eldercare Technology for Clinical Practitioners*. Humana Press, 2008.

[AKS+06] D. Anderson, J. M. Keller, M. Skubic, X. Chen, and Z. He. Recognizing falls from silhouettes. In *Proceedings of the 28th IEEE International Conference of the Engineering in Medicine and Biology Society*, pages 6388–6391, New York City, 2006.

[ALK+09] D. Anderson, R.H. Luke, J.M. Keller, M. Skubic, M. Rantz, and M. Aud. Linguistic summarization of video for fall detection using voxel person and fuzzy logic. *Computer Vision and Image Understanding*, 113(1):80–89, January 2009.

[ALKS08] D. Anderson, H. R. Luke, J. M. Keller, and M. Skubic. Extension of a soft-computing framework for activity analysis from linguistic summarizations of video. In *Proceedings of the IEEE World Congress on Computational Intelligence*, Hong Kong, June 2008.

[ARK+06] M. Alwan, P.J. Rajendran, S. Kell, D. Mack, S. Dalal, M. Wolfe, and R. Felder. A smart and passive floor-vibration based fall detector for elderly. In *Information and Communication Technologies*, volume 1, pages 1003–1007, April 2006.

[Bau74] B. G. Baumgartner. *Geometric Modeling and Computer Vision*. PhD thesis, Stanford, 1974.

[BBB+99] S. Brownsell, D. A. Bradley, R. Bragg, P. Catlin, and J. Carlier. Do users want telecare and can it be cost-effective? In *Proceedings of the First Joint BMES/EMBS Conference Serving Humanity, Advancing Technology*, page 714, 1999.

[BD01] A.F. Bobick and J.W. Davis. The representation and recognition of action using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(3):257–267, 2001.

[BGRS99] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When Is "Nearest Neighbor" Meaningful? In *Proceedings of the 7th International Conference on Database Theory*, pages 217–235, Jerusalem, Israel, 1999.

[Bis06] C.M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[Bou99] Jean-Yves Bouguet. *Visual methods for three-dimensional modeling*. PhD thesis, California Institute of Technology Pasadena, California, 1999.

[Bur98] C.J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

[BWHK06] P. Blauensteiner, H. Wildenauer, A. Hanbury, and M. Kampel. Motion and Shadow Detection with an Improved Colour Model. In *Proceedings of the of the IEEE Int. Conf. on Signal and Image Processing*, pages 1–12, Hubli, India, 2006.

[CARP07] J. F. Coughlin, L. A. D' Ambrosio, B. Reimer, and M.R. Pratt. Older adult perceptions of smart home technologies: Implications for research, policy & market innovations in healtcare. In *Proceedings of the 29th Annual International Conference of the IEEE EMBS Cite Internationale*, pages 1810 – 1815, August 2007.

[CGP+01] R. Cucchiara, C. Grana, M. Piccardi, A. Prati, and S. Sirotti. Improving shadow suppression in moving object detection with hsv color information. In *Proceedings of Intelligent Transportation Systems*, pages 334–339, 2001.

[CH04] F. Caillette and T. Howard. Real-Time Markerless Human Body Tracking with 3-D Voxel Reconstruction. In *Proceedings of the 15th British Machine Vision Conference*, pages 266–267, 2004.

[CHK+06] X. Chen, Z. He, J.M. Keller, D. Anderson, and M. Skubic. Adaptive silhouette extraction in dynamic environments using fuzzy logic. In *IEEE International Conference on Fuzzy Systems*, pages 236 –243, 2006.

[CKN90] R.G. Cumming, J.L. Kelsey, and M.C. Nevitt. Methodological issues in the study of frequent and recurrent health problems. *Falls in the elderly. Annals in Epidemiology*, 1:49–56, 1990.

[Com04] Commission Of The European Communities. Modernising social protection for the development of high-quality, accessible and sustainable health care and long-term care: support for the national strategies using the "open method of coordination", 04 2004.

[CV95] C. Cortes and V. Vapnik. Support-vector networks. In *Machine Learning*, volume 20, pages 273–297, 1995.

[CVI+03] R. Castelli, M. Vacher, D. Istrate, L. Besacier, and J.-F. Sérignat. Habitat telemonitoring system based on the sound surveillance. In *International Conference on Information Communication Technologies in Health*, pages 11–13, 2003.

[CY99] J.M. Coughlan and A.L. Yuille. Manhattan world: Compass direction from a single image by bayesian inference. In *Seventh International Conference on Computer Vision*, volume 2, pages 941–947, 1999.

[DD95] D.F. Dementhon and L.S. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15(1-2):123–141, June 1995.

[DIM02] J. Deutscher, M. Isard, and J. Maccormick. Automatic camera calibration from a single manhattan image. In *European Conference on Computer Vision (ECCV)*, pages 175–188, 2002.

[DJRW03] T. Degen, H. Jaeckel, M. Rufer, and S. Wyss. Speedy: A fall detector in a wrist watch. In *ISWC '03: Proceedings of the 7th IEEE International Symposium on Wearable Computers*, pages 184–187, 2003.

[Dou00] K. Doughty. Fall prevention and management strategies based on intelligent detection, monitoring and assessment. In *New Technologies in Medicine for the Elderly*, November 2000.

[DRA+04] G. Demiris, M.J. Rantz, M.A. Aud, K.D. Marek, H.W. Tyrer, M. Skubic, and A.A. Hussam. Older adults' attitudes towards and perceptions of 'smart home' technologies: a pilot study. *Medical Informatics and The Internet in Medicine*, 29(2):87–94, 2004.

[FAP08] H. Foroughi, B.S. Aski, and H. Pourreza. Intelligent video surveillance for monitoring fall detection of elderly in home environments. In *International Conference on Computer and Information Technology (ICCIT '08)*, pages 219 –224, Khulna, December 2008.

[Faw06] T. Fawcett. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874, 2006.

[FCLD08] Z. Fu, E. Culurciello, P. Lichtsteiner, and T. Delbruck. Fall detection using an address-event temporal contrast vision sensor. In *Proceedings of 2008 IEEE International Symposium on Circuits and Systems (ISCAS '08)*, pages 424–427, Seattle, WA, May 2008. IEEE Service Center.

[FVB+07] F. Fusier, V. Valentin, F. Bremond, M. Thonnat, M. Borg, D. Thirde, and J. Ferryman. Video understanding for complex activity recognition. *Machine Vision and Applications*, 18(3):167–188, 2007.

[GAA77] C. I. Gryfe, A. Amies, and M. J. Ashley. A longitudinal study of falls in an elderly population: I. incidence and morbidity. *Age Ageing*, 6(4):201–210, 1977.

[GKP07] L. Grammatikopoulos, G.E. Karras, and E. Pets. An automatic approach for camera calibration from vanishing points. *International Journal of Photogrammetry and Remote Sensing*, 62(1):64–76, May 2007.

[GLRS98] W. E. L. Grimson, L. Lee, R. Romano, and C. Stauffer. Using adaptive tracking to classify and monitor activities in a site. In *IEEE Computer Vision and Pattern Recognition*, pages 22–29, 1998.

[GLS+96] R.J. Gurley, N. Lum, M. Sande, B. Lo, and M.H. Katz. Persons found in their homes helpless or dead. *The New England journal of medicine.*, 26(26):1710–1716, Jun 1996.

[Han03] A. Hanbury. A 3d-polar coordinate colour representation well adapted to image analysis. In *Scandinavian Conference on Image Analysis*, pages 804–811. Springer-Verlag, 2003.

[Han08] A. Hanbury. Constructing cylindrical coordinate colour spaces. *Pattern Recognition Letters*, 29(4):494–500, 2008.

[HHD99] T. Horprasert, D. Harwood, and L. S. Davis. A statistical approach for real-time robust background subtraction and shadow detection. In *IEEE ICCV'99 FRAME-RATE Workshop*, pages 1–19, 1999.

[HHdW08] L. Hazelhoff, J. Han, and P. H. de With. Video-based fall detection in the home using principal component analysis. In *Proceedings of the 10th International Conference on Advanced Concepts for Intelligent Vision Systems (ACIVS '08)*, pages 298–309, Berlin, Heidelberg, October 2008. Springer-Verlag.

[HLS04] J.M. Hasenfratz, M. Lapierre, and F. Sillion. A real-time system for full body interaction with virtual worlds. *Eurographics Symposium on Virtual Environments*, pages 147–156, 2004.

[HS81] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1-3):185–203, 1981.

[HS97] J. Heikkila and O. Silven. A four-step camera calibration procedure with implicit image correction. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, pages 1106 – 1112, Washington, DC, USA, 1997. IEEE Computer Society.

[HS03] A. Hanbury and J. Serra. Colour image analysis in 3d-polar coordinates. In *DAGM-Symposium*, pages 124–131, 2003.

[HW03] D. Hong and W. Woo. A background subtraction for a vision based user interface. In *IEEE International Conference on Information, Communications and Signal Processing*, pages 263–267, 2003.

[Joh76] G. Johansson. Visual perception of biological motion and a model for its analysis. *Perception and Psychophysics*, 14(2):201–211, 1976.

[KCHD05] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis. Real-time foreground-background segmentation using codebook model. *Real-Time Imaging*, 11(3):172 – 185, June 2005. Special Issue on Video Object Processing.

[KD06] K. Kim and L. S. Davis. Multi-camera tracking and segmentation of occluded people on ground plane using search-guided particle filtering. In A. Leonardis, H. Bischof, and A. Prinz, editors, *European Conference on Computer Vision ECCV'06*, pages 98 – 109, 2006.

[KE06] K. Klapfer and R. Eichwalder. *Familien- und Haushaltsstatistik: Ergebnisse des Mikrozensus*. Statistik Austria, 2006.

[KG06] R. Kehl and L.J. Van Gool. Markerless tracking of complex human motions from multiple views. *Computer Vision and Image Understanding*, 103(2-3):190–209, November 2006.

[Kos90] B. Kosko. Fuzziness vs. Probability. *International Journal of General Systems*, 17:211–240, 1990.

[Lau91] A. Laurentini. The visual hull: A new tool for contour-based image understanding. In *7th Scandinavian Conf Image Analysis*, pages 993–1002, 1991.

[Lic06] P. Lichtsteiner. *An AER temporal contrast vision sensor*. PhD thesis, Eidgenössische Technische Hochschule ETH Zürich, 2006.

[LK09] M. Luštrek and B. Kaluža. Fall detection and activity recognition methods for the confidence project: A survey. *Informatica*, 33(2):205–212, May 2009.

[LP07] I. Laptev and P. Pérez. Retrieving actions in movies. In *International Conference on Computer Vision*, pages 1–8, 2007.

[LRR09] G. Lavee, E. Rivlin, and M. Rudzsky. Understanding video events: A survey of methods for automatic interpretation of semantic occurrences in video. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 39(5):489 –504, sept. 2009.

[MA95] E.H. Mamdani and S. Assilian. An experiment in linguistic synthesis with a fuzzy logic controller. *International Journal of Man-Machine Studies*, 7:1–13, 1995.

[MCB04] A. Mihailidis, B. Carmichael, and J. Boger. The use of computer vision in an intelligent environment to support aging-in-place, safety, and independence in the home. In *IEEE Transactions on Information Technology in Biomedicine*, volume 8, pages 238–247, 2004.

[MER00] E. D. Mynatt, I. Essa, and W. Rogers. Increasing the opportunities for aging in place. In *Proceedings on the 2000 Conference on Universal Usability*, pages 65–71, 2000.

[MGBB07] B. Michoud, E. Guillou, H. Briceno Pulido, and S. Bouakaz. Real-time and Marker-free 3D Motion capture for Home Entertainment Oriented Applications. In *ACCV'2007 : 8th Asian Conference on Computer Vision*, November 2007. Taux d'acceptation: (32

[MGBB08] B. Michoud, E. Guillou, H. Briceno Pulido, and S. Bouakaz. Silhouettes Fusion for 3D Shapes Modeling with Ghost Object Removal. Technical Report RR-LIRIS-2008-015, LIRIS UMR 5205 CNRS/INSA de Lyon/Université Claude Bernard Lyon 1/Université Lumière Lyon 2/Ecole Centrale de Lyon, July 2008.

[MJD+00] S. J. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler. Tracking groups of people. In *Computer Vision and Image Understanding*, volume 80, pages 42–56, 2000.

[MNC04] S.J. McKenna and H. Nait-Charif. Learning spatial context from tracking using penalised likelihoods. In *International Conference on Pattern Recognition*, volume 4, pages 138 – 141, 23-26 2004.

[MP05] Sushmita Mitra and Sankar K. Pal. Fuzzy sets in pattern recognition and machine intelligence. *Fuzzy Sets and Systems*, 156(3):381 – 386, 2005. 40th Anniversary of Fuzzy Sets.

[NBT08] A. T. Nghiem, F. Bremondo, and M. Thonnat. Shadow removal in indoor scenes. In *IEEE Int. Conf. on Advanced Video and Signal based Surveillance*, pages 291 – 298, Santa Fe, USA, Sep 2008.

[NCM04] H. Nait-Charif and S. J. McKenna. Activity summarisation and fall detection in a supportive home environment. In *Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04)*, volume 04, pages 323–326, 2004.

[NFR+07] N. Noury, A. Fleury, P. Rumeau, A.K. Bourke, G. ÓLaighin, V. Rialle, and j.E. Lundy. Fall detection - principles and methods. *Proceedings of IEEE Conference on Engineering in Medicine and Biology Society*, pages 1663–1666, Aug. 2007.

[NRB+08] N. Noury, P. Rumeau, A.K. Bourke, G.ÓLaighin, and j.E. Lundy. A proposal for the classification and evaluation of fall detectors. *IRBM*, 29(6):340–349, 2008.

[PFME06] G. Perolle, P. Fraisse, M. Mavros, and I. Etxeberria. Automatic fall detection and activity monitoring for elderly. In *Proceedings of MEDETEL*, pages 65–70, 2006.

[PGK+09] S.J. Preece, J.Y. Goulerma, L.P.J. Kenny, D. Howard, K. Meijer, and R. Crompton. Activity identification using body-mounted sensors–a review of classification techniques. *Physiological Measurement*, 30(4):R1–33, April 2009.

[PKS02] M. Peden, K. McGee K, and G. Sharma. *The injury chart book: a graphical overview of the global burden of injuries*. World Health Organization, 2002.

[PLSR08] M. Popescu, Y. Li, M. Skubic, and M. Rantz. An acoustic fall detector system that uses sound height information to reduce the false alarm rate. *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*, pages 4628–4631, August 2008.

[Rab89] Lawrence R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, volume 77, pages 257–286, 1989.

[RE95] P.L. Rosin and T. Ellis. Image difference threshold strategies and shadow detection. In *British Machine Vision Conference*, pages 347–356, 1995.

[RFW+98] J.A. Rizzo, R. Friedkin, C.S. Williams, J. Nabors, D. Acampora, and M.E. Tinetti. Health care utilization and costs in a medicare population by fall status. *Medical Care*, 36(8):1174–1188, 1998.

[RMSAR06] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau. Monocular Head Tracking to Detect Falls of Elderly People. In *Proceedings of the 38th IEEE EMBS annual Internation Conference*, pages 6384–6387, 2006.

[RMSAR07] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau. Fall detection from human shape and motion history using video surveillance. In *International Conference on Advanced Information Networking and Applications Workshops (AINAW'07)*, pages 875–880, Washington, DC, USA, May 2007.

[Roj96] R. Rojas. *Neural Networks*. Springer-Verlag, 1996.

[RSV05] P. Ribeiro and J. Santos-Victor. Human activities recognition from video: modeling, feature selection and classification architecture. In *International Workshop on Human Activity Recognition and Modeling HAREM*, pages 61–70, Oxford, September 2005.

[Sal04] E. Salvador. *Shadow Segmentation and tracking in real-world conditions*. PhD thesis, École Plytechnique Fédérale de Lausanne, 2004.

[SCFM06] J. A. Stevens, P.S. Corso, E. A. Finkelstein, and T.R. Miller. The costs of fatal and non-fatal falls among older adults. *Injury Prevention*, 12:290–295, 2006.

[Sen02] A. W. Senior. Tracking with probabilistic appearance models. In *ECCV workshop on Performance Evaluation of Tracking and Surveillance Systems*, pages 48–55, 2002.

[SG99] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 246–252, 1999.

[SJ04] A. Sixsmith and N. Johnson. A smart sensor to detect the falls of the elderly. *IEEE Pervasive Computing*, 3:42–47, 2004.

[Smi78] A. R. Smit. Color gamut transform pairs. In *Proceedings of the 5th annual conference on Computer graphics and interactive techniques*, pages 12–19, 1978.

[SMP05] T. Svoboda, D. Martinec, and T. Pajdla. A convenient multi-camera self-calibration for virtual environments. *PRESENCE: Teleoperators and Virtual Environments*, 14(4):407–422, August 2005.

[ST09] F. Sposaro and G. Tyson. ifall: An android application for fall monitoring and response. In *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, pages 6119 –6122, Sept. 2009.

[TCSU08] P. Turaga, R. Chellappa, V. S. Subrahmanian, and O. Udrea. Machine recognition of human activities: A survey. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(11):1473–1488, September 2008.

[TDc05] B. Uğur Töreyin, Y. Dedeoğu, and A. Enis Çetin. HMM based falling person detection using both audio and video. In *Lecture Notes in Computer Science*, volume 3766, pages 211–220, Berlin, 2005.

[THS99] D.-M. Tsai, H.-T. Hou, and H.-J. Su. Boundary-based corner detection using eigenvalues of covariance matrices. *Pattern Recognition Letters*, 20:31–40, 1999.

[TKBM99] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. In *IEEE Seventh International Conference on Computer Vision*, volume 1, pages 255–261, 1999.

[TM05] N. Thome and S. Miguet. A robust appearance model for tracking human motions. In *AVSS (IEEE International Conference on Advanced Video and Signal-Based Surveillance)*, pages 528 – 533, 9 2005.

[TM06] N. Thome and S. Miguet. A HHMM-Based Approach for Robust Fall Detection. In *9th International Conference on Control, Automation, Robotics & Vision (IEEE ICARCV)*, pages 1– 8, Singapore, December 2006. IEEE CNF.

[Tsa86] R. Y. Tsai. An efficient and accurate camera calibration technique for 3d machine vision. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 364 – 274, 1986.

[TYS+09] T. Tamura, T. Yoshimura, M. Sekine, M. Uchida, and O. Tanaka. A wearable airbag to prevent fall injuries. *Trans. Info. Tech. Biomed.*, 13(6):910–914, 2009.

[UB05] I. Ulusoy and C.M. Bishop. Generative versus discriminative methods for object recognition. In *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 2, pages 258–265, Washington, DC, USA, 2005. IEEE Computer Society.

[VBNL08] P. Van De Ven, A. Bourke, J. Nelson, and G. Ó Laighin. A wearable wireless platform for fall and mobility monitoring. In *PETRA '08: Proceedings of the 1st international conference on PErvasive Technologies Related to Assistive Environments*, pages 1–4, New York, NY, USA, 2008. ACM.

[VMS07] V. Vishwakarma, C. A. Mandal, and S. Sural. Automatic detection of human fall in video. In *Second International Conference on Pattern Recognition and Machine Intelligence*, volume 4815, pages 616–623, Kolkata, November 2007.

[WADP97] C. R. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):780–785, 1997.

[WCH92] J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 14(10):965–980, Oct 1992.

[Wu00] G. Wu. Distinguishing fall activities from normal activities by velocity characteristics. *Hournal of Biomechanics*, 33(11):1497–1500, 2000.

[XZ96] Gang Xu and Zhengyou Zhang. *Epipolar Geometry in Stereo, Motion and Object Recognition*. Springer, 1996.

[YKI92] J. Yamato, J. Ohya K., and Ishii. Recognizing human action in time-sequential images using hidden markov model. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pages 379 –385, 15-18 1992.

[YNC09] M. Yu, S.M. Naqvi, and J. Chambers. Fall detection in the elderly by head tracking. In *15th Workshop on Statistical Signal Processing (SSP '09)*, pages 357 –360, 2009.

[Zad65] L.A. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338 – 353, 1965.

[Zha00] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactionson Pattern Analysis and Machine Intelligence*, 22:1330 – 1334, 2000.

[ZK10] A. Zweng and M. Kampel. Unexpected human behavior recognition in image sequences using multiple features. In *International Conference on Pattern Recognition*, pages 368–371, August 2010.

[Zlo06] H. Zlotnik, editor. *World Population Prospects: The 2006 Revision*. Population Division of the Department of Economic and Social Affairs of the United Nations Secretariat, 2006.

[ZMK10] S. Zambanini, J. Machajdik, and M. Kampel. Early versus late fusion in a multiple camera network for fall detection. In *34th Annual Workshop of the Austrian Association f. Pattern Recognition (ÖAGM 2010)*, pages 15–22, Zwettl, Austria, 2010.

[ZS04] H. Zollner and R. Sablatnig. Comparison of methods for geometric camera calibration using planar calibration targets. In *28th Workshop Austrian Association of Pattern Recognition*, pages 234–244, 2004.

[ZSV04] H. Zhong, J. Shi, and M. Visontai. Detecting unusual activity in video. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 819–826, 27 2004.

[ZZK10] A. Zweng, S. Zambanini, and M. Kampel. Introducing a statistical behaviour model into camera-based fall detection. In *International Symposium on Visual Computing*, pages 163–172, Las Vegas, December 2010.