Die approbierte Originalversion dieser Diplom-/Masterarbeit ist an der Hauptbibliothek der Technischen Universität Wien aufgestellt (http://www.ub.tuwien.ac.at).

The approved original version of this diploma or master thesis is available at the main library of the Vienna University of Technology (http://www.ub.tuwien.ac.at/englweb/).

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Recognition of a vision approach for fall detection using a biologically inspired dynamic stereo vision sensor

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Biomedical Engineering

eingereicht von

Aneta Nowakowska
Matrikelnummer 0542803

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Robert Sablatnig
Mitwirkung: Dipl.-Ing. Stephan Schraml
Dr. A. Nabil Belbachir

Wien, 30.11.2011

(Unterschrift Aneta Nowakowska)

(Unterschrift Betreuung)

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Recognition of a vision approach for fall detection using a biologically inspired dynamic stereo vision sensor

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Biomedical Engineering

by

Aneta Nowakowska
Registration Number 0542803

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Robert Sablatnig
Assistance: Dipl.-Ing. Stephan Schraml
Dr. A. Nabil Belbachir

Vienna, 30.11.2011

(Signature of Author) (Signature of Advisor)

Erklärung zur Verfassung der Arbeit

Aneta Nowakowska
Vorgartenstraße 119/117, 1020 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum)

(Unterschrift Aneta Nowakowska)

Abstract

In this thesis methodologies for learning based fall detection using a dynamic stereo vision sensor are elaborated. The main characteristic of the sensor is that data is generated asynchronously and only pixels with illumination changes produce an output.

To obtain relevant information about the motion of the person, the 3D data is transformed into world coordinates with the help of the camera parameters. Intuitive features describing the pose and motion of the body are proposed and evaluated with regard to their significance for fall detection.

For the evaluation a database of about 200 scenarios containing acted falls and normal movements is prepared. In the case of normal movements particular attention is paid to downwards movements like sitting or lying down, because these are similar to falls and assumed to cause false alarms more likely than walking or upward movements. For the falls it is taken care that they can happen from static poses as well as during normal everyday movements. That means, on the one hand falls are acted from the three poses, standing, sitting and lying. On the other hand also falls happening during movements like walking, sitting down or bending down are recorded.

The recordings are used to generate samples in form of 3 seconds long sliding windows with associated features. The amount of sliding windows collected from all recordings serves as sample set for the training and evaluation. The set of sliding windows is divided into two classes, time windows containing a fall and time windows containing normal movements without fall. The features are investigated with regard to their alternation during fall and no fall time windows.

The classification task is set to distinguish between fall time windows and no fall time windows in unseen samples with the help of the features. Therefore two different learning approaches, namely the Hidden Markov Model and the Multilayer Perceptron are used. A detailed methodology for the usage of the learning models for the fall detection is proposed. Optimal parameter settings for the architecture of the two classifiers are found with the help of cross validation on the sample set. The significance of the particular features for the classification performance is investigated by training and validating classifiers with subsets of the features.

Kurzfassung

Diese Diplomarbeit befasst sich mit den Methoden für die Erkennung von Stürzen unter Verwendung eines optischen, dynamischen Stereo Sensors. Die Haupteigenschaft des Sensors besteht darin, dass die Daten asynchron und ausschließlich durch Pixel die Helligkeitsunterschiede aufweisen generiert werden.

Um Informationen über die tatsächliche Bewegung der Person im Raum zu erlangen, wird die durch den Sensor erhaltene 3D Tiefeninformation, mit Hilfe der Kameraparameter, in Weltkoordinaten transformiert. Intuitive Merkmale welche die Körperposition der Person beschreiben werden präsentiert und hinsichtlich der Signifikanz für die Sturzerkennung untersucht.

Für die Evaluierung werden Stürze sowie alltägliche Bewegungen nachgestellt. In dieser Weise wird ein Daten Set von ungefähr 200 Aufnahmen generiert. Bei den alltäglichen Bewegungen wird darauf geachtet, dass das Daten Set vertikale Bewegungen nach unten wie beispielsweise Hinsetzen oder Hinlegen beinhaltet. Es wird angenommen, dass diese Bewegungen häufig zu Fehlalarmen führen können, da diese Stürzen ähnlicher sind als Bewegungen wie Aufstehen oder Gehen. Stürze werden sowohl aus den statischen Positionen Stehen, Sitzen und Liegen, als auch aus Bewegungen wie zum Beispiel Gehen, Hinsetzen oder Bücken nachgestellt.

Die Aufnahmen werden in überlappende 3 Sekunden lange Zeitfenster aufgeteilt. Die zusammengefasste Menge der Zeitfenster aller Aufnahmen stellt das Probenset für das Training und die Evaluierung dar. Dieses wird in zwei Klassen unterteilt, Zeitfenster die einen Sturz enthalten und Zeitfenster die keinen Sturz enthalten. Die Merkmale werden für beide Klassen von Zeitfenstern hinsichtlich ihres Änderungsverlaufs untersucht.

Die Aufgabe der Klassifizierung besteht darin ein unbekanntes neues Zeitfenster, abhängig davon ob es einen Sturz enthält, einer der beiden Klassen zuzuordnen. Dazu werden zwei unterschiedliche Lernverfahren, das Hidden Markov Model und das Multilayer Perzeptron verwendet. Das methodische Vorgehen für die Anwendung der beiden Lernmethoden für die Sturzdetektion wird schrittweise erklärt. Die optimalen Parametereinstellungen für die Architektur der Lernmodelle werden mit Hilfe von Kreuz-Validierung auf dem Probenset ermittelt. Die Signifikanz der Merkmale für die Modelle wird untersucht indem bestimmte Merkmale ausgelassen und die resultierenden Klassifikations-Ergebnisse verglichen werden.

Contents

1 Introduction **1**
1.1 Motivation 1
1.2 Definition of Terms 4
1.3 Results 5
1.4 Report Structure 6

2 Related Work **7**
2.1 Wearable device approach 7
2.2 Ambience device approach 9
2.3 Camera based approach 10

3 Methodology **19**
3.1 Dynamic stereo vision sensor for fall detection 20
3.2 Data 23
3.3 Features 25
3.4 Feature processing 35
3.5 Machine learning based fall detection 38

4 Results **51**
4.1 Feature Evaluation 51
4.2 Classification with Hidden Markov Models 53
4.3 Classification with the Multilayer Perceptron 62
4.4 Classification Examples 68

5 Conclusion **75**

List of Acronyms **79**

Bibliography **81**

CHAPTER 1

Introduction

Considering the statistics from eurostat [30], in Europe the working age population is shrinking while the proportion of elderly people is increasing. One of the main reasons for the ageing population is the combination of a decreasing number of births together with a raising life expectancy [61]. Since 1960, the fertility rate of women, which is roughly speaking the average number of children born to a woman over her lifetime, fell from 2.6 children per woman to 1.4 [30]. So in 2005 there were as many children aged less than 15 years, as people aged over 65 years. Each of the both groups accounted for 15.9% of Europeans population. Figure 1.1 shows the proportion of the elderly population to the total population for the different continents over recent years. It can be recognized that Europe has the biggest proportion of elderly people.

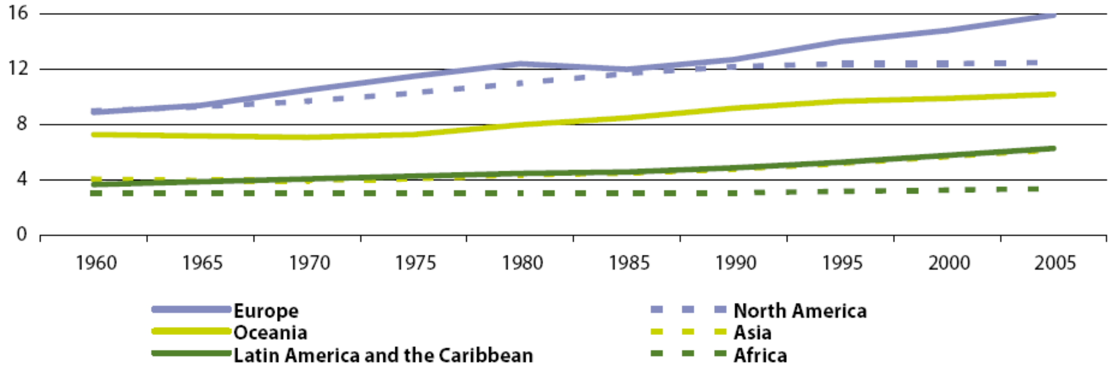


Figure 1.1: Proportion of the population aged 65 and over, in % of the total population [30].

1.1 Motivation

Considering the demographic changes described above, the importance of Ambient Assisted Living (AAL) applications, which aim to prolong the time people can live autonomously in their own home, becomes clear. Enabling elderly to live as long as possible on their own, raises their quality of life and minimizes the costs for care facilities [61].

AAL includes applications which can be integrated in the original environment of elderly people to e.g. raise their security, monitor the medical condition, improve the mobility and facilitate an active life.

The project CARE [7] running under the Ambient Assisted Living Joint Programme [4] aims to realise a monitoring and alarming system for elderly people. The target is an automated recognition and alarming of critical situations, especially falls. Therefore a biologically inspired dynamic stereo vision sensor [8] developed at the Austrian Institute of Technology GmbH (AIT) is combined with an alarming system from Everon Oy/Ab, a Finnish company. The system is designed in a way that the vision sensor has the algorithmic task of detecting falls and generating alarms, while the Everon system is responsible for the telecommunication to forward the alarm to the appropriate instances.

Using vision based monitoring systems the issue of privacy protection has to be considered. A special characteristic of the dynamic stereo vision sensor is that only pixels with illumination changes produce an output which is not a conventional grey-scale or color value, but only a binary value according to an increasing or decreasing illumination.

Although existing preliminary work on vision based fall detection is available (see Chapter 2), there are no actual approved state-of-the-art methods on this topic. Approaches which use conventional cameras have disadvantages concerning the privacy protection. Further the 2D data in not sufficient to reconstruct the actual pose and movements of the body, which is necessary to detect abnormal motion patterns like falls. With the help of the depth information about the distance between the captured object and the camera, it is possible to transform the person's motion history into a 3D world-coordinate representation. Based on this representation, comparable features describing the person's pose and motion can be extracted and used to investigate whether a fall occurred. The dynamic stereo vision sensor is suitable for the application because it provides real time stereo data and consists of a processing unit where algorithms for fall detection including the transformation into world-coordinates, can be implemented.

Scope of Discussion

Within the scope of the CARE project, this work concentrates mainly on the methodological process of extracting relevant information from the 3D data of a person and processing it in a way that models for the classification of movements can be trained automatically to be subsequently used for fall detection.

Although the work is done under the motivation that fall detection algorithms should be implemented on the board of the stereo vision sensor, this thesis describes the general possible approaches for fall detection using the processing power of a standard PC and MATLAB, without regard to the on board real time implementation.

The proposed methods address the extraction of features and the automatic training of learning models based on visual 3D data. Therefore they can be adjusted on the data of other stereo vision systems, like the recently developed Kinect [14] or time-of-flight cameras. The focus of this work is not on the characteristics of the sensor but the processing of person related world-coordinate data for fall detection. Therefore the evaluation is done using different learning approaches rather than different sensors.

Further it is not intended to describe the whole fall detection work flow including preceding segmentation and tracking of relevant objects in the scene, and down-streamed alarm handling with verification. The segmentation and tracking of the person using the dynamic stereo vision sensor is a separate extensive topic. State-of-the-art algorithms have to be adjusted for the specific data characteristics because the sensor does not provide data about static objects. Further the generated data is no grey or color image but an asynchronous stream of events.

The alarm handling and the verication is also a separate topic. It has to be considered depending on the specific application because it deals with tasks like what to do in case of alarms and how to handle false alarms.

Objective

The objectives include first the extraction of relevant features which describe the pose and motion of the body from the Address Event data generated by the dynamic stereo vision sensor. Therefore a database of recorded scenarios including falls and normal movements has to be prepared. Theoretically possible features should be computed and investigated how significant they actually are for fall detection.

As subsequent step, an appropriate processing and representation of the features should be elaborated, to obtain samples as input for the training of machine learning models. Two different machine learning approaches, the Hidden Markov Model (HMM) and the Multilayer Perceptron (MLP) are compared with regard to the applicability for the classification of person movements.

HMMs have been used for temporal pattern recognition problems like speech [46], gesture [60] and handwriting [29] recognition. Since human movements are time dependent, distinguishing falls from normal movements can be also seen as temporal pattern recognition task. Therefore it is assumed that the HMM is an appropriate learning model for fall detection. With the help of the HMM, sequences of observations can be assigned to sequences of hidden states. The HMM interprets that the temporal feature sequence is generated by a hidden state sequence, where the state at a specific time moment depends on the previous state. This fits to the concept, that on the basis of the computed features, a suggestion about the pose and movements of the body is aimed.

The MLP is an artificial neural network which in contrast to the HMM interprets the input feature sequence as combination of features with no explicit sequential aspect. For each defined class of movements, one HMM is trained, which learns the feature scopes of the particular class. Contrary to the HMM, within the artificial neural network approach, both classes are presented to only one MLP, which is trained to give then an appropriate output, depending to which class the input feature combination is assigned. Comparing the results of the MLP and the HMM classifiers enables to investigate whether the sequential time aspect of the learning model leads to better results in the classification.

Further the influences of the feature selection and preparation on the classification performance are investigated.

Main Contribution

The main contribution of this work is, to present a methodology for fall detection based on optical 3D data. The focus is, how visual 3D data can be used, to automatically train machine learning based classifiers for fall detection. The methods are demonstrated on two different machine learning approaches, HMM and MLP. The HMM is chosen because it is able to model the sequential time aspect of the features describing the person's movements. The MLP as an artificial neural network is used because it is biologically inspired and can find complex decision boundaries for the classification of input features.

The data of the dynamic stereo vision sensor has specific characteristics like the asynchronous data generation which requires an adjusted processing. However the main contribution of this thesis focusses not on the sensor characteristics but, on the processing of the 3D data and the recognition of algorithms for machine learning based fall detection.

The first contribution is the proposal of features which describe the pose and movements of the body. Thereby one aim is to investigate the features with regard to their significance for the fall detection. Therefore a preliminary feature evaluation method is introduced and the results are compared to the actual significance of the particular features within the classification models.

Reading the existing works on vision based fall detection, in some cases e.g. [15], [20] and [22], the features and the machine learning approaches are well documented, but the processing and presentation of the features to the learning models are not clearly understandable. That means, it is not well explained how the features are prepared and selected for fall detection, and how the actual input for the learning models exactly looks like. Therefore another important contribution is the presentation of a detailed methodology for the feature processing before they are used as input for the classifiers. To show the importance of this issue, models are trained with different feature preparations and subsets, comparing the resulting classification performances.

Due to the high variability of human movements, analytical methods like thresholds for the features are not sufficiently distinctive to detect falls. In this thesis two different machine learning approaches, HMM and MLP, are proposed for fall detection. Their characteristics are explained with regard to the appliance on the extracted features for fall detection. The machine learning models are trained and evaluated with the help of the database of recorded movements.

1.2 Definition of Terms

In this section, commonly used terms and abbreviations will be introduced and briefly explained.

AE Address Event: Data generated asynchronously by one sensing element (pixel) of the stereo vision sensor. Since the sensor detects only illumination changes an Address Event contains the information about the pixel position and occurrence time [45]. The stereo vision sensor is built of two sensing arrays which both produce Address Events. In following processing steps, a stereo matching algorithm adds depth information to each successfully matched Address Event and disclaims the remaining ones.

HMM Hidden Markov Model: A stochastic model which contains two random processes and is used in temporal pattern recognition to classify sequences of features [46] (see Section

3.5).

MLP Multilayer Perceptron: An artificial neural network composed of an input, a hidden and an output layer of nodes in a directed graph. It maps a set of input data onto a set of associated output values and is used for the classification of input feature sets [49] (see Section 3.5).

CCD Charge-Coupled Device: Device for the movement of electrical charge used for image sensors [28].

CMOS Complementary Metal Oxide Semiconductor: A technology for constructing integrated circuits used for active pixel image sensors where each pixel contains an active amplifier [65].

PCA Principal Component Analysis: A mathematical method, to find the directions within a point cloud, which have the maximal variance [32] (see Section 3.3).

DCT Discrete Cosine Transform: A discrete transformation which expresses a signal as sum of sinusoids of varying magnitudes and frequencies [1], [38].

1.3 Results

The basis for the results generated within this thesis is the database of acted fall and no fall scenarios. Therefore the recordings are subsequently processed with the help of 3 seconds long sliding windows of 0.4 seconds overlapping. In this way a total amount of 7568 time windows with normal movements and 113 time periods with falls is obtained. The whole amount of time windows constitutes the sample set which is used for training and validation.

A number of 8 features are proposed to extract relevant information from the world-coordinate representation of the person related data. To investigate the feature significance for fall detection, the feature alternation during fall and no fall time windows is compared (see Section 3.5 and Section 4.1). The alternation of each time window is plotted in form of histograms for every feature. The visual analysis of the histograms shows, that the fall samples form a cluster and have similar alternations. Based on the visualization, thresholds for each feature are introduced and computed, how well the fall samples can be distinguished from the no fall samples. As characteristic number the decrease of entropy is calculated on the basis of the positive and negative classified samples. The investigation results that three features, namely the highest point, its velocity and the median of the height, are most significant concerning their alternation during the time windows.

The sample set is divided in 10 subsets to investigate the classification performance of the trained classifiers on unseen data with the help of cross validation. Optimal parameter settings for the learning models are found empirically by determining the classification performance with different parameter permutations. The HMM and the MLP are trained using the Discrete Cosine transformed features as input. In the case of the MLP, additionally a simple model is trained using just the alternation of each feature as input.

For the HMM an architecture of 7 hidden states and 8 Gaussian mixtures yields the best results, namely a specificity of 0.987 and a sensitivity of 0.965. The two MLPs yield the best results with 65 hidden neurons for the complex and 1 hidden neuron for the simple MLP. In the case of the complex MLP, the obtained specificity and sensitivity is 0.986 and 0.947, while with the simple MLP, a specificity of 0.965 and a sensitivity of 0.974 is reached.

The influence of the features on the classification performance is investigated by training and cross validating the classifiers, using only subsets of the features as input. The results show, that the significance of the particular features depends on the preprocessing and the classification method.

1.4 Report Structure

After assessing the motivation and introduction into the topic, the existing related work is presented in Chapter 2. This chapter is divided into three main parts describing different fall detection methods according to the used principles and sensors. The first section describes wearable devices like accelerometers and gyroscopes, which are integrated in belts, neck cords or wristbands. The second part details approaches like pressure or proximity mattresses grouped together under ambience devices. For this thesis the existing work about camera based fall detection is mainly relevant, therefore the third section is presenting related works of vision based approaches in more detail.

Chapter 3 explains the methodology of the thesis beginning with an overview of the sensor characteristics and the obtaining data. The second part describes the database of recorded scenarios, which is the basis for the evaluation of the proposed fall detection methods. In this section the features introduced to describe the motion patterns during falls and normal movements are presented. The appropriate processing for quality improvement and an efficient representation of the features are described in Section 3.4. The last section of the methodology deals first with the preliminary evaluation and investigation of the features. Then the basic concepts, including the training and the fall recognition, of the two different machine learning approaches are explained.

Chapter 4 presents the results of the feature evaluation and the classification performance of the two models trained. The results of the feature evaluation are compared to the actual significance of the features within the learned models. The chapter is divided into four sections, where the first section shows the results of the preliminary feature evaluation. The following two sections are of similar structure and present the results for the two models including a parameter and feature evaluation, and characteristics of the classification performance. The last section of the chapter shows the feature scopes of four example scenarios with according classification outputs.

Finally, a conclusion is given in Chapter 5, which discusses insights obtained through the results, and advantages as well as disadvantages of the proposed methodology. Further in the future outlook some ideas are given how the method can be improved and extended.

CHAPTER 2

Related Work

In this chapter an overview of the existing approaches for fall detection of elderly is given. The objective of this chapter is to identify camera based approaches using CCD or CMOS image sensors in opposite of approaches based on accelerometers, tilt, pressure and vibration sensors. [68] proposed a division of fall detection methods according to the used principles and sensors. Depending on how the fall is detected, three approaches are distinguished: wearable device, ambience device and camera or vision based. Figure 2.1 shows the proposed class hierarchy of fall detection methods.

The structure of this chapter goes along with this hierarchy and therefore in the first two sections wearable and ambience devices are explained. The asynchronous temporal contrast vision sensor is referred to the camera based approaches which are discussed in Section 3 of this chapter.

2.1 Wearable device approach

The wearable device approach comprises orientation and/or acceleration sensors embedded in garments and accessories to detect the posture or the motion of the wearer (see Figure 2.1). Accelerometers measure the dynamic acceleration, due to shocks and vibration, together with the static acceleration, caused by the gravitation [42]. Depending on the design and the mounting on the body, orientation sensors, like gyroscopes measure the pose of the person. The orientation sensors are combined with acceleration sensors to gain information about the body's motion and orientation. The solutions differ in the kind of sensors used and the handling. To communicate ideas of existing wearable fall detection devices, a selection of state of the art examples is described in the following.

One of the early works is in [66], where a piezoelectric shock sensor to detect the impact and a mercury tilt switch to measure the person's posture, e.g. lying or standing, are combined in a fall detector device. To detect falls, the device is programmed for each wearer with an individual threshold for the fall impact, to compensate weight and gender variations. When an

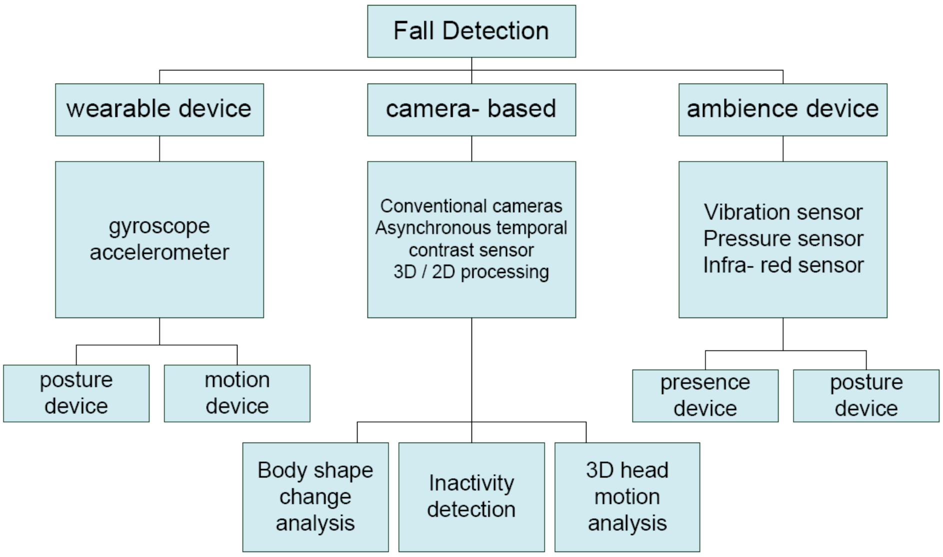


Figure 2.1: Fall detection methods [68] (modified).

impact excesses the threshold, the device monitors the orientation of the wearer to check whether the person is in a lying position.

Later [42] developed a sensor which integrates two orthogonally oriented accelerometers and a micro controller to compute the body's motion. The device is worn under the armpit on the trunk and detects when the velocity exceeds a specific threshold, the sequence from a vertical to the lying posture and the absence of movements after fall. The specificity achieved was 0.83 and the sensitivity was 0.79 on a set of 15 situations performed by 10 people, including backward fall, forward fall, syncope and normal conditions. The system failed to detect a syncope ending in sitting, and forward falls with recover were detected as false positives.

[34] divide human activities into two categories: static postures and dynamic transitions between these postures. Acceleration and angular velocity are measured with two accelerometers and gyroscopes. The sensors are attached at the chest and the thigh. This enables a more accurate identification of the posture than using one sensor. Data is collected for one second intervals and for each interval the change of the sensor readings is used first to decide whether the person is static or dynamic during the present time segment. Then, for static segments, the specific postures, including standing, bending, sitting, and lying are determined. If a lying posture is detected, it is examined whether the transition to the lying posture is an intentional movement by analyzing the previous 5 seconds of data. If the transition is unintentional, it is flagged as a fall. To differentiate intentional and unintentional transitions thresholds to peak values of acceleration and angular rate are applied. This algorithm can distinguish fast sitting down from falls and detects also falls on stairs where the orientation of the person is not horizontal afterwards. The evaluation highlighted that the system could detect typical falls like backward, forward and even falling on stairs sufficiently.

Other approaches are to mount accelerometers and orientation sensors on the waist [37], into a hearing aid housing [35], at the sternum [10], in a jacket [24] and even in walking sticks [2].

A novel wearable device approach is to combine acceleration sensors with air pressure sensors [36], which are able to detect a change of the air pressure of as low as 0.03 hpa. This corresponds to an altitude change of about 25 cm and therefore changes in the height of the person can be incorporated into the fall detection algorithm.

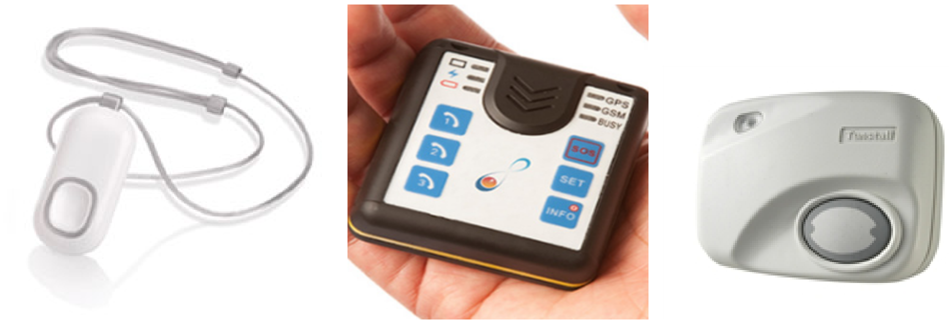
Commercial products

Commercial fall detectors can be can be worn on a cord around the neck, clipped to the belt or worn with a waist pouch accessory.

As example Philips Lifeline [19] provides alert services like a personal help button on a neck cord with the option for automatic fall detection called AutoAlert Pendant (see Figure 2.2(a)).

Gemshield Personal Safety Device [56] manufactured by Skyguard Ltd. is a compact personal safety GPS device and locator, which can be set to detect rapid movement followed by sudden impact for example in the event of a fall (see Figure 2.2(b)).

Tunstall Healthcare Ltd. manufacture a belt-worn fall detector [27] triggered by a change of angle and impact of a fall. It emits a pre alarm buzzer to alert user that the system is about to raise an alarm so that false alarms can be cancelled by the wearer (see Figure 2.2(c)).



(a) AutoAlert Pendant from Philips [19]
(b) Gemshield from Skyguard [56]
(c) Tunstall fall detector [27]

Figure 2.2: Commercial wearable fall detectors

2.2 Ambience device approach

The ambience device approach is to install amongst others vibration, pressure or infra - red sensors on walls, floors or furniture to detect person related data when the person is close to them.

[3] use a passive piezoelectric transducer to measure floor vibrations caused by human activities. Based on experiments they observed that there are significant differences in the patterns of vibrations induced on the floor by different activities and therefore falls generate a different

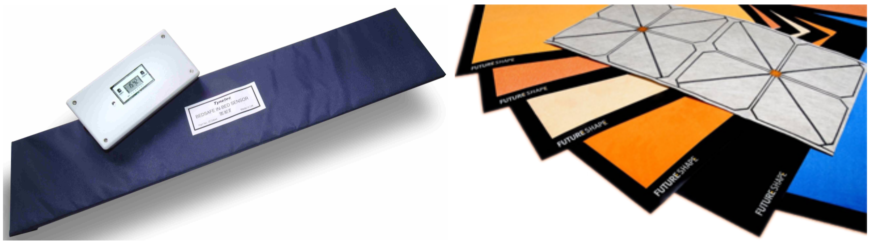
vibration pattern than normal activities. A total of 70 dummy falls and 53 object drops were performed with 100% detection rate of falling dummies and 0% detection rate of dropped objects.

[54] installed a wall mounted pyro electric infrared sensor array of 16x16 elements for thermal imaging and fall detection.

Commercial products

Common are occupancy sensors, e.g. Tynetec Altec Bed Occupancy Sensor [64], which can be placed in the bed or on chairs and start a timer when the person exits the place. If it does not get back within a specific time period, an alarm is triggered.

Other examples of commercial products are pressure and proximity sensing fall mattresses, e.g. Senstec fall mattress [18], which detects falls occurring on them. Further also movement detectors can be used to monitor the activity of people.



(a) Tynetec Altec Bed Occupancy Sensor. [64] (b) Senstec fall mattress. [18]

Figure 2.3: Commercial products associated with the ambience device approach

2.3 Camera based approach

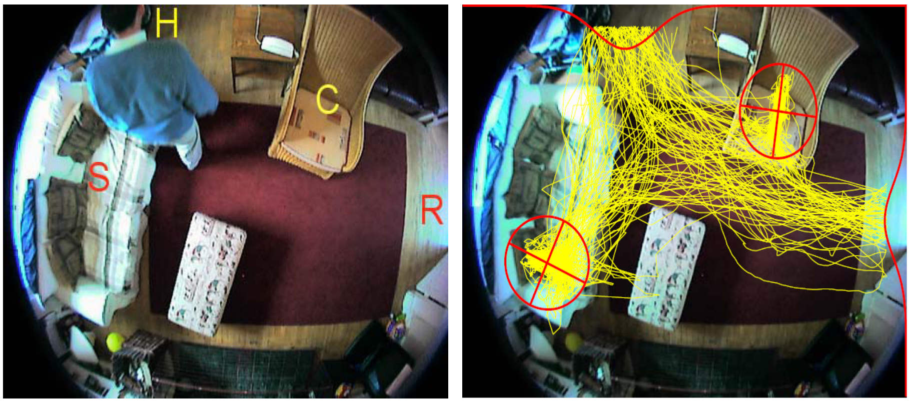
The camera based approach proposes the use of one or more cameras to monitor the person and extract features related to the person's motion, pose and activity.

Although during research within this work no existing commercial camera based fall detection system for elderly could be found, there are a variety of different approaches in this field. The majority uses conventional CCD cameras but also CMOS [23] cameras, Time-of-flight cameras [16] and infra-red vision sensors were proposed for fall detection.

In inactivity algorithms, the principle that a serious fall will end in an inactivity period on the floor [68] is used. Inactivity is detected by tracking the subject and computing the motion between the frames. If the activity decreases below a threshold, the pose and spatial context are checked to decide whether the situation fits to the normal activity pattern of the person or an unusual event, like a fall happened. The model of the person's activity should be learned on-line and therefore adapt to the individual subjects and rooms.

[41] mount a standard wide-angle camera on the ceiling to minimize occlusion of the person by furniture. The interpretive aim for the vision system is to monitor room occupancy to detect falls and perform analyses of activity patterns. The idea is to learn activity patterns of the person in a specific room to distinguish normal from abnormal activities like falls. Therefore an adaptive

background model with shadow detection is build and an overhead tracking is performed with a particle filter. The trajectories provided by the tracker are used to model the activity pattern of the person in the specific room. Here the aim is to automatically detect inactivity zones, where the person moves little, e.g. a sofa, and entry zones on the entrances of the room (see Figure 2.4(a)). In this way a model of spatial context is learned using maximum a posteriori estimation of Gaussian mixture models (see Figure 2.4(b)). When a person's speed drops to an extent that indicates inactivity, it is checked whether the inactivity is occurring in a known inactivity zone, otherwise a possible fall alarm is triggered.



(a) Zones of the room. S labels the sofa, C the couch, H and R are entry zones of the room. [41]

(b) Model of spatial context. [41]

Figure 2.4: Inactivity algorithm for fall detection. Inactivity (red ellipses) and entry zones (red probability density functions on the margins) in Figure 2.4(b) are learned automatically and represented as Gaussian mixture models. The summarized trajectories of the person's activities are marked yellow. [41]

In [31] low level resolution (64x64 pixels) depth images obtained from a time of flight camera are combined with gray level intensity images. Based on matching shape context from 2D visual images, a model defined by seven key points is fitted to the images to obtain the 2D orientation and position of the body. By estimating the position of the camera in the room, the 2D information is transformed into 3D. The orientation and activity of the person are combined with a simple context model of the room, which is built by dividing the floor into a discrete grid. For every cell of the grid it is investigated how long the person is located in it and so a histogram is associated with every cell, which relates the probability of inactivity over a time period.

In shape change analysis algorithms, the principle is used that during a fall the shape of a person changes from standing to lying.

To describe the shape of the body, features like, the bounding box height to width ratio, are extracted from video data and used for thresholding or training a model.

In [69] they use four cameras and for each of them a separate fall confidence is computed. The confidences are then merged into an overall decision with a voting algorithm. They combine features related to the body posture and motion with a statistical behavior model, which represents the likelihood that an activity occurs in a specific area. The motion is described by the relative number of new motion pixels in the current frame compared to the previous frame. For the posture the bounding box aspect ratio, orientation of the major axis and the axis ratio of the fitting ellipse are computed. Sets of empirically determined fuzzy thresholds in the form of trapezoidal functions are defined to relate these features to the three postures, standing, in between and lying (see Figure 2.5). The motion speed feature is combined with the estimated

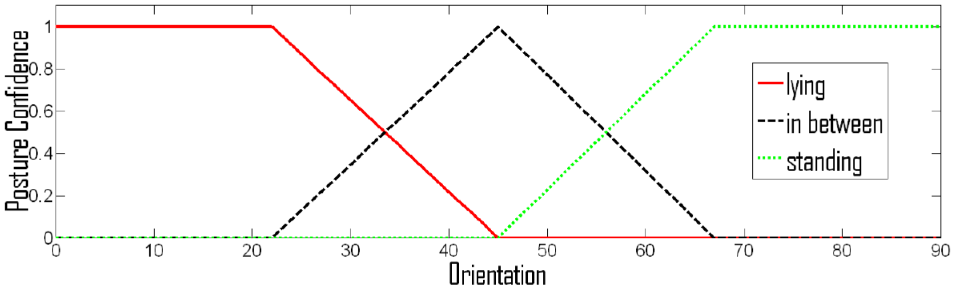


Figure 2.5: Trapezoidal functions to relate the orientation of the major axis to the three postures standing, in between and lying. [69]

posture confidences to model a fall as relatively high motion speed, followed by a period with a lying posture. A potential fall detected by these rules is verified with the statistical behavior model. The statistical behavior model (called accumulated hitmap) is a matrix which counts consecutive foreground pixels (see Figure 2.6). The values in the hitmap therefore specify the duration of the continuance of a person in a position in the scene. For classification the trained hitmap is used as a reference for normality in the scene, and compared with the current computed hitmap in order to detect unexpected situations. Figure 2.6 shows different camera views (top) and the corresponding trained hitmaps (bottom) of the laboratory environment. The hitmap counts consecutive foreground pixels in a way that they are increased as long as the corresponding pixel in the input image is classified as a foreground pixel. Higher values in the hitmaps correspond to positions where the person usually stays for a longer duration, e.g. the region near the two chairs in the left top image. As the values in the hitmap specify the duration of the person's presence in a position, the corresponding hitmap (bottom left) is marked with brighter i.e. higher values in the near the chairs. If the corresponding pixel in the input image is no foreground pixel anymore, the pixel values in the hitmap are decreased.

[63] use a stationary mounted camera and detect moving regions with a background estimation method presented in [13]. From the moving regions wavelet coefficients for the aspect ratio of the bounding box are calculated. Wavelet coefficients instead of aspect ratios have the advantages that they can reveal the aperiodic characteristic which is intrinsic in the falling case and thresholds in the wavelet domain are robust to variations of posture sizes and aspect ratios for different people. For classification Hidden Markov Models for walking and falling are created with wavelet coefficients of 20 consecutive frames by estimating the transition probabilities empirically. Additionally an analysis of audio track data is incorporated in the decision process using also wavelet coefficients and Hidden Markov Models.

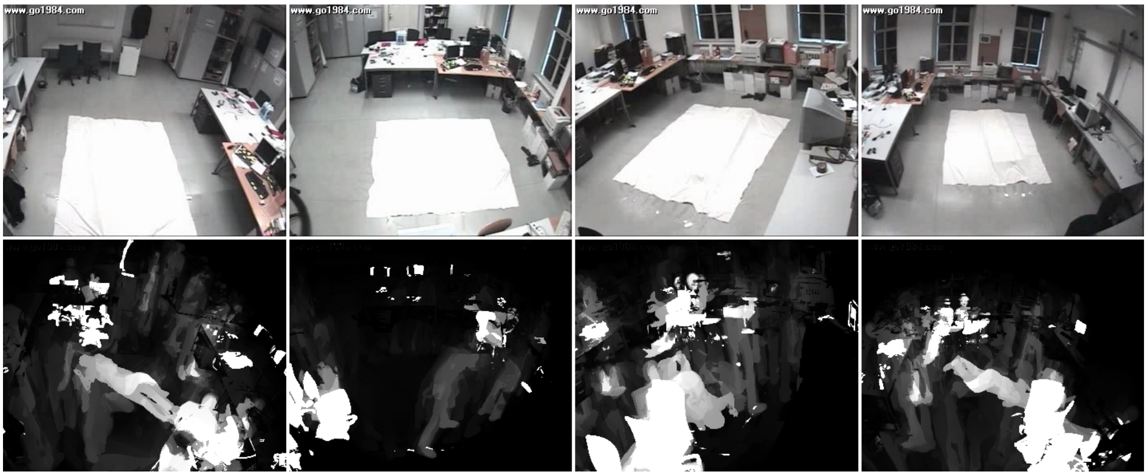


Figure 2.6: Four camera views (top) and the corresponding trained hitmaps (bottom). [69]

[62] propose a study of the relationship between angles in the 3D world and their projection onto the image plane. They determine the 3D world angles between the principal axis of the detected human blob and the vertical direction. The 3D angles obtained of consecutive frames are used as observation vector for the first layer of a two layer Hierarchical HMM for human activity. The first layer motion models the elementary behavior pattern and corresponds to movements with small temporal extent. It is composed of two states, one for an upright and one for a lying pose. The elementary motion pattern constitutes the states for the second layer, which correspond to global motion that lasts longer than the elementary motion pattern, and can be denoted as behavior (see Figure 2.7). For example, a FALL is supposed to be composed of a sequence of several “Is Walking“, followed by some “Is Falling“, and finally many “Is Lengthened“ elementary patterns. The two levels of abstraction enable to filter out possible false alarms that occur using a single level model. The evaluation was performed on 50 cases of fall and walk respectively. The results are a rate of 82 % correct detections for falls and 98 % for walking.

[15] use projection histograms to classify four main postures: standing, crawling, sitting and lying, frame by frame. They improve the frame by frame classification by exploiting the temporal coherence of the postures and therefore adopt a Hidden Markov Model which takes the visibility status of the person into account.

[12] use a method based on a combination of 2D skeleton features and human shape variation. Fast posture changes are detected by thresholding the distance between two sampling skeletons. Additionally the orientation and the axes ratio of the bounding ellipse are computed using zero and second order image moments. Fast posture changes are detected as potential falls and the final verification of a fall-down incident is done by checking if the person is inactive for a period of time after a possible fall.

[16] extract a 3D skeleton from the data of a wall-mounted Time-Of-Flight camera with the help of a graph based approach. To detect falls, they use thresholds for the distance of the human centroid from the ground floor and the orientation of the human spine (see Figure 2.8).

[22] extract 2D features related to the bounding ellipse, the Fourier coefficients of the pro-

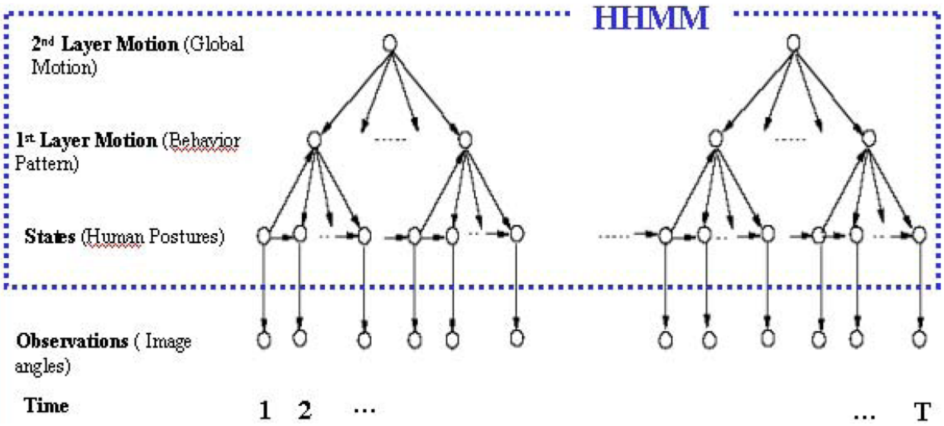
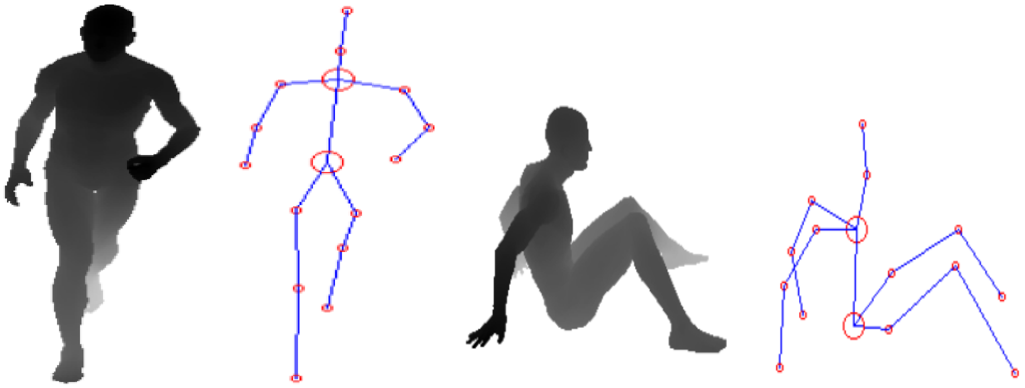

HHMM
2nd Layer Motion (Global Motion)
1st Layer Motion (Behavior Pattern)
States (Human Postures)
Observations (Image angles)
Time
1
2
...
...
T

Figure 2.7: Hierarchical Hidden Markov Model for human activity. [62]



(a) Standing posture (b) Standing posture (c) Sitting posture (d) Sitting posture

Figure 2.8: Two postures and the corresponding extracted skeletons. a) and c) are the depth maps, b) and d) the corresponding skeletons. The upper and lower principal nodes of the skeleton are marked with bigger red circles. The axis between these nodes is used to compute the orientation of the human spine. [16]

jection histograms and the motion of the head, from data of a single horizontal mounted camera. To detect changes, the head motion and the standard deviation of the ellipse features are computed for the duration of two seconds. Support Vector machines and in [20] also a Multilayer Perceptron are used to distinguish between 10 postures including walking, bending down, sitting down and falling.

[21] use adapted motion history images called integrated time motion images (see Figure 2.9), which take the time aspect into account. The basic idea is to construct an image that can be matched against stored representations of known movements. A motion history image is a kind of temporal template that is constructed by successively layering selected image regions over time using a simple update rule. To reduce the dimensionality of the motion history images

they are projected into a set of points in the eigenspace and finally neural networks are used for classification.

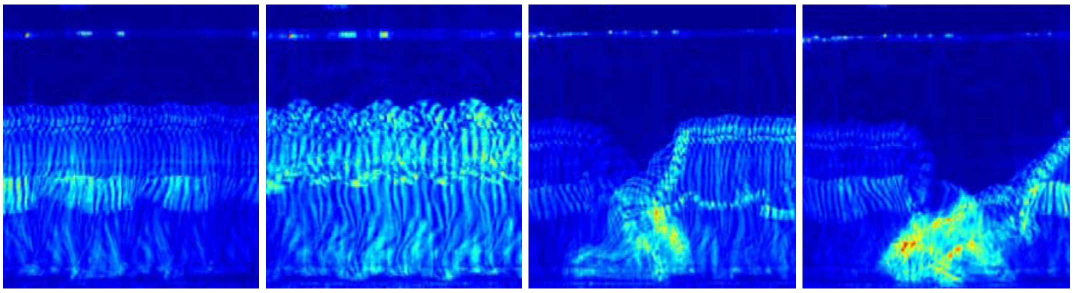

(a) Walking (b) Running (c) Forward fall (d) Backward fall

Figure 2.9: Integrated time motion images. A spatio temporal template stores motion history and occurrence time of motion with emphasizing the final action (brighter regions in the images). [21]

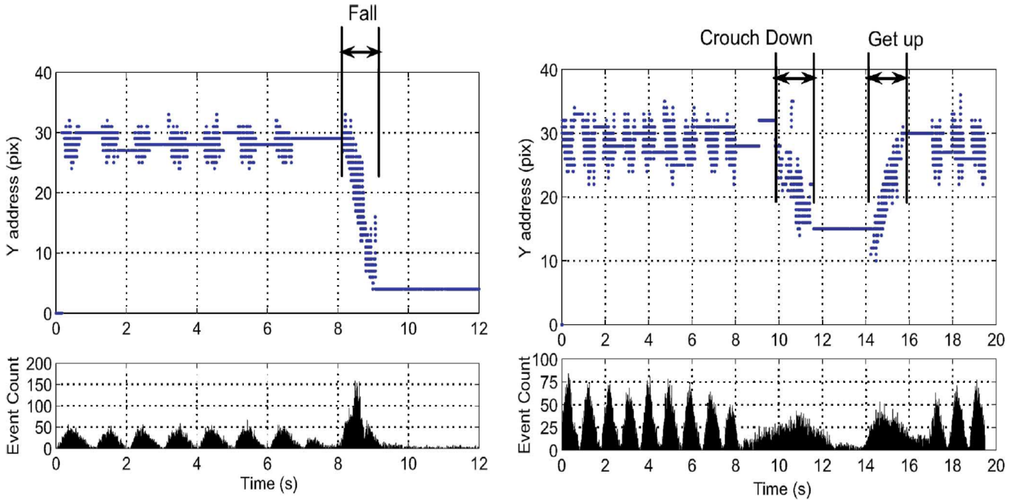
In [5] a multiple camera network is used for reconstructing the 3D shape of people. Falls are detected by analyzing the volume distribution of the detected human blob along the vertical axis. When the major part of this distribution is near the floor for a predefined duration, a potential fall is assumed.

[11] found a method to extract velocity estimates of the person's centroid from data of a differential infra-red sensor. A number of 84 fall and 26 non-fall sequences were recorded. The data points obtained by the velocity estimation method are labeled in a specific way according to whether they occur during a fall or not. This provides approximately 50.000 classified data points, of which 20% are extracted at random for training of a Multilayer Perceptron, leaving 80% for testing the network.

An 64x64 pixel asynchronous temporal contrast vision sensor for fall detection is proposed in [23]. They compute centroids as temporal averages of a series of events to estimate object motion in space. The spatial average is performed over a time window of 30 ms or a minimum of 10 AEs. For the implementation they used a First In First Out (FIFO) buffer, which stores events that occur in a fixed time period. Figures 2.10(a) and 2.10(b) show the course of the vertical centroid (the average of an amount of AEs in y- direction) during a fall and a crouch. For fall detection they use the velocity of the vertical centroid addresses. To provide a velocity measurement that is invariant to distance, they normalize the computed velocity by dividing the height of the subject in pixels. A potential fall is detected if the velocity exceeds a specific threshold. In this case a timer for verifying the alarm starts. For verification it is checked if the person stays near the ground or manages to get up.

In 3D head motion analysis the advantage, that the head of a person is usually visible in the image and has a large movement during a fall, is used.

In [50] the head is tracked with a particle filter to extract the 3D head trajectory from a single camera video sequence. The fall detection is based on thresholds for the vertical and horizontal velocity of the head.



(a) Fall (b) Crouch

Figure 2.10: Course of Vertical centroids and AE count during two different activities. In Figure 2.10(a) when the person falls, the centroid vertical address decreases by 25 pixels in 0.9 s. The event rate is approximately 5600 event/s. In Figure 2.10(b), when the person crouches down, the centroid vertical address decreases by 20 pixels in 2 s. The event rate is approximately 310 event/s. The fall causes faster decrease in the vertical address than the crouch-down. [23]

Comparison

Wearable devices provide compared to the other fall detection approaches the most accurate information about the motion of the person because they are mounted directly on the wearer. This is yet only possible if it can be assumed that the worn device keeps a fixed relation to the wearer's body.

Further [34] addressed the problem that one placement of a device cannot measure accurately the pose of the body in all cases. As example, if a person bends down, a tilt sensor on the trunk measures a horizontal pose while the person's lower part is in a vertical position. To get more accurate measurements, more than one device should be mounted on different places, which is not practicable for elderly in daily life.

Probably the biggest practical disadvantage is that wearable devices are intrusive and even if the devices are small, elderly have to be willing to wear them all the time. They have to care about not forgetting it, even if they feel well and secure. The general comment from doctors is that most of the patients have low will to wear the fall detector because normally they feel secure before a fall occurs [68].

The advantage of ambience device is that once it is mounted, it does not need any user interaction. Therefore it is less intrusive than wearable devices. On the other hand ambience devices can only detect events near the mounting place. To cover the whole home environment a high amount of sensing elements is needed, which is expensive and intrusive concerning the home environment of the person.

The camera based approach has the advantage that, in comparison to the ambience device approach, a relatively big space can be monitored with a single device, and remote visual verification is possible. It is also less intrusive than the wearable devices.

Using cameras yet, the issue of privacy protection has to be considered. Conventional Charge Coupled Device (CCD) cameras capture the whole irradiation of the scene in a specific frequency domain. Therefore it has to be ensured that the data is kept in confidence and is protected at every step of the systems processing chain. Here the infra-red and asynchronous contrast vision sensors have the advantage that less details are captured, and so the person cannot be recognized as well as in conventional images.

Also related to the privacy protection aspect, there is the psychological aspect that a person may feel observed by a pair of optical sensors reminding of eyes. Even if they are informed about the privacy protection and the working principles of the sensor, it may be hard especially for elderly, to keep that in mind the whole time, and be comfortable in the presence of camera based devices.

Summary

In this section, related work about fall detection approaches was depicted. Analyzing the results of the existing works, every approach has specific strengths and weaknesses. Most of the presented works are evaluated only on a small amount of situations in a laboratory environment and their application is very constrained.

As commercial products for fall detection, mostly wearable and ambience devices, are available. The manufacturers point out, that only specific kinds of falls can be detected, because the used sensors provide not enough features to distinguish the high variety of normal movements from the falls. To enable a sufficient ambience assistance and protection, a combination of different approaches should be considered.

CHAPTER 3

Methodology

This chapter describes the steps of the methodical process for the work within the thesis. The first section begins with a short introduction to the dynamic stereo vision sensor and the 3D data obtained. This data is a continuous stream of pixel positions with associated depth information. To get information about the actual pose of the person's body, the data is transformed into world coordinates with the help of the camera and set up parameters.

The basis for the design and evaluation of the fall detection method is a database of recorded movements. The database contains 205 scenarios acted by men and women including different fall scenarios and normal movements, like sitting down or walking. For the records, the laboratory environment is arranged with common furnishing including a sofa, desks, boxes and chairs. The scenarios and the laboratory environment are described in Section 3.2.

For fall detection, significant features describing the pose and movement of the body are proposed in Section 3.3. The features are extracted from the world coordinate representation of the person related data. This data is still present in form of a stream with subsequent events of world coordinates. Because of the asynchronous data generation of the sensor, the data is not produced in fixed time intervals but according to the occurrence time. To compute features at points in time, an amount of data points is aggregated according to a defined time interval and quantity.

In the next steps the features are processed with different filters to improve the quality. The DCT of the features is used to represent them in an effective way and simplify the classification task for the fall detection method. The filtering and transformation of the features are described in Section 3.4.

To partition the scenarios into samples, their continuous computed and processed features are divided into overlapping 3 seconds long sliding time windows. In this way a sample set of about 7500 time windows with according features is obtained. Due to the short duration of a fall, the majority of the samples contain normal movements while 113 of the whole amount are "fall samples". On the set of samples, the features are investigated with respect to their significance for fall detection. Therefore their alternation during fall and no fall time windows is compared and visualized in form of histograms.

Two machine learning approaches are investigated on the basis of the obtained sample set. The HMM and the MLP are trained and cross validated to establish a classifier which can classify unseen feature samples, whether they contain a fall or not. The influence of the features on the classification performance is analyzed by training and cross validating the models with different subsets of features.

3.1 Dynamic stereo vision sensor for fall detection

The dynamic stereo vision sensor ([8] and [7]) consists of two arrays of 304x240 pixels, built with 0.18 Complementary Metal Oxide Semiconductor (CMOS)- technology, where each pixel contains a change detector and a photo-measurement device. Figure 3.1 depicts the sensor. The pixels respond to relative light intensity changes and only pixels, which detect an illumination change, produce an output. By sending their position in the pixel matrix and their polarity (ON or OFF according to illumination increase or decrease) asynchronously, the so called AEs are generated. In further processing steps, the AEs of the two sensing elements are multiplexed to one AE- stream, supplemented with time-stamps and buffered. Further main components of the board are the Field Programmable Gate Array (FPGA) where the stereo algorithm is integrated and a digital signal processor.

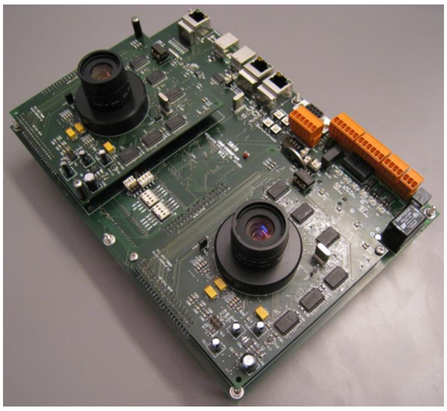


Figure 3.1: Dynamic stereo vision sensor.

Stereo algorithm

An area based stereo matching algorithm is implemented on the FPGA. The algorithm first divides the continuous AE stream into timeslots of a specified duration, and accumulates the AEs according to their polarity. Then for each spiking pixel in the left vision sensor, the corresponding spiking pixel in the right vision sensor is searched. Due to the epipolar geometry, the search is constrained to the epipolar line, which after specific transformations can be assumed as horizontal line. Therefore the matching candidates are searched in the right sensor only on the corresponding horizontal line with the same index. The search is performed by investigating patches around the pixels, and computing the normalized sum of absolute differences as similarity measure. The deviation between the two pixels with the most similar patches is inserted in the disparity map on the position of the left pixel. The disparity for each pixel can be directly transformed into the normal distance i.e. depth, with following formula [57]:

$$D = \frac{f * B}{dx * p} \tag{3.1}$$

where D is the normal distance, f is the focal length of the lenses, B is the base distance between the two sensor arrays, dx is the disparity in pixels and p is the pixel pitch. The lenses used in this case have a focal length of 0.48 cm, the pixel pitch is 0.003 cm and the baseline between the two sensors is 16.5 cm. Figure 3.2 shows the AE data collected by the left and right sensor for a scenario of two people walking over a time span of 40 ms. The AE stream is rendered to an image like representation. Figure 3.3 depicts the corresponding depth map obtained by the stereo matching algorithm. The AEs are color coded in a way that nearer objects are colored yellow to red.

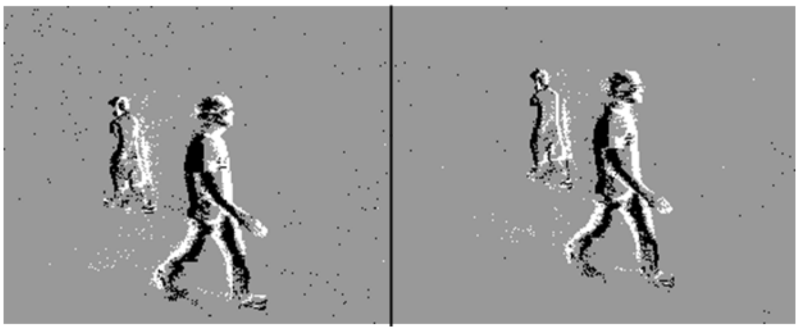


Figure 3.2: AEs from the left and right sensors in an image-like representation. The pixels color codes the polarity of the events. White pixels denote ON-events, with an illumination increase, while black pixels stand for OFF-events caused by illumination decrease.

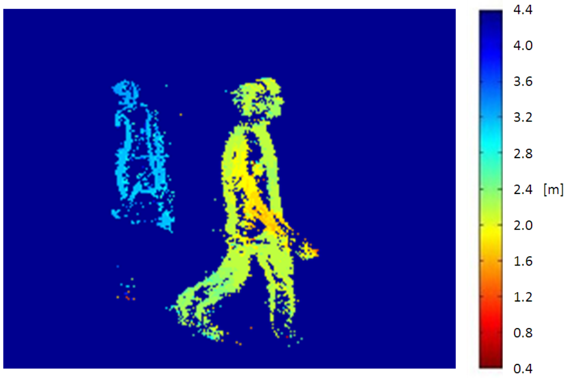


Figure 3.3: Depth map of people walking in front of the sensor. The depth is color coded; red objects are nearer to the sensor than blue ones.

Transformation to world coordinates

To compute significant features describing the person's pose and motion, the depth map obtained has to be transformed to a representation of the data in the 3D world coordinate system. Therefore the extrinsic and intrinsic camera parameters are used. Figure 3.4 depicts the relationship between the image coordinates in pixels and the camera coordinates in cm. The camera coordinates are obtained by following equations:

$$xc = \frac{(x - midx) * p * D}{f} \tag{3.2}$$

$$yc = \frac{(240 - y - midy) * p * D}{f} \tag{3.3}$$

$$zc = D \tag{3.4}$$

where xc, yc, zc are the camera coordinates in cm, x and y are the image coordinates in pixels, p is the pixel pitch, f is the focal length, $midx$ is half of the image resolution in x direction (304/2), $midy$ is half of the image resolution in y direction (240/2) and D is the depth obtained in Equation 3.1.

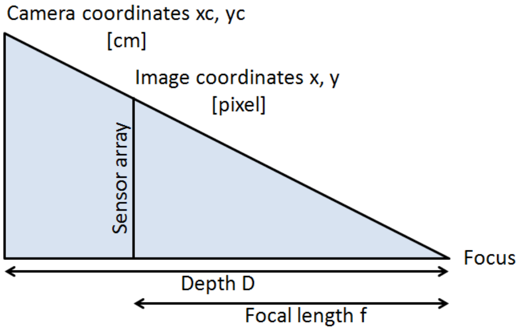


Figure 3.4: Relation between image and camera coordinates.

The camera coordinates are then transformed by rotation and translation into world coordinates with the help of the transformation matrix R which involves the extrinsic parameters of the stereo vision sensor:

$$R = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & cos(\alpha) & -sin(\alpha) & transy \\ 0 & sin(\alpha) & cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.5}$$

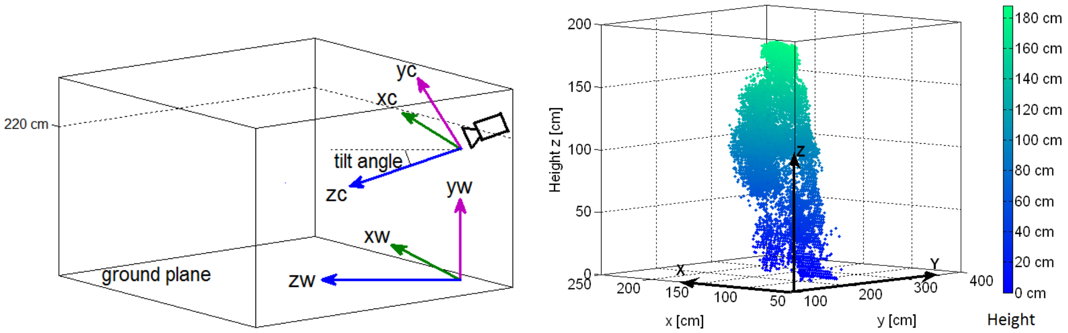
The transformation effects a rotation around the x axis with a rotation angle of α, and a translation of $transy$ in y direction. The world coordinates can be obtained by following multiplication of the camera coordinates with the transformation matrix R:

$$\begin{pmatrix} xw \\ yw \\ zw \\ 1 \end{pmatrix} = R * \begin{pmatrix} xc \\ yc \\ zc \\ 1 \end{pmatrix} \tag{3.6}$$

where xw, yw and zw are the world coordinates, R is the transformation matrix described in expression 3.5, xc, yc and zc are the camera coordinates. Figure 3.5(a) shows the configuration of the vision sensor with the camera and world coordinate systems. The person related data is now represented as point cloud in a 3D world coordinate system, where the yw coordinate denotes the height of the person and zw stands for the distance to the origin of the world coordinate system. The sensor is mounted in a height of $transy$= 220 cm with a tilt angle of $alpha$ = 30. The world coordinate system is defined by shifting and rotating the camera coordinate system in a way that the resulting coordinate system lies below the camera coordinate system, and the xz-plane accords to the ground plane of the room.

For a better understanding and conformity with the common denotation as z for the height, the labels of the z and y coordinates are switched, so that in this thesis z denotes the height and y the depth. Figure 3.5(b) shows an example for the person related data points in the resulting world coordinate system. The z coordinate denotes the height, the yx-plane accords to the ground plane of the room. The height is color coded, so that brighter green means higher z coordinates.

On the basis of the data obtained in the 3D space, the features for fall detection described in Section 3.3 are computed.



(a) Configuration of the vision sensor.

(b) Person related data points.

Figure 3.5: Configuration of the vision sensor (left) and person related data points in the world coordinate system where z denotes the height (right).

3.2 Data

To train and evaluate the models, a database of acted scenarios including 113 falls and 92 normal activities is recorded with the set up described in Section 3.1. The scenarios are performed in a laboratory environment by 7 men and 3 women. Each of the 10 people performs around 9 normal activities and 11 falls leading to a total amount of 205 recorded scenarios. A couch, desks, chairs and bins on the floor are included to emulate a possible home environment of an elderly. Figure 3.6 shows a floor plan of the room.

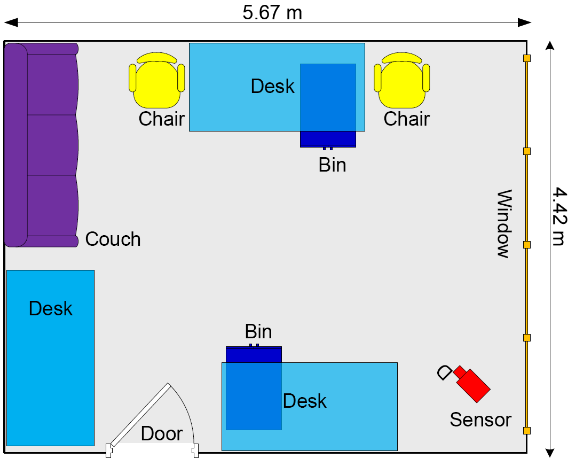


Figure 3.6: Laboratory environment for the recording of the acted scenarios.

The falls are performed with a mattress to ensure the security of the actors. Following scenarios are performed by every actor:

No fall scenarios

- Sitting on a chair
- Standing up from a sitting pose in a chair/ couch
- Standing up from a lying pose on a couch
- Walking around slowly
- Standing near the desk or chair
- Bending down to lift a small object from the floor
- Bending down to take an object from a bin on the floor
- Sitting down on a chair
- Sitting down on a couch
- Lying down on a couch
- Stumbling

Fall scenarios

- Fall from standing lateral/ frontal/ backwards
- Fall during standing up where the person tries to hold but fails
- Fall from walking because of collapse
- Fall during trying to lift an object from the floor
- Fall backwards because of losing balance
- Fall from walking because of stumbling
- Fall while trying to sit down and miss the chair/ couch
- Fall from lying on the couch
- Fall from sitting on a chair

3.3 Features

The AE data of each scenario is streamed continuously and saved to a file. Due to the asynchronous data generation, the AEs are not produced in fixed time intervals. The amount of AEs produced within a specific time span is called activity. Therefore the higher the activity, the more AEs are available in a specific time span. To compute features, AEs have to be aggregated. Therefore an amount of 800 events and a minimal time span of 0.06 sec are defined empirically. The minimal time span constrains the time extend of the aggregated AEs during high activity periods, where 800 events may be collected in a shorter time than 0.06 sec. On the other hand in time periods with low activity, a minimal time span of 0.06 sec may not provide enough data to compute significant features, and therefore a minimal amount of AEs constrains the data collection in this case. The combination of the amount of AEs and the minimal time span gave empirically the best results and enables to beware the asynchronous characteristic of the data. Based on this aggregation of data, continuous features that describe the motion and pose of the body can be extracted directly from the person's world coordinate point cloud. An amount of eight features is computed in that way and processed with different filters to improve the quality and reduce noise.

In this section the features are introduced and demonstrated on a fall from walking scenario including standing up from the right chair in Figure 3.6, walking a few steps in the direction of the camera and then falling on the floor acting a collapse. Figure 3.7 visualizes 6 snap-shots of the depth map during the fall. It shows the person walking and then collapsing and falling on the floor. In the last snap shot (bottom right), the person is moving little and therefore less AEs are visible in the depth map. The depth is color coded so those nearer objects are marked with brighter green. Because of the upper mounting, the person's head is nearer to the sensor and therefore brighter, than the legs (see upper, left image).

Figure 3.8 shows a motion history (similar to the concept of integrated time motion images in Figure 2.9) of the fall scenario in 3D world coordinates. The time is color coded so that the last generated data points are marked green. For clarity only the time span, where the fall occurs (between second 7 and 11 of the recorded scenario) is depicted. The z coordinates accord to the height of the person and the y direction encodes the distance to the sensor. The person first nears the sensor, and then falls in the direction away from the sensor. The dark blue data points (second 7 to 8 on the color bar) show the walking person in an upright pose. Between second 8 and 10, the person falls and ends up in a lying pose on the floor (green data points which accord to second 10 to 11).

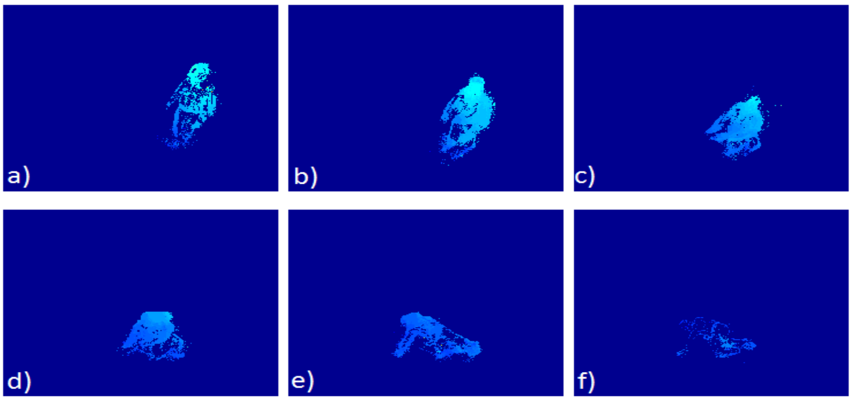


Figure 3.7: Snap-shots of the depth map during the fall. The depth is color coded in a way that brighter green colors stand for nearer objects.

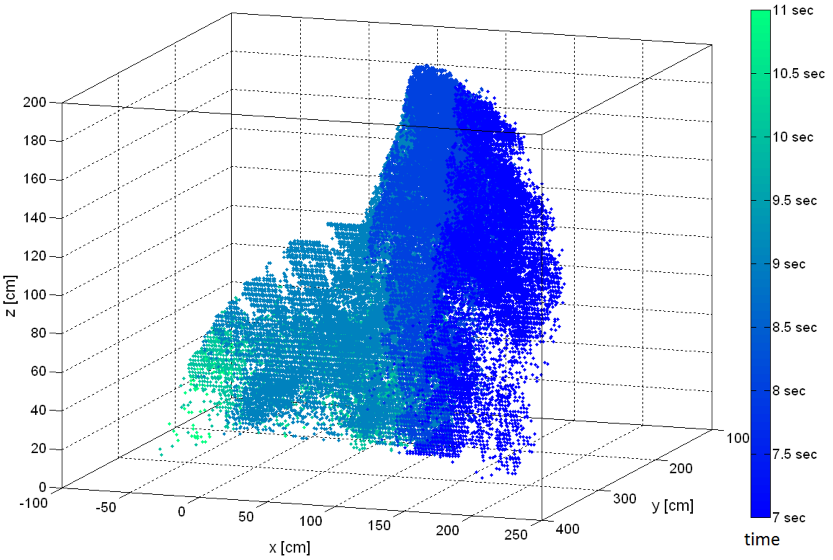


Figure 3.8: 3D world coordinate representation of a fall scenario. The time is color coded to illustrate the course of the movement.

Highest point

The determination and tracking of one characteristic point of the objects data, reduces the dimensionality, but still allows drawing conclusions about the pose at a specific time. Using the highest point as feature has advantages, even if the computation is left simple and no explicit head detection is included. The highest point has less occlusion, and its motion has more correlation with fall than other parts of the body. To avoid outliers it is computed as the height under which 97 % of the data points lie.

Median of the height

As additional characteristic point, the median of the height can be computed. The median is described as the numerical value, separating the higher half of a sample from the lower half, and is more robust against outliers than the mean. Figure 3.9 shows the highest point and the median of the fall scenario. At the beginning the person is sitting on a chair with little movement. The highest point and the median stay constant at about 120 cm and 40 cm. After approximately 4 seconds the person stands up and walks in the direction of the sensor. According to this, the characteristic points are rising. At second 8 the person acts a collapse and falls. Note that in this case the highest point declines below the height it was in the sitting position (second 1 to 4), while the median drops to nearly the same height. After the fall, the person remains lying on the floor.

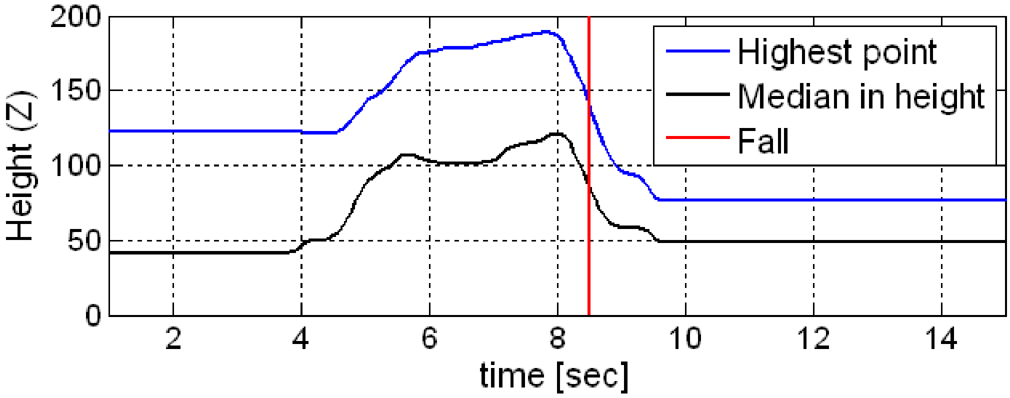


Figure 3.9: Highest Point and Median in height of a scenario including a fall.

Vertical velocity of the highest point

In addition to the course of the highest point itself, the vertical velocity of this characteristic point provides relevant information for fall detection. [67] investigated the velocity characteristics of falls with respect to distinguishing them from normal activities. They placed three markers on the posterior side of the trunk and measured the marker movements with 3 cameras. Normal activities and different fall scenarios were recorded. Concerning the vertical and the velocity in the horizontal plane, two characteristics were found. Firstly, the magnitude of both horizontal and vertical velocities increases during the falling phase, reaching up to 2-3 times that of normal velocities. Secondly, the increase of the two velocities occurs simultaneously, which is not the case during normal movements.

For the fall detection in this work, no markers are used and therefore it is not possible to extract exact velocities of specific body parts. The horizontal velocity of the center of gravity, and the timing of the increase of the velocities, were investigated empirically, but did not bring significant information for fall detection. Further by intuition, a fall does not categorical induce an increase in the horizontal velocity, e.g. collapses or attacks where the person slumps down with little lateral movement.

The vertical velocity of the highest point is included as feature because it empirically approved to provide significant information about the downwards movement during falls. The velocity is computed by differentiation of the highest point with respect to the time. Therefore it is positive during upward movements and negative while moving down towards the ground.

Figure 3.10 shows the course of the vertical velocity of the highest point during the fall scenario described above. In the first 4 seconds the person is sitting on the chair, and the velocity is about zero because the person is nearly not moving. Between second 4 and 6, the person stands up which is recognizable in the increasing positive velocity. During walking (second 6 to 8) the velocity fluctuates near zero. Finally it reaches a peak of -140 cm/sec during the fall. After the fall it remains zero because the person does not stand up any more.

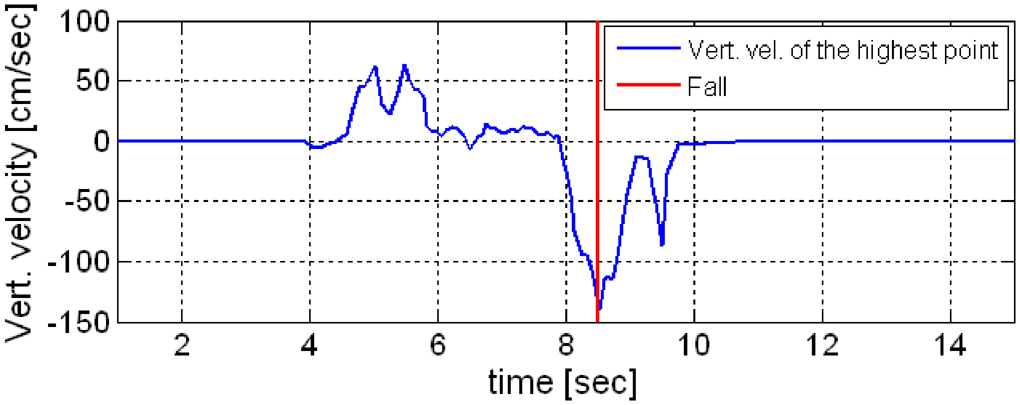


Figure 3.10: Vertical velocity of the highest point during the fall scenario.

Vertical volume distribution ratio

The Vertical Volume Distribution Ratio (VVDR) describes the distribution of the data points with respect to the height and is therefore also a measure for the pose of the body. [5] reconstructed a human blob with the help of multiple cameras and proposed the vertical volume distribution ratio as indicator for fall detection. It is computed as the amount of data points lying below a defined height threshold, related to the whole amount of data points, according to Equation 3.7. In this work an empirically determined threshold of 30 cm is used.

$$vvdr = \frac{n}{N} \tag{3.7}$$

where N is the whole amount of data points, n is the amount of data points lying below 30 cm measured from the lowest point, and $vvdr$ is the vertical volume distribution ratio. Figure 3.11 shows the course of the ratio during the fall scenario as before. During sitting (second 1 to 4) with little movement, the ratio remains nearly constant at about 0.3. While standing up at second 4 it decreases to almost zero. During the fall around second 8.5 the ratio increases and remains at 0.7 while the person is lying on the ground. This means that at the end of the fall 70 % of the data points are lying below 30 cm. Note that the VVDR does not increase stringently up to 1 because of the variability of the body build. Lying on the floor a corpulent body may over-top the 30 cm. On the other hand, if the threshold is too high, the ratio will increase too much during normal movements like bending down or sitting. Further in this case the person is lying on a mattress and supporting with its arms after the fall. Therefore not all data points lie below 30 cm and so the ratio does not increase to 1. Another remark is, that here the ratio alternates during standing up, walking and even afterwards during the fall more than the before mentioned features. The feature is less robust because it is not only affected by the pose, but also by the distribution of the motion with respect to the body. If the lower body parts are moved more than the upper ones, the Vertical Volume Distribution Ratio increases, even if the person is in a standing or sitting position.

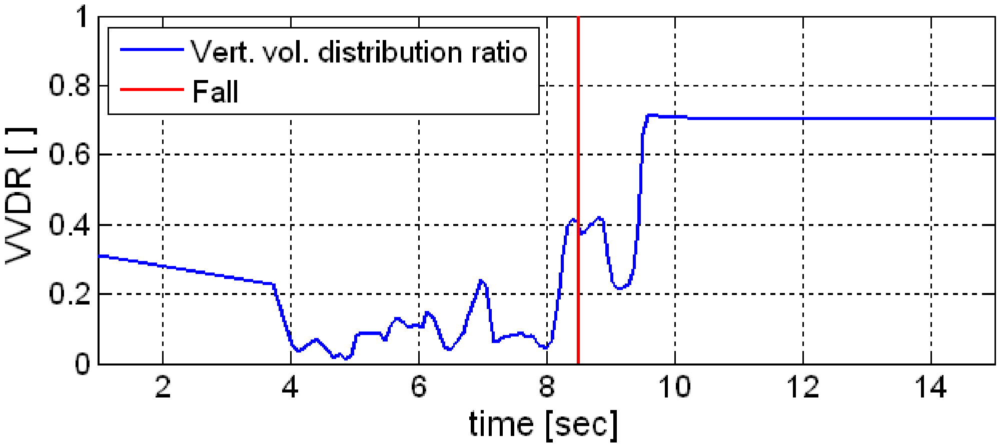


Figure 3.11: Vertical volume distribution ratio for the fall scenario.

Activity

The activity or AE rate describes the average amount of AEs produced within a specified time span. It is computed as the amount of AEs per second. According to the inverse-square law, the activity decreases with increasing distance between the object and the sensor. Therefore it is corrected by multiplying it with the square of the distance. A high activity can be caused by more or bigger moving objects, higher velocities, shadows and fast changes of the illumination conditions. Further also the texture and the contrast to the background of the person's clothes influence the activity. Figure 3.12 shows the course of the activity during the fall scenario as before. While sitting, between second 1 and 4, the person moves little and the activity is therefore low. During standing up and walking, the activity increases up to 190 kAE/sec, and while falling it reaches a peak of 270 kAE/sec. After second 11, the person is not moving; therefore the activity is about zero.

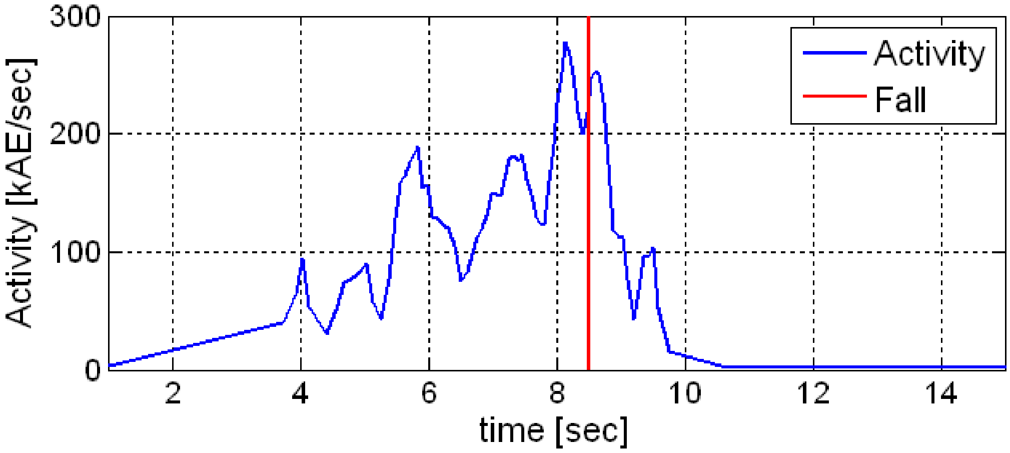


Figure 3.12: Activity of the fall scenario in kAE/sec.

Orientation of the main axis

Due to the elongate shape of the human body, it is possible to gain information about the orientation of the body by extracting the main axis. In this work the main axis is defined as the axis with the highest variance, and computed by 3D PCA of the point cloud. Therefore first the covariance matrix of the point cloud is computed. In the case of three dimensions, it is a 3x3 matrix and contains information about the variances and covariances of the data in each dimension. The matrix is composed as follows in Equation 3.8 [9].

$$C = \begin{pmatrix} cov(x,x) & cov(x,y) & cov(x,z) \\ cov(y,x) & cov(y,y) & cov(y,z) \\ cov(z,x) & cov(z,y) & cov(z,z) \end{pmatrix} \tag{3.8}$$

where x, y, z are the coordinates of all data points and $cov(dim1, dim2)$ is the covariance of the two dimensions $dim1$ and $dim2$ (here x, y or z). The covariances describe the correlation between the three dimensions of the data, that means how much an alternation in one coordinate affects the other two dimensions. The diagonal of the matrix contains the covariances of the dimensions with themselves, which is simple the variance or scatter in every dimension x, y and z.

The covariance is computed by following Equation 3.9 [9]:

$$cov(x, y) = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x})(y_i - \overline{y}) \tag{3.9}$$

where x and y are the two dimensions, x_i and y_i their corresponding elements, n is the number of elements in each dimension and $\overline{x}$, $\overline{y}$ are the means of the two dimensions.

As next step the orientations of the principal components i.e. the axes with the greatest variances when projecting the data on them, are found by computing the eigenvectors of the covariance matrix. The eigenvectors of the covariance matrix are defined as the vectors that, after being multiplied by the matrix, change only in magnitude, not in direction. For the three dimensional case, three eigenvectors and their corresponding eigenvalues exist. The eigenvalue is the factor by which the eigenvector changes when multiplied by the matrix. The eigenvector with the greatest eigenvalue defines the orientation of the principal component i.e. the axis with the greatest variance which is here defined as the main axis of the body. Finally the angle between the main axis and the ground plane is computed as measure for the inclination of the body. Figure 3.13 shows the main axis of the body and the angle between the axis and its projection onto the ground plane. The figure shows an amount of data points at second 7 of the scenario as before, where the person is walking away from the chair. In opposite to Figure 3.8 here the height is color coded so that green colors mean higher z coordinates. The angle between the main axis and the ground plane is about 85 at this snap shot.

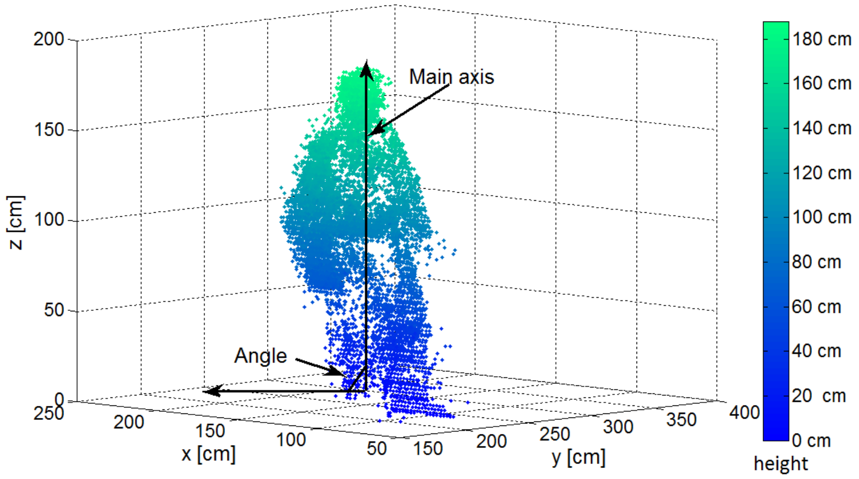


Figure 3.13: Orientation of the main axis of a walking person.

Figure 3.14 shows the course of the angle during the fall scenario as before. In the first four seconds the person is sitting on the chair and the angle conducts about 20. During sitting it depends which part of the body is moved and how upright the person is sitting. Therefore a higher angle of the main axis during sitting is just as possible. While standing up (second 4 to 6) the angle increases to about 80 and alternates around this magnitude while walking. During the fall the tilt of the body increases, and so the angle spanned between the main axis and the ground plane decreases to about 15.

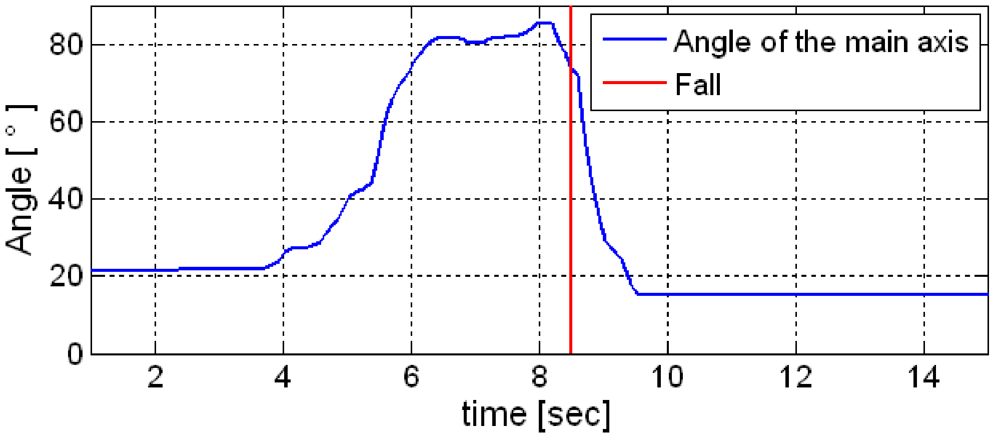


Figure 3.14: Orientation of the main axis during the fall scenario. The orientation of the main axis is relevant concerning the tilt of the body during fall. Here it decreases from about 85 to 15 during the fall.

Ratio of the occupied ground area to height

The ratio of the occupied ground area to height describes the pose of the body with respect to its distension in the ground plane and height. The occupied ground area is approximated by the spread of the data points in x- and y-direction. The ratio is computed with following Equation 3.10 for a specific amount of data points:

$$ah = \frac{std(y) + std(x)}{hp - lp} \tag{3.10}$$

where $std(x)$ and $std(y)$ are the standard deviations of all x and y coordinates, hp is the highest point, lp is the lowest point of the body, and ah is the ratio of the occupied ground area to height. The occupied ground area can be imagined as projection of all data points onto the ground plane, which is in this case, stretched by the x and y axes. If the person is in a lying pose, the variation of the projected data points in the ground plane is higher than in a standing pose. To distinguish bending down or sitting, where the occupied ground area also increases, from lying on the floor, it is combined with the height.

Figure 3.15 depicts the occupied ground area and the height of a standing person. The height is color coded so that brighter green means higher z coordinates. The figure constitutes a snap shot of the fall scenario as before at second 7, where the person is walking in direction of the camera. The ratio is about 0.2 at this moment, which means that the occupied ground area is small in relation to the height.

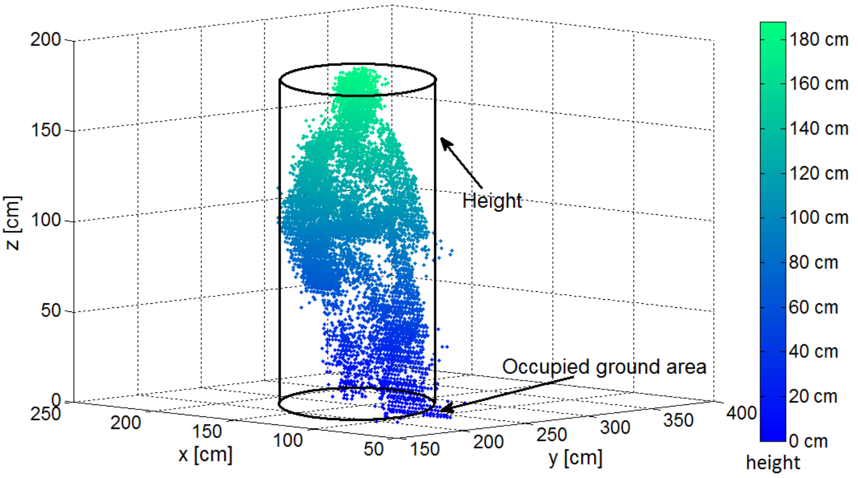


Figure 3.15: Ratio of the occupied ground area/ height of a walking person.

Figure 3.16 shows the course of the feature during the fall scenario. In the first four seconds the person sits on the chair and the ratio remains constant below 0.5. While standing up, the ratio first increases a little, which is not to be expected. According to the above explanation, while moving to a standing pose the ratio should decrease. The increase occurs because while sitting, the person moves only little with the trunk and head causing a small spread of the data in the ground plane. While standing up, it first stretches the legs and moves with the whole body. Therefore the occupied ground area increases more than the height at the beginning of the standing up movement. Then shortly before second 6, the ratio decreases according to the expectation, and stays below 0.2 while walking. During the fall, the ratio increases to 0.9 and stays constant because after the fall, the person remains in a lying pose on the floor. Note that the ratio is higher during the lying pose on the floor, than during the sitting position in the chair.

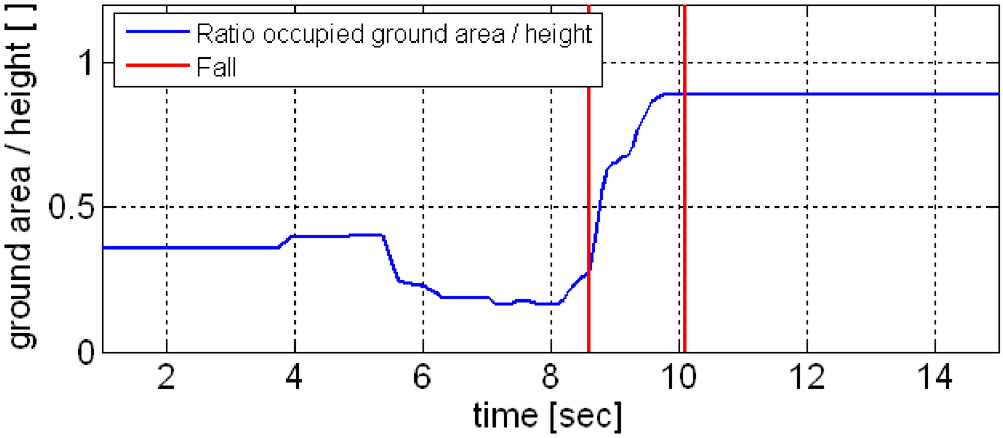


Figure 3.16: Ratio of the occupied ground area/ height during the fall scenario.

Standard deviation in height

As additional feature the standard deviation of the z coordinates is used to measure the statistical spread of the height. If the person is in an upright pose, the spread should be higher than in a lying pose. It is computed as follows in Equation 3.11:

$$std(z) = \sqrt[2]{\left(\frac{1}{n-1}\sum_{i=1}^{n}(z_i - \overline{z})^2\right)} \tag{3.11}$$

where $std(z)$ is the standard deviation of all z coordinates, n is the number of data points, z_i are the heights of the corresponding data points and $\overline{z}$ is the mean of the z coordinates.

Figure 3.17 shows the course of the standard deviation for the fall scenario. The feature remains constant during sitting on the chair, then before standing up (second 4 to 5) it decreases because the person is moving mainly with the upper part of the body. During standing up the height variation increases and reaches a peak of 45 cm while walking. Because of the leg movements it alternates during walking. While falling it decreases to about 17 cm.

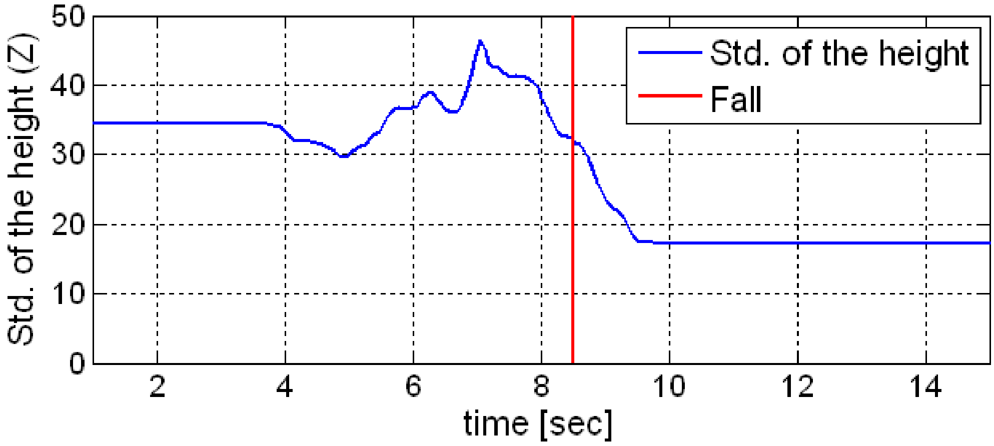


Figure 3.17: Standard deviation of the height during the fall scenario.

The combination of several features is necessary, because none of the features introduced is distinctive enough to distinguish falls from normal movements by its own. Due to the high variability of the people's movements in everyday life, there is a high amount of possible scenarios, where a single feature takes a similar course than during falls. For example, fast downward movements, like sitting or lying down, can cause high negative vertical velocities, which also occur during falls. In this case combining the velocity feature with others like the ground area to height ratio, may enable to distinguish these movements from falls.

On the other hand using too many features with poor significance, can lead to an increase of the dimensionality without the appropriate profit. Increasing the dimensionality of the data without an enlargement of the amount of data samples for the training, leads to a scarcity of the data in the high dimensional feature space. In this case too less data is available to train the model appropriate.

3.4 Feature processing

To minimize noise and smooth the features computed, filters are used at different processing steps.

Prior to the transformation into world coordinates, a noise reduction filter is applied on the AE stream. The filter investigates the neighborhood and the time intervals of the AEs to detect events caused by noise.

After the transformation into world coordinates, a simple mean based outlier removal is applied to segment the person related data points.

While feature computation, the time span and amount of accumulated data affects the smoothness of the feature. If the time span and the amount of data are specified too high, the computed features are blurred and single movements are not clearly recognizable. On the other hand, if too less data is taken into account for the computation, the features may be unstable. As mentioned in Section 3.3, for feature extraction a minimal time span of 0.06 seconds and a minimal amount of 800 data points are aggregated.

Due to the asynchronous data generation of the sensor, less data is produced during low activity periods. Therefore the quality of the computed features is decreasing. A Kalman filter, with the activity as input for noise modeling, is included in the computation, to improve the feature reliability during periods with little movement.

Finally a median filter is applied on the features obtained to smooth them additionally and remove possible remaining outliers.

To present the features in an efficient way to the learning models, the one dimensional DCT is applied on each feature. To show the improvement by the DCT, the learning models are trained on the one hand with the interpolated feature values, and otherwise with their DCT coefficients as input. The classification performances of the different models are compared in Section 4.

Noise reduction

The noise reduction filter is based on the event space filter algorithm described in [59]. It works on the AE stream, taking into account the position in pixel coordinates and the occurrence time of each AE. The direct neighborhood of each AE is investigated with respect to the time. If no new AE occurs there within a specified time span of 0.2 seconds, the AE is considered as noise and deleted.

Mean filter

For the final application, different moving objects like pets or curtains have to be considered, therefore a segmentation and tracking algorithm is necessary. In this thesis the main attention is concentrated on the fall detection method assuming a successful preceding segmentation of the person related data points. Therefore other moving objects are not included in the laboratory environment and the scenarios captured. To remove possible small impacts caused by e.g. a moving piece of paper or reflections, a mean filtering is applied. In every axis (x, y, z) the mean and the standard deviation of the aggregated data points are computed. Under the assumption that the data generated by a single person scenario is nearly normally distributed, 95% of the data lies within 2 times the standard deviation from the mean [44]. Therefore data points lying beyond the thresholds determined by two times the standard deviation are assumed as outliers and removed.

Kalman filter

Based on the remaining person related data points, the features are computed and subsequently a one dimensional Kalman filter is applied to each computed value. During low activity periods the computed feature value may be inaccurate because less data is produced. With the help of the Kalman filter inaccurate measured values can be adjusted by estimations, based on the previous measurements and the approximated noise. The inaccuracy of the measurements is modeled by a Gaussian distributed noise described by its variance. Every computed feature value is corrected by a correction factor K according to the following Equation [48]:

$$x_{n+1} = x_n + K_{n+1} * (m_{n+1} - x_n) \tag{3.12}$$

where x_{n+1} is the corrected feature value of the actual measurement (i.e. feature computation), x_n is the corrected feature value of the previous measurement, m_{n+1} is the actual measurement (i.e. the computed feature value) and K_{n+1} is the actual correction factor. K contains information about the noise and is computed by following Equation [48]:

$$K_{n+1} = \frac{(1 - K_n) * Var(noise)_n}{(1 - K_n) * Var(noise)_n + Var(noise)_{n+1}} \tag{3.13}$$

where K_{n+1} is the actual correction factor, K_n is the previous correction factor, $Var(noise)_n$ is the noise variance of the previous measurement and $Var(noise)_{n+1}$ is the noise variance of the actual measurement. $Var(noise)_{n+1}$ is the input for the Kalman filter and is approximated inversely proportional to the activity. That intends, that during low activity the noise rises and a stronger correction by the Kalman filter is needed because the computed features are less reliable.

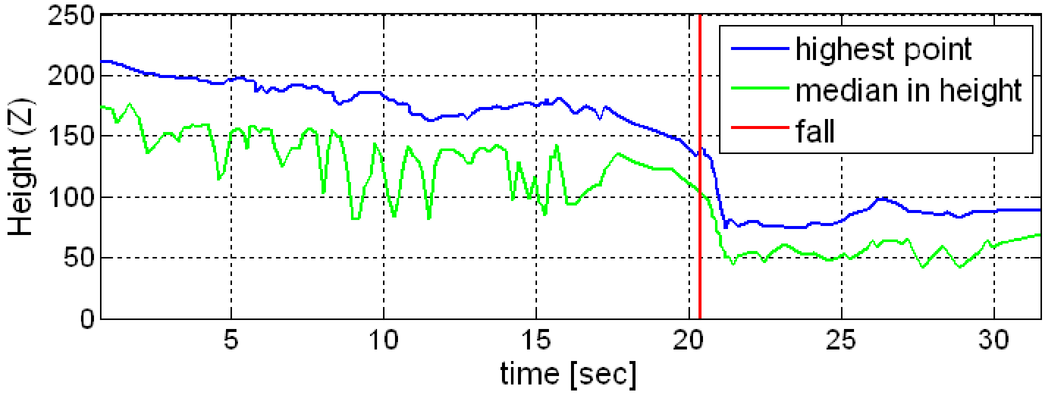
$$Var(noise) \approx \frac{1}{activity} \tag{3.14}$$

If the activity is high, the variance of the noise ($Var(noise)_{n+1}$ in equation 3.13) drops to a minimum and K_{n+1} nears 1. If K_{n+1} is 1 in equation 3.12, the corrected feature value of the previous measurement has no influence and x_{n+1} is equated with m_{n+1} without any correction. In contrast, if the activity declines, the actual measurement is weighted less and corrected by the combination of the factor K with the previous corrected measurement x_n.

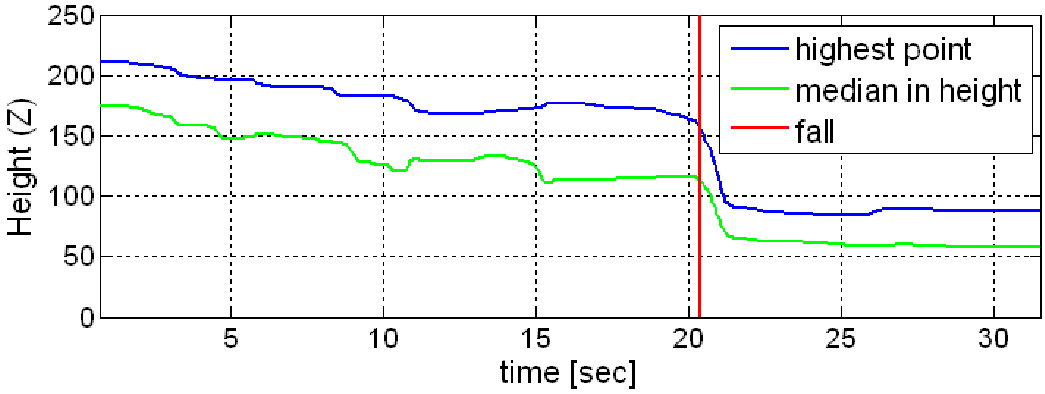
Median filter

For an additional smoothing and noise reduction a median filter is applied on each feature. The one dimensional median filter replaces each value of the feature vector by the median of an amount of preceding and subsequent values.

Figure 3.18 demonstrates the impact of the feature processing using the Kalman and median filter. It shows two features, the median in height and the highest point of a fall from walking scenario. The upper Figure 3.18(a) shows, how the features behave before filtering. The person is walking in the laboratory room until second 15, then stands still and falls around second 21. The actor is walking in a slowly and irregularly way trying to imitate a weak elder person. Therefore the features alternate during walking. By processing with the mentioned filters, the quality and stability of the features can be improved. The lower Figure 3.18(b) shows the result after applying the Kalman and median filter. Comparing the filtered to the unfiltered features especially during the walking period before, and the inactivity period after the fall, demonstrates the improvement.



(a) Features before applying the Kalman and median filter.



(b) Features after applying the Kalman and median filter.

Figure 3.18: Demonstration of the feature processing on the highest point and the median in height during a fall from walking scenario. The upper figure shows the unfiltered features, while the lower figure depicts the features after applying the median and Kalman filter.

Discrete Cosine Transform

To present the features to the learning models in a more compact way and simplify the classification task, the one dimensional DCT is applied on each feature filtered. The DCT represents a signal as sum of sinusoids of varying magnitudes and frequencies. It has the property that significant information about the feature scope is concentrated in a few coefficients and a sequence can be accurately reconstructed from less coefficients than its original length [38].

Since there are only 8 features in this application, here the purpose of the DCT is not to reduce the dimensionality, but to represent the pattern of the features in a more efficient way, to improve the classification performance of the learning models.

The feature measurements within the 3 seconds long sliding time windows (see Figure 3.19 in the following section) are interpolated to 6 values per second, obtaining 18 values per time window. Then the one dimensional DCT is applied on each interpolated feature in the dimension of the time. To show the improvement in the classification performance, the models are trained first without DCT, using the interpolated features as input. Then the models are trained with DCT, using the 10 first DCT coefficients instead of the feature values as input.

3.5 Machine learning based fall detection

For training and classification a way to present the features to the learning model has to be defined. After the training of the model, the newly presented samples for classification have to be processed in the same way as it was done with the training data. Therefore a sliding window of 3 seconds is defined as one sample. After the training, a new sample of 3 seconds can be presented to the trained model, and with its help classified whether it contains a fall or not. The sliding window shifts in time steps of 0.4 seconds over the captured data streams of the scenarios. Figure 3.19 depicts the sliding window over a captured fall scene.

The fall times are annotated manually, and depending on if the fall occurs during a sliding window, it is defined as “fall time window“ or “no fall time window“. Due to the short sliding step of 0.4 seconds, successive time windows are overlapping and therefore contain a majority of the same data. In this way an annotated fall can occur in several successive time windows. To simplify the training only those time windows where the fall is occurring at the beginning, are used for modeling “fall time windows“. Time windows containing a fall partially, and time windows where the fall occurs not at the beginning are sorted out for training and testing. In this way, data is deliberately omitted because in the context of fall detection it is less relevant, how time periods containing partial falls are classified as long as the fall is detected once. So for every fall only one “fall time window“ exists. The set of samples obtained contains 7568 sliding time windows without fall and 113 “fall time windows“.

For all sliding windows the features are computed and saved. In a following step they are used to train two different machine learning models, the Hidden Markov Model and the Multilayer Perceptron. A requirement for a successful training and classification is that the features used are significant and allow drawing conclusions whether a fall occurred without any additional information. Therefore the features are investigated within a preliminary feature evaluation in the next section.

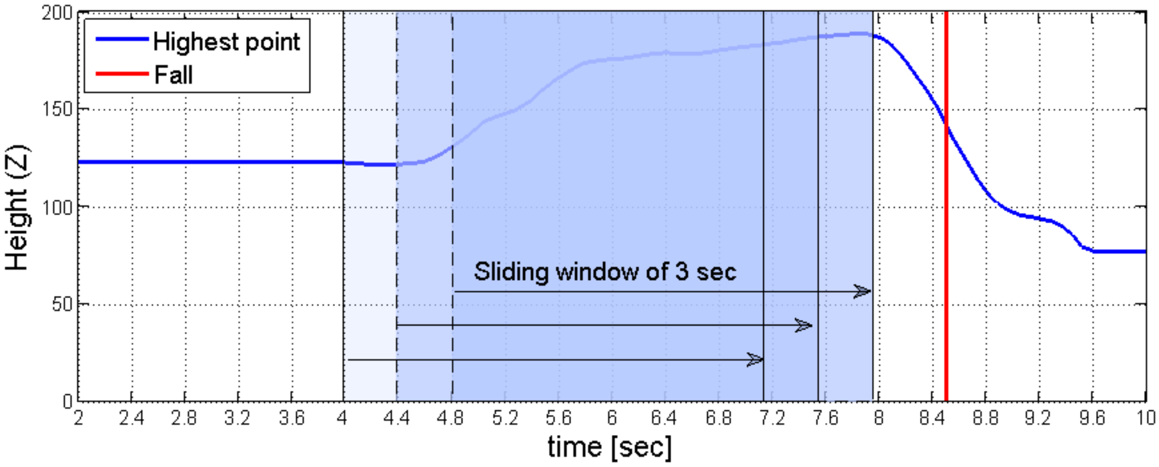


Figure 3.19: Sliding time window over the highest point of a fall scenario.

Feature Evaluation

To evaluate the actual significance of the features, the sliding windows are investigated with respect to the feature scope during fall and no fall scenarios. The aim of the feature evaluation is to investigate, whether the sample set of features computed actually contains significant information about falls, and how distinctive the individual features are. The evaluation method is presented in [8] with consecutive time windows of variable length. In this thesis it is used on the sample set of sliding time windows with uniform length.

If a feature is significant, it alters during falls in a specific way, which allows to drawing conclusions whether a fall occurred or not. As example, the highest point should decrease during a fall time window more than during walking or sitting down. For all features excepted for the activity and the vertical velocity of the highest point, the alternation is measured by the difference between the magnitude at the end of the sliding window and the magnitude at the beginning. In case of the activity, the difference between the magnitude at the end of the time window and the peak point is preferred, because it is not constrained that an activity peak occurs at the beginning of the fall. Similar certainty applies to the velocity because, due to the gravitation, the velocity downwards rises during a fall, and reaches a peak shortly before the impact on the ground. Therefore for the velocity two values are used within the evaluation. First, the minimal velocity and second, the sum of the minimal velocity and the maximal velocity after the moment when the minimal velocity occurs. As described in Section 3.3, downward movements induce negative velocity magnitudes, while upwards movements induce positive velocities. So, higher values of the mentioned sum indicate, if an upward movement, like standing up, occurs after a negative peak of the velocity.

The characteristic numbers obtained for the alternation can be plotted in a bar histogram for every feature. As example the Figure 3.20 shows the histogram of the difference of the highest point for each sliding window, plotted in form of bars. The height of the bars indicates the amount of time windows with the same difference of the highest point. The red bars mark fall time windows, the blue bars stand for time windows with normal movements. The fall time bars form a cluster around -60 cm while the majority of time blocks with normal movements lies between 100 and -100 cm. The histograms allow to visually investigating the significance of the features. If the fall bars form a cluster which can be well separated from the no fall bars, the feature is assumed as significant. In Figure 3.20 the red fall bars form a cluster which is overlapping the cluster of the blue no fall bars. Although a clear separation is not possible the cluster center of the fall bars lies apart from that of the no fall bars. Figure 3.21 shows the histogram for the difference of the Vertical Volume Distribution Ratio. Comparing the two figures, in both cases the fall time blocks form a cluster, but the cluster centers of the second histogram in Figure 3.21, lie closer together. The fall time blocks cannot be separated as good as in the histogram of the highest point and therefore it is assumed that the Vertical Volume Distribution Ratio is a less significant feature.

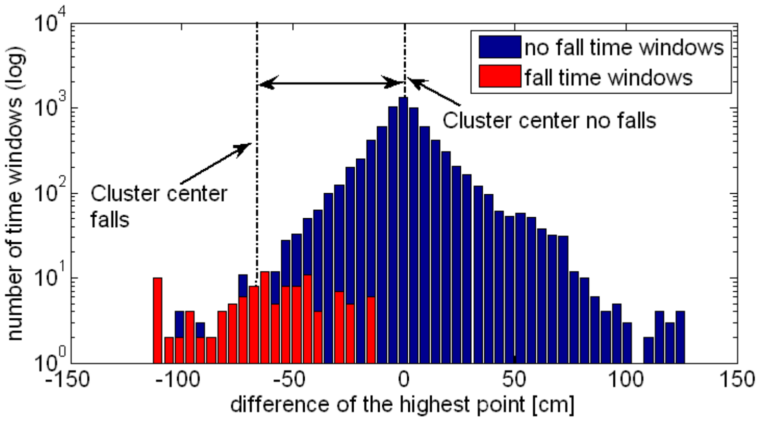


Figure 3.20: Histogram for the alternation of the highest point during the sliding time windows.

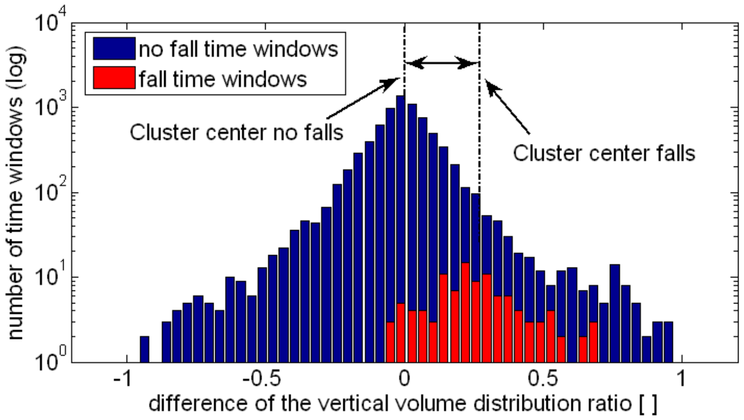


Figure 3.21: Histogram for the alternation of the Vertical Volume Distribution Ratio.

To quantify the significance of the features, thresholds for each feature are empirically determined with the help of the histograms, and computed how well the fall time blocks can be distinguished from the no fall time blocks using these thresholds. In the case of the highest point (compare Figure 3.20), the fall time blocks lie mainly between the thresholds -120 and -30 cm. In the next step, all time blocks are classified with the help of the thresholds defined. That means, all sliding time windows with a highest point difference within these thresholds are classified as falls, and the remaining time windows as no falls. The number of misclassifications is used to compute the decrease of the entropy for every feature as characteristic number for the significance. The entropy is a measure of the unpredictability or disorder associated with a variable. It is assumed that arranging the time windows with the help of a significant feature into fall and no fall time windows, reduces their disorder and so their entropy. In this way the entropy can be introduced as significance measure for the features. Following equations [52] are used to obtain the entropy:

$$H = \frac{p * Ip + n * In}{S} \tag{3.15}$$

where H is the entropy, p is the number of correctly, n the number of falsely classified time blocks, S is the sum of all time blocks (the sample set contains 7681 sliding windows as mentioned in Section 3.5), Ip is the information content of the correct classified and In the information content of false classified time blocks. The information content is computed by following equations [52]:

$$Ip = -log_2(\frac{p}{S}) \tag{3.16}$$

$$In = -log_2(\frac{n}{S}) \tag{3.17}$$

As mentioned above, the decrease of the entropy is used as characteristic number for the significance of each feature. A detailed description of the calculation can be found in [43] and [8]. In summary the decrease of the entropy is computed in the following way:

$$\Delta H_{feature} = H_i - H_c \tag{3.18}$$

where H_i is the entropy for the initial situation, that all time windows are classified as falls. In this situation all fall time blocks are classified correctly and the remaining no fall time blocks falsely (p=113 and n=7568 in Equations 3.15 to 3.17). H_c is the entropy for the situation after the classification with the specified thresholds. Here for p and n the particular amounts of correctly and falsely classified time blocks are inserted. $\Delta\, H_{feature}$ is the decrease of entropy obtained indicating the significance of the feature.

The feature evaluation is used as preliminary evaluation to get a suggestion about which features are most significant and which features may be disclaimed. The descriptors for the alternation e.g. the difference of the highest point during a time window, are used as input for the Multilayer Perceptron in Section 3.5. To compare the actual feature influence within the learning models to the results of the feature evaluation, the training is executed by disregarding particular features and computing the influence on the classification performance.

Classification with Hidden Markov Models

The features of the sliding windows describe the motion and pose of the person during the time periods. For the fall recognition, two Hidden Markov Models are trained with the feature sample set. Sliding windows containing a fall are used to train the “Fall Model“ and time windows with normal movements serve as input to train the “No Fall Model“. During the training, the parameters of the models are set to best explain the input training patterns for each category of movements [17]. To classify a new sample, the features are presented to both models to figure out which model is more probable to produce a similar output.

Figure 3.22 depicts the components of the HMM. The HMM is a stochastic model containing two random processes. The first is the not directly visible, hidden Markov chain. The second process generates the visible output observations. The hidden Markov chain is a mathematical system that examines chainlike transitions from one state to another under the condition that the current state depends only on the previous state. So it is characterized by the states $q(t)$ and the transition probabilities a_{ij} between these states. The second stochastic process generates the visible output observations $o(t)$ with a probability distribution b_{ik} dependent on the states of the hidden process. [33]

In connection with the fall detection in this application, the output observations are defined by the features because they can be observed i.e. measured by the dynamic stereo vision sensor. The hidden Markov states are considered as the movements of the person, which can be estimated with the help of the features.

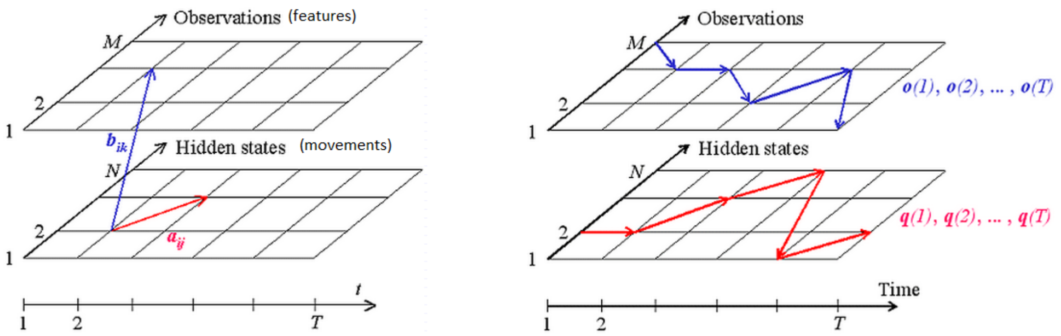


Figure 3.22: Hidden Markov Model [53].

In accordance with the formal definition and figure 3.22, the characterizing parameters of the Hidden Markov Model are summarized in following listing [53]:

Hidden states $Q = \{q_i\}$, i=1,...,N.

Transition probabilities $A = \{a_{ij} = P(q_j \text{ at } t{+}1 | q_i \text{ at } t)\}$, where $P(a|b)$ is the conditional probability of a given b. That means, a_{ij} is the probability that the state q_i is followed by the state q_j.

Observations $O = \{o_k\}, k =$1,...,M. In this application the observations are the measurable features.

Emission or output probabilities $B = \{b_{ik} = b_i(o_k) = P(o_k|q_i)\}$, where $P(o_k|q_i)$ is the conditional probability that the output is the observation o_k given that the current state is q_i.
The observations O can be discrete, with a specified set of output variables, e.g. the symbols of the alphabet, or continuous, where the emission probabilities are modeled by probability density functions. In this application the features achieve continuous values and therefore Gaussian mixtures are used to describe the emission probability density functions for the observations of each state and feature [40].
Gaussian mixtures are combinations of Gaussian or normal distributions and can be written as weighted sum of Gaussian densities. The characterizing parameters of a Gaussian mixture are the number of Gaussians and the weight, mean and covariance matrix of each Gaussian [47].

Initial state probabilities $P = \{p_i = P(q_i \text{ at } t{=}1)\}$. P specifies the initial state arrangement at the time t =1.

Recognition of unknown observation sequences

To classify a new sequence of features into a fall or no fall sequence, it is presented to both trained models, the “fall HMM“ and the “no fall HMM“. Given the above parameters of the two models, the recognition task is to compute the probability of the particular observation sequence for both HMMs, and find out which one is more probable to generate a similar output sequence.

The straight forward way for calculating the probability of an observation sequence with the length T is to enumerate every possible hidden state sequence of length T, and sum up the joint probabilities for the output sequence over all these possible state sequences. The probability $P(O|HMM)$ of a HMM generating the observation sequence $O = O_1O_2...O_T$ is in this case computed by following equation [46]:

$$P(O|HMM) = \sum_{s_1,s_2,\ldots,s_T}^{N} p_{q1}b_{q1}(O_1)a_{q1q2}b_{q2}(O_2)....a_{qT-1qT}b_{qT}(O_T) \tag{3.19}$$

where p is the initial state probability, q is the state, s is the set of possible state sequences, a is the transition probability between the states, O is the output symbol and b is the emission probability of a state to generate a particular output. The parameters are interpreted the following way. Investigating one possible state sequence, initially (at time $t = 1$) the state is q_1 with the probability p_{q1}. In this state the output symbol O_1 is generated with the probability $b_{q1}(O_1)$. At time $t = 2$, the transition from state q_1 to state q_2 is undergone with a probability of a_{q1q2} and the output symbol O_2 generated with a probability of $b_{q2}(O_2)$. At time $t = T$, the transition from state q_{T-1} to state q_T is undergone with a probability of a_{qT-1qT} and the output symbol O_T generated with a probability of $b_{qT}(O_T)$. This multiplication has to be done for all possible state sequences and summed up to obtain the probability of a given observation sequence [46].

If N is the number of states and T the length of the output sequence, there are N^T possible state sequences. Calculating the probability for the output sequence involves on the order of 2T*N^T calculations, which is a high computational effort [46].

Therefore the recursive forward algorithm is used to compute the probability of an output sequence in a more efficient and dynamic way. Figure 3.23 shows the computational flow of the algorithm. It exploits the fact that since there are only N states, all the possible state sequences will merge into these N nodes, no matter how long the observation sequence is. To recursively compute the probability of an output sequence, the auxiliary variable α is introduced. $\alpha_i(t)$ represents the probability that the HMM is in the state q_i at time t having generated the first t elements of the output sequence. $\alpha_i(1)$ is initialized the following way with the help of the initial state probability p of each state and the according emission probability b of the given output symbol $O(1)$ [53]:

$$\alpha_i(t=1) = p_i b_i(O(1)) \text{ with } i\text{=1,...,N} \tag{3.20}$$

Then α is recursively computed for each time step and state using following equation [53]:

$$\alpha_i(t+1) = \left[\sum_{j=1}^{N} \alpha_j(t) a_{ji}\right] b_i(O(t+1)) \text{ with } i\text{=1,...,N and } t\text{=1,...,T-1.} \tag{3.21}$$

The probability of the whole output sequence of length T is then computed with the help of the α at time T [53]:

$$P(O|HMM) = \sum_{j=1}^{N} \alpha_j(T) \tag{3.22}$$

The computational effort of the forward algorithm is in the order of N^2 which is significant lower than the direct computation in Equation 3.19.

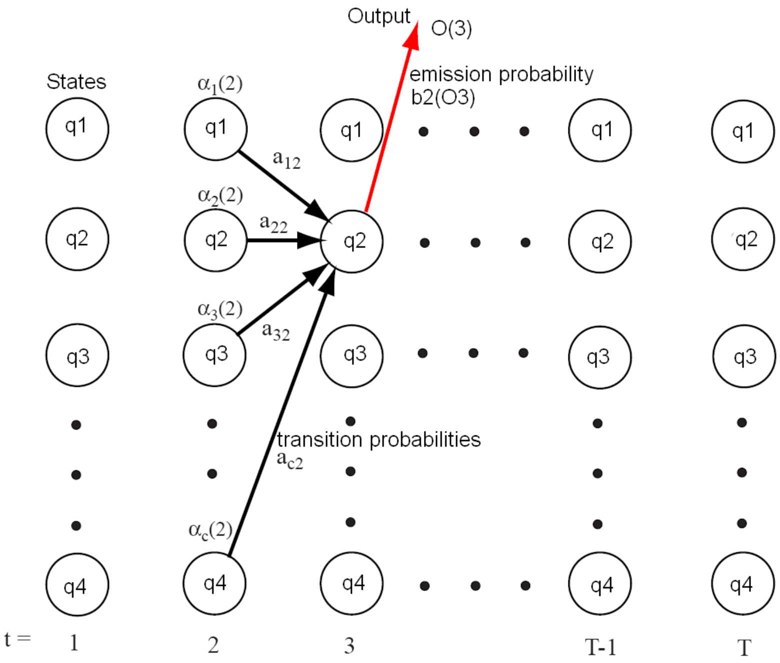


Figure 3.23: Computational flow of the forward algorithm [17].

Training of the HMM

Given the output observation sequences i.e. features of the “fall“ and “no fall“ sliding windows, the task is to find the most likely set of state transitions A, output B and initial state probabilities P for two HMMs, the “fall HMM“ and the “no fall HMM“. Both HMMs are trained separately in the same way but with different input observation sequences. The parameters A, B, P are adjusted in order to best explain the pattern of the presented observations for each class. The number of hidden states N, and Gaussian distributions G, modeling the output probabilities for each state, has to be defined before the training and is not changed during the training. The two parameters define the structure of the HMM and are no training parameters.

To solve the training task, the iterative Baum-Welch, also called forward-backward algorithm is used. This algorithm is also based on computing α. Above, the forward variable $\alpha_i(t)$ is the probability that the model is in state q_i at time t and has generated the output sequence up to step t. Here additionally a backward variable $\beta_i(t)$ is defined analogously to α. Differently from $\alpha_i(t)$, $\beta_i(t)$ indicates the probability that the model is in state q_i at time t, and will generate the remaining of the given output sequence, from step $t + 1$ to time T [17].

The algorithm starts with an initial guess of the model parameters and computes the probability of obtaining the observations presented with the specified HMM. It computes the forward α and backward variables β for each state. With the help of the two values it is estimated how often each transition and emission was used. Then, the parameters of the model are updated according to the estimated occurrence in a way that, the probability of the frequent transitions and emissions is raised. So an optimized model is obtained and the iteration is started from the beginning, computing the probability of the presented observations with the new model. After every successful optimization step this probability i.e. likelihood rises. The iteration is stopped when no significant changes in the model parameters and therefore also in the likelihood, are obtained through the optimization step [33].

A common certainty in most optimization problems is that, the optimization surface is complex and has many local maxima. Reaching such a local maximum, the training algorithm stops even if the parameters are not optimal, because a movement in any direction results in a decrease of the model performance to describe the input observation pattern [46].

To avoid the training to become stuck in a local maximum far away from the global optimum, the initial guess of the parameters is relevant. Therefore especially the emission probabilities B should be rather estimated than randomly determined. K-means clustering is applied to initialize the Gaussian mixtures for each state. Thereby each input observation sequence is divided into as many uniform straps as the specified number of hidden states. That means, if the input observation sequences are 1x12 long vectors of subsequent feature measurements (i.e. the highest point scope during a time window) and the number of hidden states N is 4, all observation sequences are split into $N = 4$ equal sets in the following way: The first set contains the first three feature measurements of all observation sequences (i.e. time windows), the second set contains the following three feature measurements, etc. Then, K-means clustering with as many clusters as the specified number of Gaussian distributions G for each state, is applied on each of the four sets. In this way the means and covariances of the Gaussians, modeling the emission probabilities are obtained.

The initialization of the remaining parameters is not crucial and therefore kept simple. The initial state probabilities are specified as the same for each state. The transition probabilities are initialized as equal for every transition [40].

HMM Toolbox for Matlab

For the classification and training of the HMM, the open source HMM toolbox from K. Murphy [40] with modifications from B. Resch [47] is used.

The HMMs are trained once with the interpolated features and then with the DCT coefficients of the features, to compare the classification performance of both methods.

Further the HMMs are trained using different subsets of features to investigate the influence of the particular features on the models classification performance.

Classification with Multilayer Perceptron

The Multilayer Perceptron as type of artificial neural network is a machine learning approach, inspired by the micro-structure of the brain. The human brain consists of a high amount of interconnected cells, called neurons. Learning and information processing is based on the forwarding of electric signals across these neuronal connections.

Analogously to the neurons of the brain, an artificial neural network is a mathematical model which consists of an interconnected group of artificial neurons. They are represented in the form of a directed graph, where the nodes are understood as the neurons and the arcs as the connections between these neurons [55]. With the help of this architecture, an input vector is mapped to the output vector by propagation through the connections between the neurons. This is similar to the principle of the human brain, where stimuli are sent over connected neurons to the brain, and through activating specific groups of neurons a sensation is generated.

The input vector contains the set of features to be classified. After presenting it to the MLP, an output vector is computed as associated target vector. Different to the HMMs where a fall HMM and a "no fall HMM" is trained, here both classes, fall and no fall, are presented to one MLP. Therefore only one MLP is trained. To classify a new sample, it is propagated through the MLP from the input to the output neurons. In this way the output target vector, which contains the classification results, is computed.

Figure 4.6 shows the general structure of a Multilayer Perceptron. The neurons are organized in layers with feed forward connections where the activations (arrows) move only in one direction, from the input to the output layer. The depicted network has 3 layers, an input, a hidden and an output layer with 3 neurons in each case. Although here it is the case, the number of neurons in each of the layers does not have to be the same. The input layer consists of one neuron for each entry in the input sample vector x, and distributes the values to the neurons of the hidden layer. Arriving at a neuron in the hidden layer, the value x from each input neuron is multiplied by a weight w, and the resulting weighted inputs are added up to a combined variable u at each neuron. This weighted sum u, is fed into a transfer function σ, which outputs the result h. The transfer function is also called activation function and determines whether a neuron fires or not. An example for the activation function is the sigmoid function, $h = tanh(u)$, which is the hyperbolic tangent and ranges from -1 to 1 [55]. The outputs h from the hidden layer are distributed by the weights w to the output layer. Analogously to the hidden layer, here the weighted values are also summed up and fed into the activation function σ. The output is the target vector y for the presented input sample x.

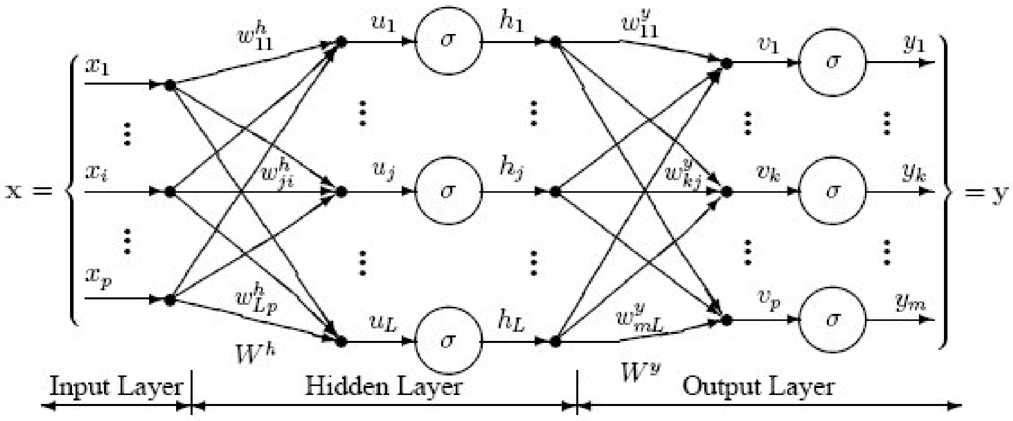


Figure 3.24: Multilayer Perceptron overview [39].

The characterizing parameters of a standard MLP are summarized in the following listing:

Number of hidden layers H. The MLP consists of one input and one output layer. Between these two layers it is possible to use more than one hidden layer, but this may introduce a greater risk of converging to a local minimum during training. Therefore, if a more complex model is needed, it should be first tried to raise the number of neurons in the hidden layer before introducing more layers [39].

Number of neurons in the hidden layer Pt. The number of hidden neurons characterizes the model complexity. If too less hidden neurons are chosen, the network is “under fitted“, which means that it is unable to model decision boundaries for complex data. On the other hand, if too many neurons are specified, the resulting network is “over fitted“. That means, it learns the noise of the training set and therefore fits relatively well to the training data, but generalizes poorly to new unseen data [39].

Number of neurons in the input layer I. The number of neurons in the input layer I is defined by the length of the input sample vector.

Number of neurons in the output layer O. The number of neurons in the output layer O is specified by the amount of target classes. In this application there are two classes which are marked as 0, for no falls and 1, for falls. These two states can be coded by one output variable and therefore the output layer consists only of one neuron.

Weights w_{ji}. At each neuron the inputs are multiplied by specific weights w and summed up. For the particular weight, w_{ji}, the output of the neuron i is multiplied by the weight w_{ji} and forwarded to the neuron j.

Activation function σ. At every neuron the weighted sum is fed into a function which computes the extend of the neurons activation. The output of the function is forwarded to the following neurons.

Recognition of unknown input vectors

Given the weights of the MLP, an unknown input vector is classified by simply propagating the input values through the subsequent weights and activation functions of all neurons. The resulting values of the neurons in the output layer are also called target values and represent the classification result for the specific input. In this case there is only one output neuron which yields between 1 for falls and 0 for no falls. To determine the class, a threshold for this output value has to be defined. With the help of the threshold, it can be adjusted how sensitive the model will be. If the threshold is specified low e.g. 0.2, more input samples will be detected as falls, which means that the classifier is more sensitive. However with a low threshold the probability for classifying many no falls as falls rises and so the model becomes less specific. Finding the optimal threshold means a tradeoff between a too sensitive but low specific, and a too specific but low sensitive classifier. The ideal classifier would be maximal sensitive and specific at once, that means it would detect all falls and no falls correctly.

Training of the MLP

The training of the MLP involves adjusting the weights of the network to optimize the classification performance on the training set used. Therefore the input samples for fall and no fall time windows are presented to the MLP at once, but with different associated target output values (1 for fall, 0 for no fall).

The back-propagation algorithm is used as common method for the training. After a random initialization of the weights, the outputs for the training samples presented are computed. Then the difference between the calculated output and the actual target values is determined, using a performance function F, e.g. the Mean Square Error (mse), which is computed the following way [6]:

$$F = mse = \frac{1}{N}\sum_{i=1}^{N}(t_i - a_i)^2 \tag{3.23}$$

where t_i are the target outputs, a_i are the outputs computed and N is the number of input samples presented.

The weights are updated in the direction in which the performance function i.e. the error decreases most rapidly. To obtain this direction, the gradient of F with respect to the weights is computed by propagating the error backwards from the output to the input neurons. Knowing the error F of the output neurons and the initialized weights, the error and so its gradient can be computed for each neuron backwards. Therefore the training algorithm is called back-propagation.

With the help of the gradient, the weights are updated according to following equation [6]:

$$x_{k+1} = x_k - \alpha_k g_k \tag{3.24}$$

where x_k is the vector of current weights, g_k is the computed gradient and α is the learning rate which determines the speed of the training. This technique is a simple optimization method called gradient descend because the weights are updated in the negative direction of the gradient of the performance function.

The training iterates until the network converges and the gradient gets zero. In this case a local or global minimum of the error function is reached.

To obtain an output between 0 and 1, an activation function which ranges from 0 to 1 has to be chosen. Therefore the Log-Sigmoid transfer function, shown in Equation 3.25, is used as activation function.

$$logsig(u) = \frac{1}{1 + e^{-u}} \tag{3.25}$$

where u is the weighted sum of each neuron.

Matlab Neural Network Toolbox

For the classification and training of the MLP, the commercial Neural Network Toolbox for Matlab is used. Two kinds of MLPs are trained. One MLP is trained with the features DCT coefficients obtaining an input vector with 80 entries (10 DCT coefficients for each of the 8 features). Another MLP is trained with the alternation descriptors introduced in Section 3.5, resulting in an input vector of length 9 (one alternation descriptor for each feature, except the velocity). For a longer input vector, a higher number of hidden neurons is needed and so the complexity of the MLP rises. After training both kinds of models, they can be compared and investigated, whether the complex model is performing better.

Further the networks are trained with different subsets of features to evaluate the actual significance of the particular features for the classification.

Summary

In this chapter the subsequent steps of the methodical approach were discussed. The first section described the main characteristics of the dynamic stereo vision sensor and the data obtained. It was discussed how the AE were processed in order to compute a world coordinate representation of the person related data.

To train and evaluate learning models, different scenarios containing acted falls and normal movements were recorded in a laboratory environment. The set-up of the stereo vision sensor and the scenarios were described in Section 3.1 and Section 3.2.

As next step, features describing the pose and movement of the body were proposed and shown how they can be extracted from the recorded data. To improve the quality of these features they were processed using different filters. To represent the features in an efficient way for the learning models, the DCT coefficients of each feature were computed in the direction of the time.

The significance of the features for fall detection was preliminary investigated with the help of a proposed method for feature evaluation. Finally it was explained how the recorded scenarios were divided into samples of 3 seconds and used for the training of the learning models. The basic characteristics and functionalities of two different learning approaches, the HMM and the MLP, were explained in Section 3.5.

The next chapter shows the results of the feature evaluation and the trained classifiers for fall detection.

CHAPTER 4

Results

In this chapter the features and the models trained are evaluated. The first section presents the results of the feature evaluation with a ranking of the features according to the computed significance for fall detection. Therefore the alternation of the features during the time windows is measured by descriptors like the difference of the magnitude between the end and the beginning of the time slide. The descriptors obtained are visualized in form of histograms for each feature and the significance is quantified with the help of the entropy decrease.

Section 4.2 and Section 4.3 are of identical layout and present separately the results for the two classifiers trained, the HMM and the MLP. Cross validation is used to investigate the classification performance of the models on unseen data. For each model a parameter evaluation is conducted to obtain the optimal model architecture. The architecture of the HMM is defined by the number of hidden states and Gaussian mixtures, while the MLP is defined by the number of hidden neurons. Optimal parameters for the learning models are found empirically by permuting different parameter settings and comparing the resulting classification performances. The settings leading to the best performance are chosen as appropriate for this application.

Then the characteristics of the optimized models are further evaluated with statistical numbers. In addition the feature influence on the performance of the particular models is measured to get information about the actual significance of the features. Therefore the models are trained and validated by using different subsets of features.

Finally in the Section 4.4, features and classification results for different scenarios are presented.

4.1 Feature Evaluation

The evaluation of features described in Section 3.5 results in histograms for every feature and a sorting of the features according to the entropy decrease obtained as significance measure. Figure 4.1 shows the histograms obtained for the visual investigation and determination of the thresholds. For every feature, except the vertical velocity of the highest point, one histogram is generated. As described in Section 3.5, for the vertical velocity two histograms are obtained.

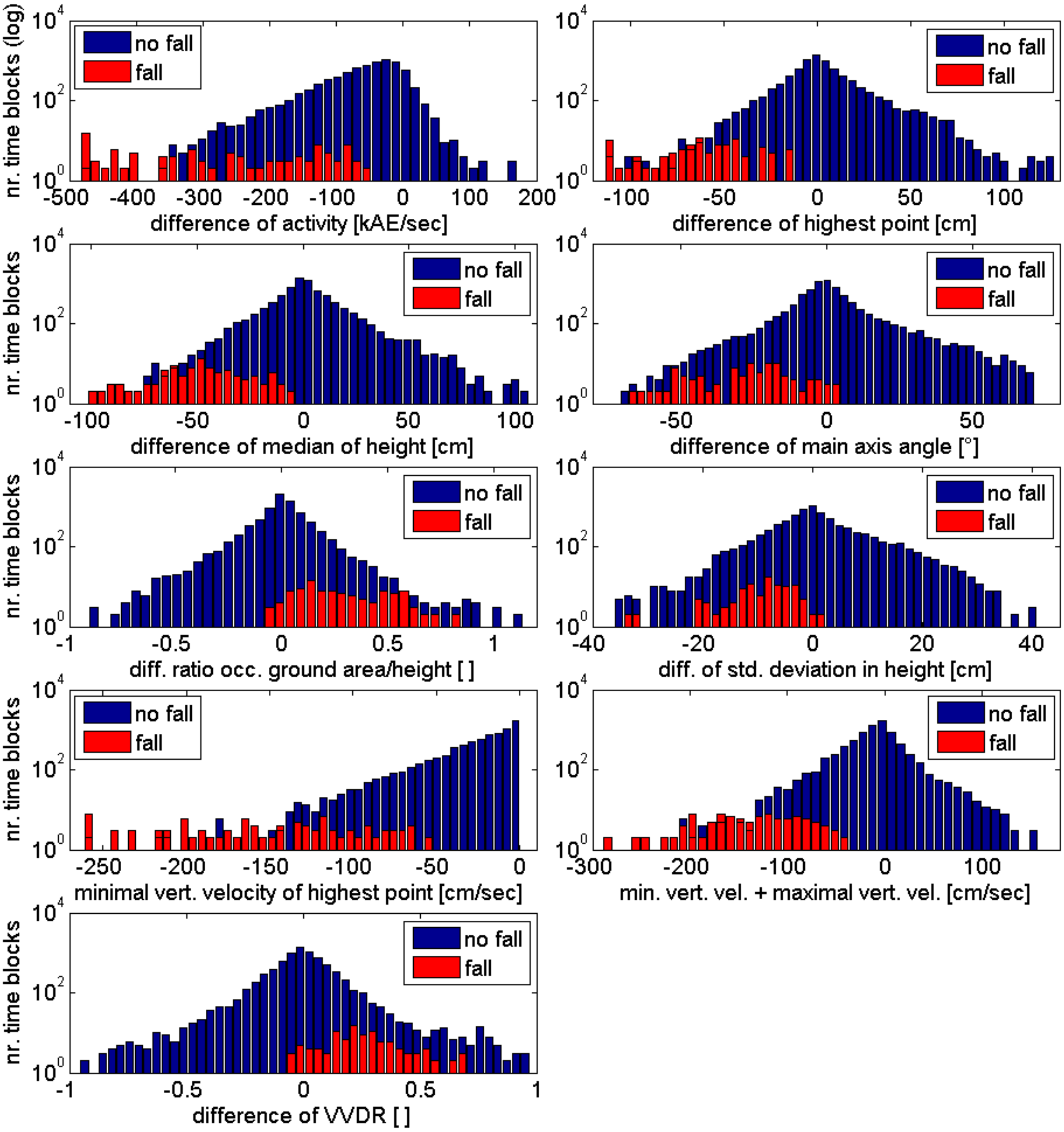


Figure 4.1: Histograms of the feature alternation during the time slides.

Table 4.1 lists the features sorted descending with respect to the decrease of entropy. The feature evaluation shows, that the most significant feature is the minimal vertical velocity of the highest point, while the less significant feature is the activity decrease. The entropy decrease of the first four features lies within a similar range compared to the remaining features, which have an entropy decrease smaller by the factor 10. This suggests that the highest point, the median of the height and the vertical velocity of the highest point are, for this application, more significant for the fall detection, than the remaining features.

Entropy decrease	feature	descriptor for alternation during time window
0,0454	velocity of the highest point	minimal velocity
0,0435	highest point	difference magnitude end-beginning
0,0373	velocity of the highest point	min. vel. + maximal vel.
0,0266	median of height	difference magnitude end-beginning
0,0078	std. deviation of the height	
0,0068	VVDR	
0,0066	angle of the main axis	
0,0053	occupied ground area/height	
0,0001	activity	difference magnitude end-maximal activity

Table 4.1: Descendent sorting of the features according to the evaluated decrease of entropy i.e. significance for fall detection.

4.2 Classification with Hidden Markov Models

In this section the HMMs are investigated with respect to the optimal parameter setting and classification performance. The parameters of the HMM which have to be optimized, are the numbers of hidden states and Gaussian mixtures. Therefore in the first section, cross validation is conducted with varying settings for these parameters, to find out which model architecture obtains the best classification performance.

In the second section, the performance and behavior of the optimized models are further investigated with the help of statistical measures.

In the last section, cross validations are conducted with different subsets of features. It is measured how the features affect the classification performance to find out their actual significance for the HMMs.

Parameter evaluation

The Baum-Welch training algorithm for the HMM (described in Section 3.5) optimizes the transition probabilities A between the states, the output probabilities B for each state, and the initial state probabilities P. However, the number of hidden states N and Gaussian mixtures G per state have to be defined before the training. These parameters describe the architecture of the HMM and are usually empirically determined [26]. There is literature ([26], [51]) that addresses the optimization of the HMM architecture but it is mainly designed for specific speech or character recognition applications. Therefore in this thesis N and G are empirically defined.

The parameter settings are chosen with the aim for keeping the complexity or dimensionality of the models as low as acceptable. The model dimensionality is defined by the number of training parameters which have to be optimized by the training algorithm. Increasing the dimensionality without enlarging the training data set, may lead to a scarcity of the training samples in the high dimensional parameter space. Then, the training data does not cover the parameter space in a sufficient extent and the optimal parameter values cannot be found during the training. Therefore the risk for stopping at a local maximum and obtaining a poorly performing model is higher if too many parameters have to be trained.

The parameters N and G define the models architecture and so the amount of parameters to be trained. For clarification, if more states N are specified, there are more transition A and output B probabilities which have to be optimized for each state. Analogously if more Gaussian mixtures G are defined, there are more means, weights and covariance matrices which have to be optimized for each state and feature.

The optimal settings for N and G are empirically found by training and validating models with different values for the two parameters. To investigate the classification performance of the models on new unseen samples, cross validation is used. Therefore the data set of the recorded fall and no fall scenarios (92 no fall and 113 fall scenarios) is split into $K = 10$ subsets with approximately 20 scenarios. The two classes are as uniformly as possible distributed over the subsets. The cross validation is conducted by using 9 subsets for training and the remaining one for validation. To investigate the different parameter values for N and G, one cross validation is conducted for each parameter setting (e.g. for N=3, G=2). The validation results are averaged over the rounds and for each parameter setting the classification performance is obtained. The parameter setting with the best performance is assumed as being appropriate for this application.

The validation in each round is conducted by classifying the validation subset with the trained model and measuring the classification error of both classes. For the “no fall samples“ (3 seconds sliding windows of no fall scenarios) the error is computed by:

$$Error_{nofalls} = \frac{FP}{TN + FP} \tag{4.1}$$

where FP is the amount of false positive classified samples, that means “no fall samples“ which are wrongly classified as fall samples, and TN is the number of correctly classified “no fall samples“ (true negatives). So $TN + FP$ is the total amount of “no fall samples“.

For the fall samples (3 seconds long time periods containing a fall) the error is computed analogously:

$$Error_{falls} = \frac{FN}{TP + FN} \tag{4.2}$$

where FN is the amount of false negative classified samples, that means fall samples which are classified wrongly as “no fall samples“, and TP is the number of correctly classified fall samples (true positives). So $TP + FN$ is the amount of all fall samples.

As mentioned above, the aim is to keep the number of training parameters low, therefore the cross validation is started with small values for N and G. Within the parameter evaluation an amount of 63 different parameter settings is investigated with 10 rounds of cross validation for each setting. Starting from the combination $N = 3, G = 1$, both parameters are increased and cross validated as long as a successful training is possible. At a certain parameter setting the training data set becomes too small for the amount of parameters and the training is aborted.

Figure 4.2 and Figure 4.3 show the classification results of falls and no falls for the investigated parameter settings. The height of the bars denotes the averaged classification error. Each bar stands for one cross validation with specific parameters, which can be read from the associated axes N and G. Figure 4.2 shows that the classification error for no falls becomes smaller when the HMMs contain more states and Gaussian mixtures. The error of all parameter settings lies below 0.08. The highest error is 0.073 and occurs when 3 hidden states and one Gaussian mixture are specified. The lowest error, 0.0087, is obtained using 9 hidden states and 10 Gaussian mixtures.

In contrast to the no falls, the error of the fall samples in Figure 4.3 behaves more complex. There is no trend showing, that using more states and Gaussian mixtures minimizes the error. The highest error, 0.08, occurs using 8 states and one Gaussian mixture and the lowest error, 0.018 with 4 or 5 states and two Gaussian mixtures.

Figure 4.4 depicts the results in another form. For each investigated number of states the sensitivity and specificity are plotted in dependency of the Gaussian mixtures. The sensitivity of a model relates to the ability to identify positive results, in this case fall samples. It is computed the following way [25]:

$$Sensitivity = \frac{TP}{TP + FN} \tag{4.3}$$

with the same parameters as in Equation 4.2.

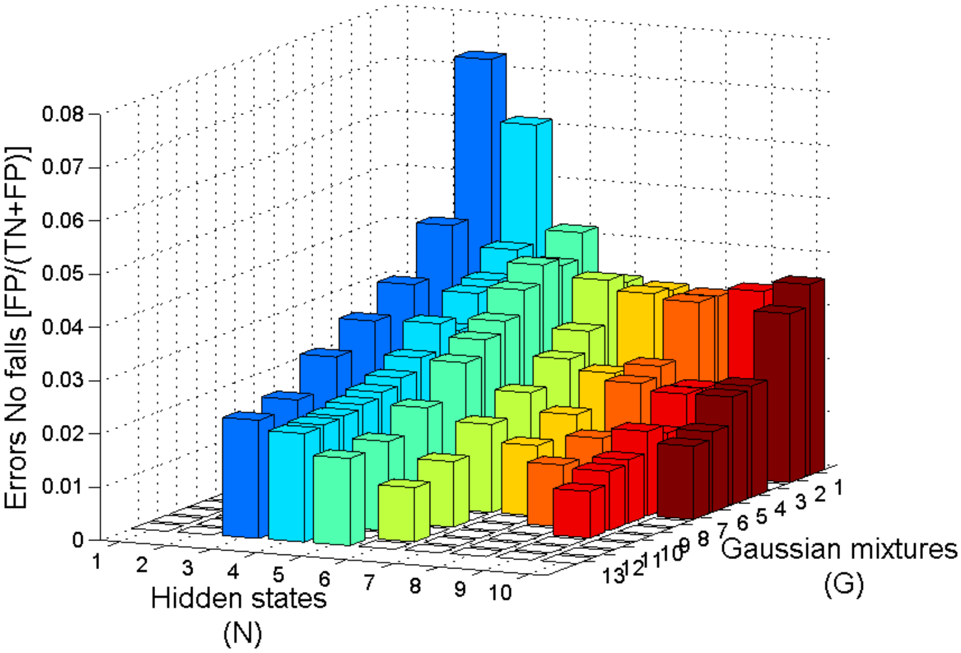


Figure 4.2: Errors for the classification of “no fall samples“ with different HMM architectures.

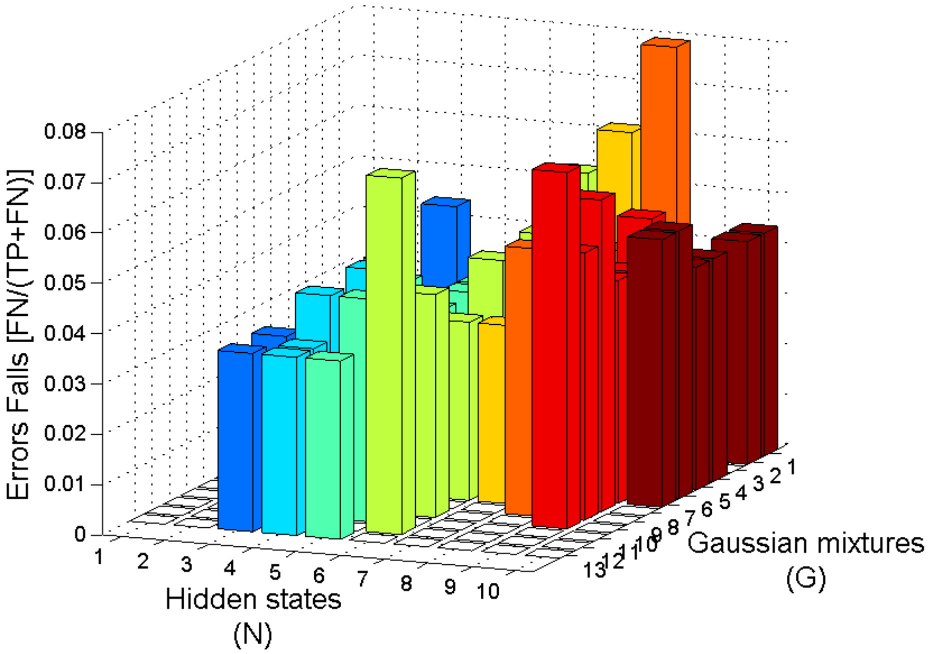


Figure 4.3: Errors for the classification of "fall samples" with different HMM architectures.

The specificity relates to the ability of identifying negative results and is computed by [25]:

$$Specificity = \frac{TN}{TN + FP} \tag{4.4}$$

with the same parameters as in Equation 4.1.

In the lower plot of Figure 4.4, the three graphs for N=3, N=4 and N=5 have a significant lower specificity than the remaining ones. Using more than five states the specificity lies within a similar range for different N and increases when more Gaussian mixtures are defined. This trend accords to the illustration in Figure 4.2 where the error for the no falls decreases with the increasing number of states and Gaussian mixtures.

In Figure 4.4, the upper plot of the sensitivity shows that, in opposite to the specificity, there is no trend showing, that the sensitivity raises when more states and Gaussian mixtures are used. Because of the lower number of positive i.e. fall samples, the graphs are not as smooth as for the specificity. According to the lower error in Figure 4.3 by using a low number of states, the sensitivity raises with lower N. As an example in opposite to the behavior of the specificity, the sensitivity is relatively high for N=3, N=4 and N=5.

Because of the complex and different characteristics for falls and no falls, the best parameter setting cannot be found by simply investigating the overall error. In this case the results of the fall samples would be too less involved because of the unbalanced amounts of samples for both classes. Therefore the most appropriate parameter setting is chosen investigating the specificity and sensitivity in Figure 4.4. The specificity is indirectly related to the classification error of the no fall samples, and the sensitivity to the error of the falls. Because of the different behaviors a tradeoff between high sensitivity and specificity is empirically defined. In this case using

N=7 hidden states with G=8 Gaussian mixtures has pointed out to be appropriate because, the specificity of 0.988 is relatively high compared to the other parameter settings. A high specificity reduces the number of false positive alarms. This is crucial concerning the applicability because when too many false alarms are raised, the fall detection method will not be accepted by the users. On the other hand the sensitivity, 0.965 for N=7 and G=8 is also relatively high compared to the other parameter settings. Though using e.g. N=9 and G=9 would yield a little higher specificity, the sensitivity would be 0.947, which is low in relation to the other investigated parameter settings. Based on this considerations the parameter evaluation shows that the parameters N=7 and G=8 are appropriate for this application.

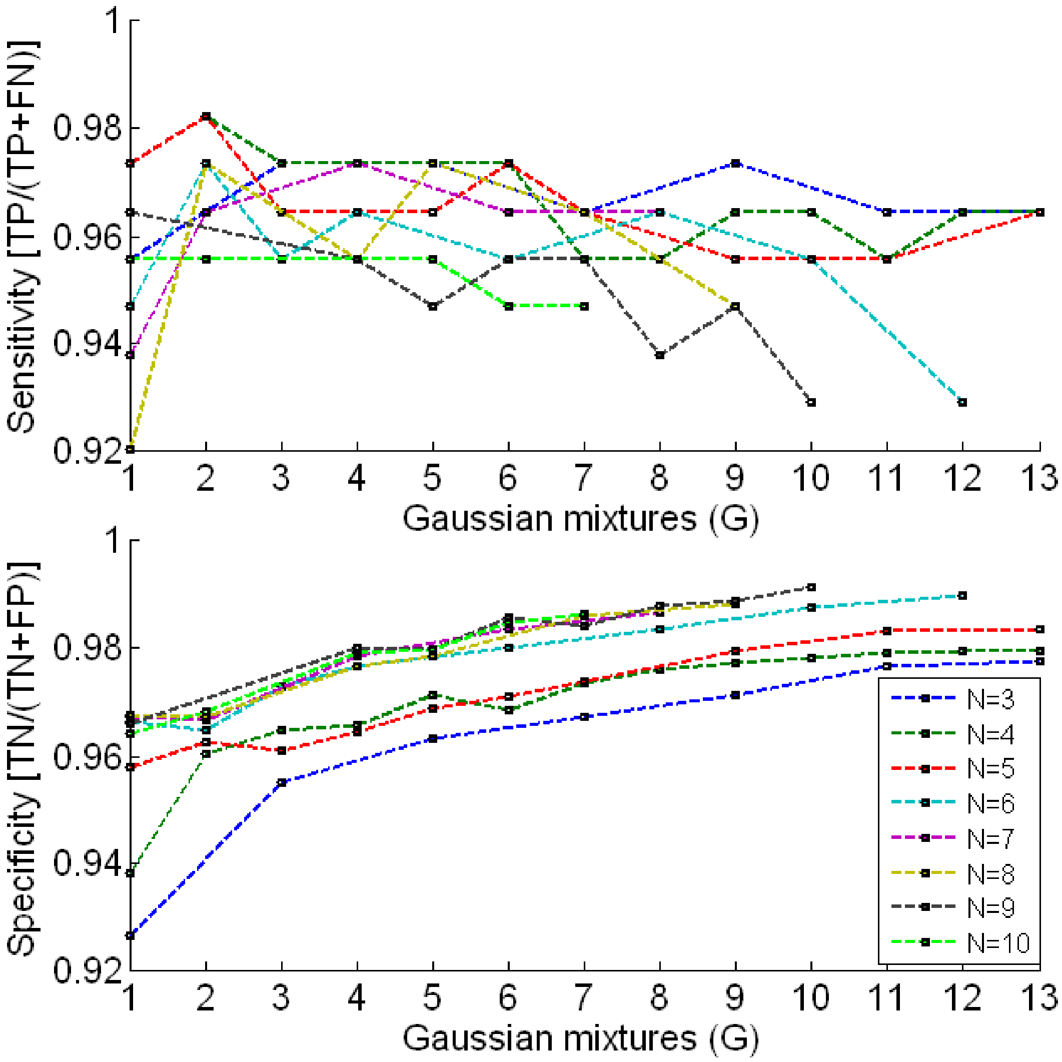


Figure 4.4: Results of the cross validation for the parameter evaluation of the HMM. The upper figure shows the sensitivity, the lower figure depicts the specificity of the different models.

In the next section the characteristics of the resulting HMMs with N=7 hidden states and G=8 Gaussian mixtures are described in detail.

Characteristics of the classifier

To show the models performance in detail, different characteristic numbers are computed. As basic step, the confusion matrix is obtained by summing up all wrong and correct classifications during the cross validation of the model. The confusion matrix contains following values:

True Positives (TP):	Number of correctly classified fall samples.
True Negatives (TN):	Number of correctly classified no fall samples.
False Positives (FP):	Number of wrong classified no fall samples.
False Negatives (FN):	Number of wrong classified fall samples.

Table 4.2 shows the computed values for each round of the cross validation.

Cross validation	1	2	3	4	5	6	7	8	9	10	sum
FN	0	2	0	1	0	0	0	0	0	1	4
Sum Positives	11	11	11	11	12	11	12	11	12	11	113
FP	18	17	3	20	9	11	11	6	1	4	100
Sum Negatives	1563	1251	1372	1127	919	393	274	216	216	237	7568

Table 4.2: Results of the cross validation of the HMM.

Summarizing the results of each round of the cross validation, the confusion matrix in Table 4.3 is obtained.

True Positives (TP) 109	False Negatives (FN) 4	Actual Positives 113
False Positives (FP) 100	True Negatives (TN) 7468	Actual Negatives 7568
Classified Positives 209	Classified Negatives 7472	Total Sum 7681

Table 4.3: Confusion matrix for the classification with HMM.

Based on the entries of the confusion matrix different quality criteria are computed. The sensitivity (also true positive rate or recall) relates the correctly classified positives to the sum of actual positive samples. The specificity or true negative rate measures analogously the ratio between correctly negative classified and the sum of actual negative samples. The formulas for the computation of the two values are described in the above Section 4.2 (see Equation 4.3 and Equation 4.4).

Further derivations from the confusion matrix are the positive and negative predictive values which measure the relation between correctly positive/negative classified samples and the total sums of positive/negative classified samples. The positive predictive value is also called precision.

To combine the recall and the precision in one value a weighted average of these, the F score, is introduced. Computing the F score it can be defined which of the two values is more important for the application.

Following formulas are used to compute the quality criteria [58]:

$$Precision = \frac{TP}{TP + FP} \tag{4.5}$$

$$NegativePredictiveValue = \frac{TN}{TN + FN} \tag{4.6}$$

$$F_\alpha = \frac{(1 + \alpha) * precision * recall}{\alpha * precision + recall} \tag{4.7}$$

with α defining the weight relation of the precision and the recall. In the case of $\alpha = 0.5$ the precision is weighted twice as much as the recall. For $\alpha = 2$ the recall is weighted twice as much as the precision. When $\alpha = 1$ both are weighted the same.

Table 4.4 summarizes the computed values for the trained model. The relatively low value for the precision can be explained by the unequal amount of fall and no fall samples. The actual sum of no fall samples is 7568 while there are only 113 fall samples in the training set. In this way dividing the True Positives which are 109 out of 113 fall samples, by the sum of True Positives and False Positives, which are 100 out of 7568, results nearly 50 %. Because of the low precision, the $F_{0.5}$ score where the precision is weighted more than the recall, is also low, and the F_2 score is higher.

Statistical measure	value
Specificity	0.987
Precision (Pos. Predictive Value)	0.522
Recall (Sensitivity)	0.965
Negative Predictive Value	1.000
$F_{0.5}$ score	0.616
F_2 score	0.752

Table 4.4: Quality criteria of the HMM classifier.

A graphical method to visualize the characteristics of a classification model is the Receiver Operating Characteristic (ROC) curve. Here the sensitivity and false positive rate, which is $1 - specificity$, are plotted against each other for varying model adjustments. Figure 4.5 shows the ROC for the HMM with N=7 states and G=8 Gaussian Mixtures. It is generated the following way. During the classification an unseen sample is presented to both Hidden Markov Models, the fall HMM and the no fall HMM. The result of the classification is the likelihood for both models to produce an output sample similar to the presented one. The model with the higher computed likelihood is assumed as the more probable model. By adjusting the difference between the two likelihoods, the models characteristics are changed. That means, first the likelihood differences of all samples are computed. Then the classification is repeated shifting the decision boundary from the smallest difference to the biggest. In this way first all samples are classified as fall samples. This situation is represented by the point in the right upper corner where the sensitivity is 1 (no False Negatives) and the specificity 0 (no True Negatives). Shifting the decision boundary to the biggest difference, the point in the left lower corner is reached. Here all samples are classified as no falls, which means that the sensitivity is 0 (no True Positives) and the specificity is 1 (no False Positives). An ideal classification would be represented by a point in the left upper corner with a specificity and sensitivity of 1. The red line shows the boundary to the random guess. If the points are lying below this line, the classification performance is not better than randomly guessing the classes and the classifier has to be improved.

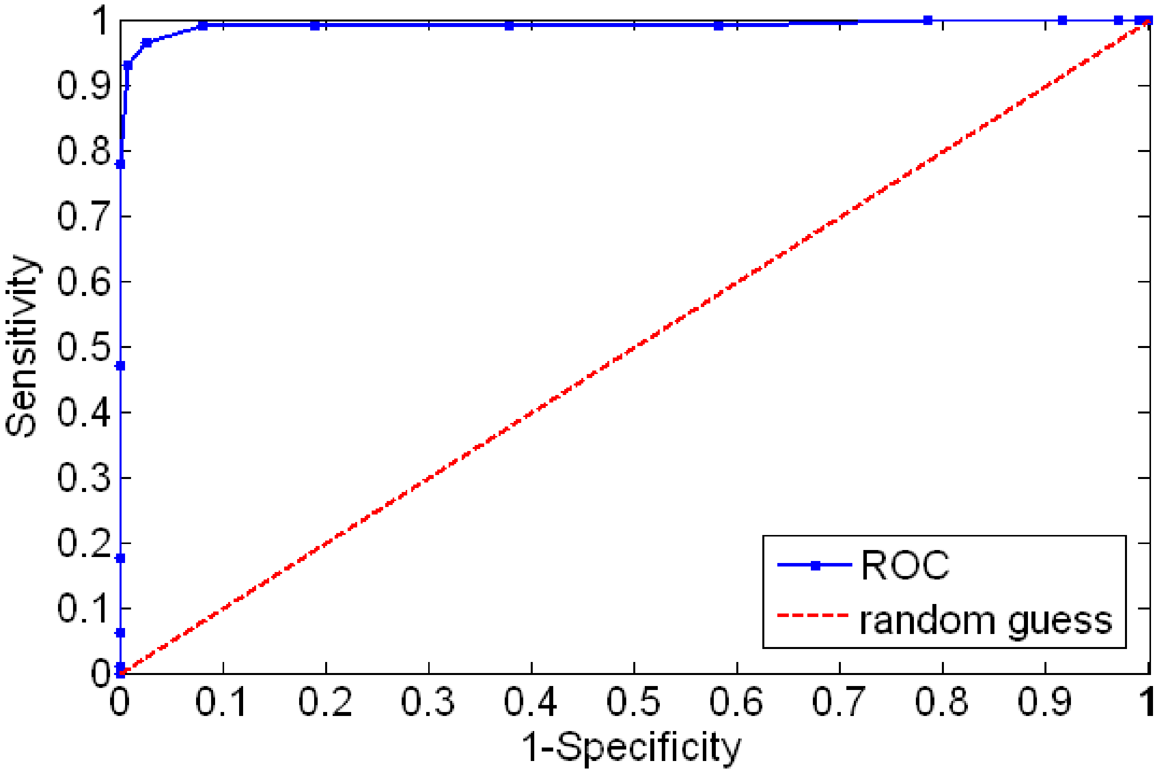


Figure 4.5: ROC of the HMM.

Evaluation of the feature influence

Instead of using the feature values of the time periods directly as input for the classifier, the first 10 coefficients of their DCT are used. To demonstrate the improvement by the DCT, the confusion matrix for the model without the feature transformation is shown in Table 4.5. Compared to the confusion matrix with DCT in Table 4.3, the amount of false classified samples here is higher.

True Positives (TP) 95	False Negatives (FN) 18	Actual Positives 113
False Positives (FP) 496	True Negatives (TN) 7072	Actual Negatives 7568
Classified Positives 591	Classified Negatives 7090	Total Sum 7681

Table 4.5: Confusion matrix for the HMM classification without DCT.

To evaluate the actual significance of the particular features for the HMM classifier, cross validations with different feature subsets are conducted. To investigate every single feature, 8 cross validations are run, leaving out one feature each time. The performance of the classifier is measured with the F_1 score which combines the precision and the recall (see previous section 4.2). Table 4.6 shows the resulting F_1 scores of the models with single features left out. The entries are sorted by ascending F_1 scores. For example, leaving out the first listed feature, the velocity of the highest point, results in the lowest classification performance with an F_1 score of 0.595. So it can be assumed that, compared to the other features, the velocity of the highest point is most significant for the classification performance. According to this evaluation, the VVDR has the lowest influence on the performance.

Excluded feature	F_1 score
Velocity of the highest point	0.595
Activity	0.620
Angle of main axis	0.624
Median of height	0.625
Highest point	0.628
Occupied ground area/height ratio	0.639
Std. deviation of the height	0.660
VVDR	0.667
All features used	0.677

Table 4.6: Single feature influence on the F_1 score of the HMM classifier.

As further investigation, two cross validations with different subsets of features are conducted. The first subset contains only the three best scored features of the histogram based feature evaluation in Section 4.1. According to Table 4.1 these features are the highest point, its velocity and the median of the height. The second subset contains all features except for these best 3 features. Table 4.7 shows the results for the two cross validations. The entries are sorted the same way like in above table with ascending F_1 score. The results demonstrate that, leaving the best scored 3 features out, leads to a lower classification performance than leaving out the whole set of the remaining features. According to this result, the 3 features (median, highest point, velocity) are more significant than the remaining ones but however using all 8 features leads to the best classification performance.

Excluded features	F_1 score
Median of height Highest point Velocity of the highest point	0.420
Occupied ground area/height Activity Angle of main axis Std. deviation of the height VVDR	0.545
All features used	0.677

Table 4.7: Feature influence on the F_1 score of the HMM classifier.

4.3 Classification with the Multilayer Perceptron

This section is of similar layout like the previous but deals with the results of the MLP. First the optimal parameter setting, in this case, number of hidden neurons is evaluated with the help of cross validation for two MLPs with different complexity. Then the characteristics of the trained MLPs are described using statistical values. Further the influence of the features on the classification performance is investigated.

Parameter evaluation

The training of the MLP optimizes the weights for each neuron. The numbers of neurons in the input and output layer are defined by the size of the input and output vectors. For the hidden layer the size depends on the complexity of the data and has to be defined before the training. As described in Section 3.5, choosing the number of hidden neurons means, finding a tradeoff between a too simple classifier, which is not able to learn the pattern of the data sufficiently, and a too complex model, which learns the noise of the training data. Further, specifying a high number of hidden neurons enlarges the number of weights to be trained. As explained in Section 4.2, the number of training parameters should be kept as low as possible to avoid a scarcity of the data in the high dimensional parameter space.

To evaluate different settings for the number hidden neurons Pt, cross validation is used beginning with small hidden layer sizes. The validation is conducted the same way like in the case of the HMM, with the difference that here only one parameter, the number of hidden layer neurons, has to be investigated. For each cross validation an optimal output threshold is determined by calculating the classification error with different thresholds and choosing the one leading to the minimal error.

Figure 4.6 depicts the results of the cross validations with the features DCT coefficients as input and optimized output thresholds. The figure shows that the error of falls decreases when less than about 65 hidden neurons are used and afterwards it increases strongly. The error of the no falls alternates less. It increases slightly before about 65 and then starts to decrease again slightly. Based on the investigation of this plot, an optimal hidden layer size of 65 neurons is chosen because the combined error of both classes, 0.01 for no falls and 0.05 for falls, is minimal at this setting. Respectively the specificity of 0.99 and the sensitivity of 0.95 are maximal with 65 hidden neurons.

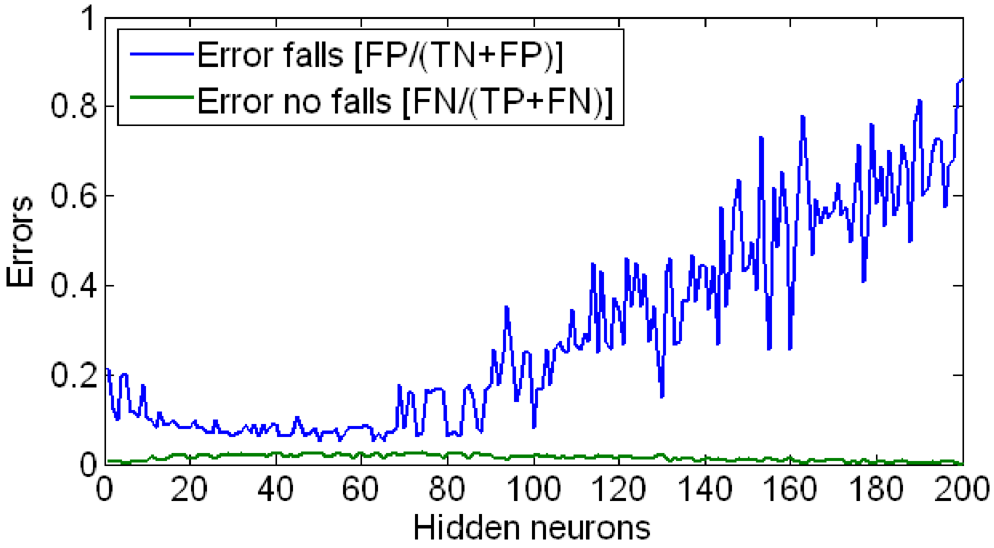


Figure 4.6: Results of the cross validation for the parameter evaluation of the MLP using the DCT coefficients of the features as input.

To evaluate the classification performance of the descriptors introduced in Section 3.5, the cross validation is also conducted with these as input. The aim of the investigation is to find out if the classification is also possible by using only one descriptor for each feature. Compared to the DCT coefficients, the descriptors are less computational expensive and lead to a smaller input vector (9 input values). Using 10 DCT coefficients for each feature leads by contrast to an input vector of length 80.

Figure 4.7 shows the results of the cross validation for different numbers of hidden neurons with the descriptors as input. While using the DCT coefficients the best classification results are obtained with 65 hidden neurons. In this case the number of neurons needed is expected to be lower because the input vector is nearly ten times smaller. Investigating the results shows that using more hidden neurons does not reduce the errors significantly. There are no trends recognizable in the scope of the errors. Although the error decreases around 100 neurons, this number is not preferable because of the small input layer of 9 neurons. Considering this, 1 hidden neuron is assumed to be the optimal choice. The errors are in this case 0.04 for falls and 0.05 for no falls. The sensitivity is 0.96 and the specificity 0.95.

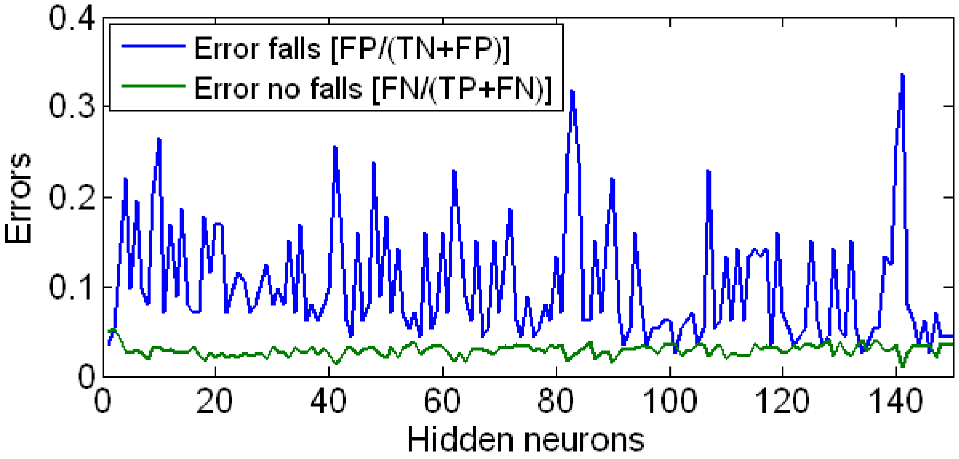


Figure 4.7: Results of the cross validation for the parameter evaluation of the MLP using descriptors of the feature alternation as input.

Characteristics of the classifier

The characteristics of the MLP classifier are described like in the case of the HMM with quality measuring numbers. Table 4.8 shows the results of the cross validation for the MLP with 65 hidden neurons using the DCT coefficients of the features as input. The results for the MLP with one hidden neuron and the descriptors for the feature alternation as input, are listed in Table 4.9. The sum of false classified falls (FN) is lower when using the alternation descriptors while the number of false classified no fall samples (FP) behaves the opposite way. Using the DCT coefficients results in 103 false positives. Using the feature alternation descriptors 265 false positives are obtained.

The results of the cross validation are summarized in the confusion matrices for both MLPs (65 and 1 hidden neuron). Table 4.10 and Table 4.11 show the confusion matrices. Using the less complex model with simple features (Table 4.11) leads to a lower classification error of the fall samples but to a significant higher amount of false positive classified samples.

Table 4.12 shows the quality measures for both models. The equations for the computation of the values are described in Section 4.2 where they are used to measure the classification performance of the HMM.

Cross validation	1	2	3	4	5	6	7	8	9	10	sum
FN Sum Positives	1 11	1 11	0 11	0 11	1 12	0 11	1 12	0 11	1 12	1 11	6 113
FP Sum Negatives	11 1563	28 1251	9 1372	18 1127	20 919	2 393	4 274	4 216	2 216	5 237	103 7568

Table 4.8: Results of the cross validation of the MLP with the DCT coefficients of the features as input.

Cross validation	1	2	3	4	5	6	7	8	9	10	sum
FN	0	0	0	0	0	0	1	0	1	1	3
Sum Positives	11	11	11	11	12	11	12	11	12	11	113
FP	69	66	12	24	13	7	17	8	15	34	265
Sum Negatives	1563	1251	1372	1127	919	393	274	216	216	237	7568

Table 4.9: Results of the cross validation of the MLP with the alternation descriptors of the features as input.

True Positives (TP) 107	False Negatives (FN) 6	Actual Positives 113
False Positives (FP) 103	True Negatives (TN) 7465	Actual Negatives 7568
Classified Positives 210	Classified Negatives 7471	Total Sum 7681

Table 4.10: Confusion matrix for the MLP classification using the DCT of the features.

True Positives (TP) 110	False Negatives (FN) 3	Actual Positives 113
False Positives (FP) 265	True Negatives (TN) 7303	Actual Negatives 7568
Classified Positives 375	Classified Negatives 7306	Total Sum 7681

Table 4.11: Confusion matrix for the MLP classification using descriptors of the feature alternation.

Statistical measure	DCT	alternation
Specificity	0.986	0.965
Precision (Pos. Predictive Value)	0.510	0.293
Recall (Sensitivity)	0.947	0.974
Negative Predictive Value	1.000	1.000
$F_{0.5}$ score	0.602	0.382
F_2 score	0.736	0.549

Table 4.12: Quality criteria of the two MLPs, the complex MLP, which uses the DCT of the features and the simple MLP, which uses the descriptors of the feature alternation as input.

To compare the results graphically and investigate the behavior of the MLP classifiers, the ROC curves are generated for both models by varying the threshold for the output neuron. The output neuron generates an output between 0 for no falls and 1 for falls. The obvious threshold for the classification is 0.5, that means if the output is higher than 0.5, a fall is detected. By varying this threshold between 1 and 0, the sensitivity and specificity can be adjusted. Figure 4.8 shows the ROC curves obtained of both models. It can be recognized that the ROC of the more complex MLP (DCT) yields a higher sensitivity and specificity, but it reaches a sensitivity of 1 later than the ROC of the simple MLP (alternation). That means, using the complex model a higher specificity and sensitivity can be obtained at once because the values are closer to ideal point in the left upper corner. However with the simple model a higher sensitivity can be obtained but a decrease of specificity has to be accepted.

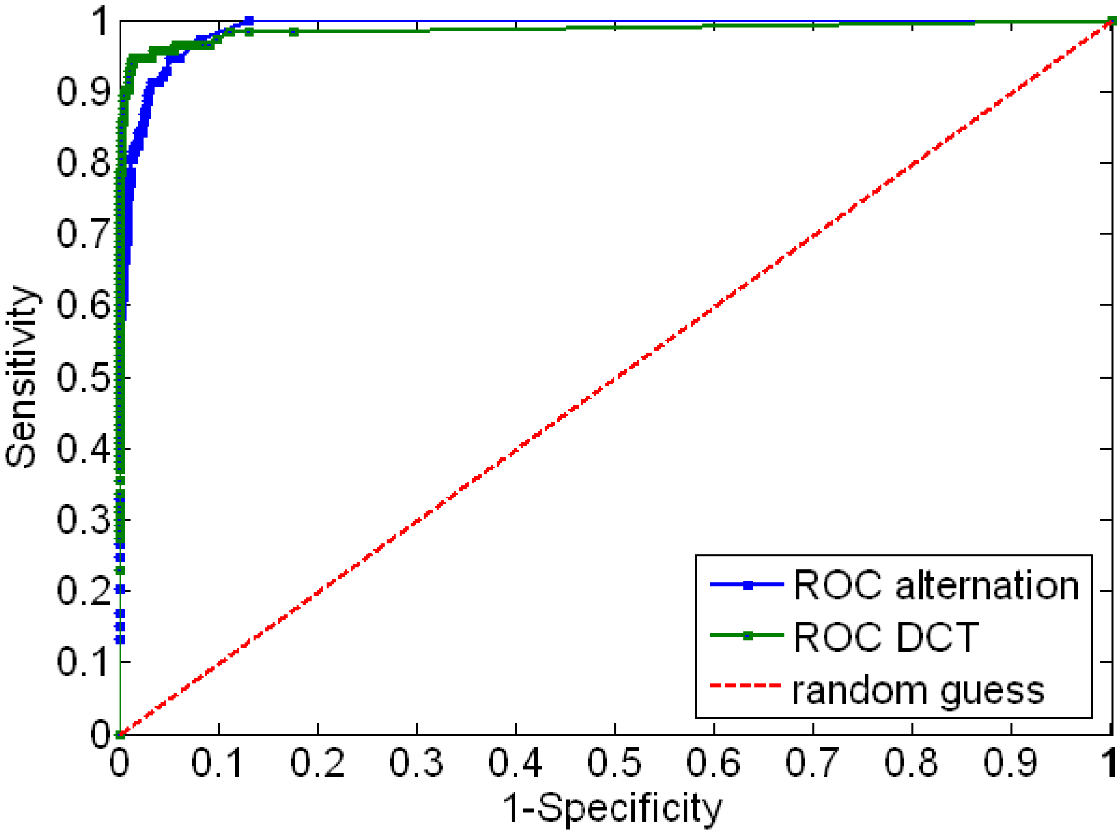


Figure 4.8: ROC of the MLPs.

Evaluation of the feature influence

The evaluation of the actual feature influence is conducted for both models by measuring the F_1 of the classifiers when different subsets of features are used. Table 4.13 shows the results of the investigation for the simple model, while Table 4.14 presents the results for the complex model. The features are sorted with respect to an ascending F_1. In the simple model, the highest point has the most influence on the F_1 score while in the complex model, disclaiming the activity leads to the most decrease of the F_1 score. When using all features the complex model has a higher F_1 score. By comparing the results of the complex MLP with DCT, to the results of the HMM with DCT (Table 4.6), the order of the features does not exactly match but general similarities can be found. The activity, the velocity and the median of the height are relatively significant for the classification in both cases, while the VVDR and the standard deviation of the height have less influence.

Excluded feature	F_1 score
Highest point	0.300
Activity	0.301
Min. + Max. Velocity of the highest point	0.332
Angle of main axis	0.331
Min. Velocity of the highest point	0.336
Median of height	0.340
Std. deviation of the height	0.341
Occupied ground area/height ratio	0.346
VVDR	0.361
All features used	0.451

Table 4.13: Single feature influence on the F_1 score of the MLP classifier using the alternation descriptors of the features as input.

Excluded feature	F_1 score
Activity	0.353
Median of height	0.429
Velocity of the highest point	0.454
Highest point	0.484
Occupied ground area/height ratio	0.500
VVDR	0.539
Angle of main axis	0.570
Std. deviation of the height	0.574
All features used	0.663

Table 4.14: Single feature influence on the F_1 score of the MLP classifier using the DCT coefficients of the features as input.

The results for the simple model can be compared to the results of the feature evaluation in Section 4.1. In both cases the same descriptors for the alternation of the features are used to classify the samples. In the feature evaluation with the histograms, the activity has the lowest significance, while according to the results of the MLP in Table 4.13, it can be located on the second place of the ranking. The highest point and the velocity are high significant in both cases, while the angle of the main axis is more significant in the MLP than assumed within the feature evaluation.

When leaving out subsets of features, both MLPs behave the same way like the HMM. Table 4.15 shows the results for different feature subsets. Leaving out the median of the height, the highest point and the velocity leads to a lower F_1 score than leaving out the other features. This accords to the results of the feature evaluation, that these three features are most significant for the classification. The F_1 scores of the simple MLP are lower than the values of the complex MLP. The lowest F_1 score occurs within the simple MLP model. There it decreases from 0.451 to 0.15 when the three most significant features are excluded.

Excluded features	F_1 score DCT	F_1 score alternation
Median of height Highest point Velocity of the highest point	0.436	0.15
Occupied ground area/height Activity Angle of main axis Std. deviation of the height VVDR	0.539	0.3241
All features used	0.663	0.451

Table 4.15: Feature influence on the F_1 score of the MLP classifiers.

4.4 Classification Examples

To give an imagination how the classification of unseen examples works, Figure 4.9 demonstrates the classification results of the trained models for four different scenarios. Each column shows the results of the models for one scenario. The first row contains the results of the HMM classifier which consists of the two trained HMM models, one for fall and one no fall. It shows the logarithm of the likelihood for both models to obtain a feature output sequence like the presented one. The likelihood of the models is the output of the classification because the input sequence is assigned to the class with the higher probability.

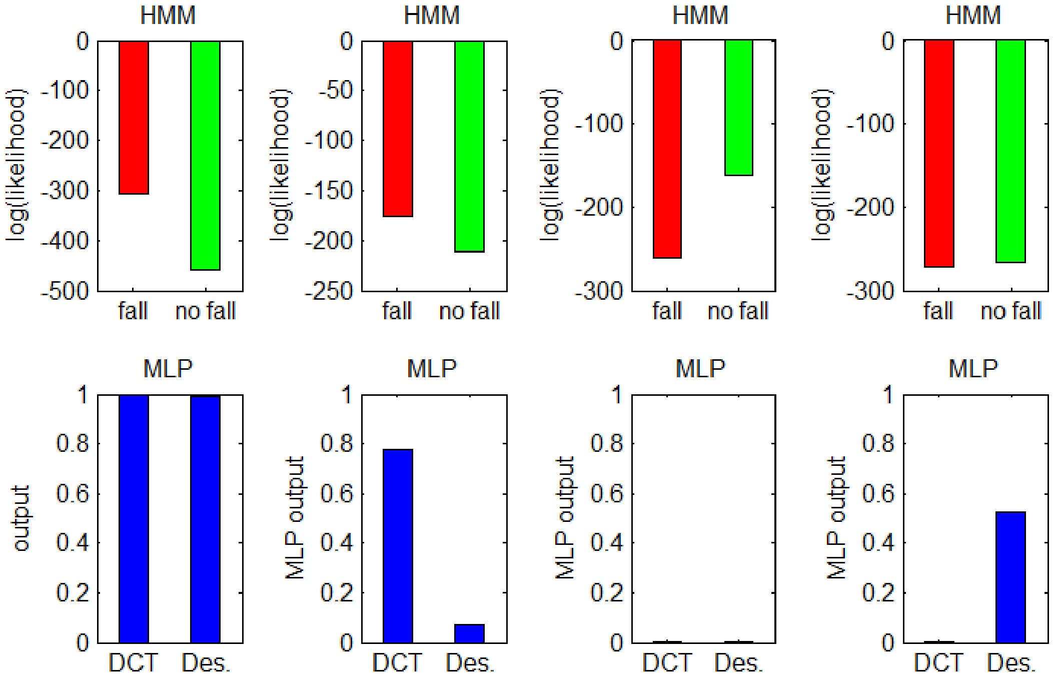
The second row shows the according output of the complex MLP which has 65 hidden neurons and uses the DCT of the features, and the simple MLP which uses the alternation descriptors and has only one hidden neuron (marked with Des. in the figure). For clarity, in the case of the MLP each bar belongs to one MLP classifier, which gives an output between 0 for no falls and 1 for falls. In contrast, for the HMM both bars belong to one classifier and the classification is conducted by presenting the sequence to the two HMMs comparing the resulting probabilities for both models.

The associated features which are the input for the models and therefore basis for the classification, are depicted in Figure 4.10 for the falls and in Figure 4.11 for the no fall scenarios. The first fall scenario has more distinctive features than the second one because the alternation of the features is higher in the first case. This depends on the height of the person, the kind of fall and the pose of the body after the fall. In the first case the log-likelihood of the fall HMM (red bar) is higher than the log-likelihood of the no fall HMM (green bar). That means, the fall is correctly classified by the HMMs. The output of both MLPs is one for the first fall, so both MLPs detect the fall also correctly.

The results for the second fall reflect that the features are less distinctive in this case. The difference between the probabilities of the two HMMs is smaller but still the likelihood of the fall HMM is higher. So the fall is classified correctly but with less confidence. The complex MLP (DCT) outputs about 0.8 while the simple MLP (Des.) has 0.07 as output. This demonstrates the influence of the threshold because depending on which threshold is specified, the fall is classified correctly or wrongly. If for example 0.5 is the threshold, in this case, the complex MLP yields a correct classification, while the simple MLP fails.

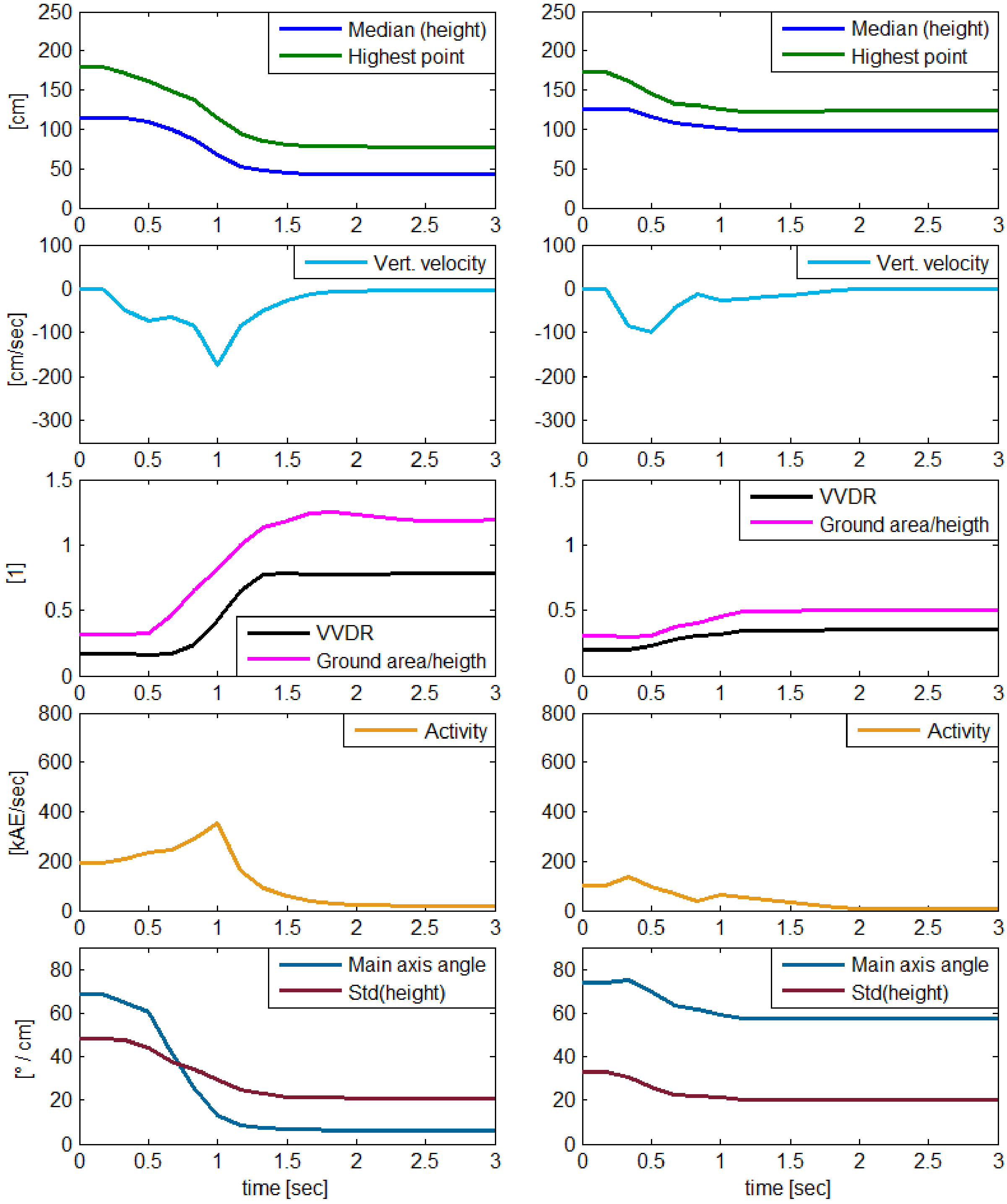
The third column shows the results for a walking scenario. Figure 4.11 shows that the features are relatively constant except the VVDR which is alternating below 0.5. The results for this sample in the third column of Figure 4.9 show that the likelihood of the no fall HMM is higher which means that the sample is classified correctly. The difference between the two likelihoods is also relatively high, so the result has a high confidence. This accords to the fact that the sample is clearly distinctive from a fall sample. The outputs of the two MLPs are both near zero which means that they classify the sample also correctly.

The fourth column shows a no fall scenario which has similar feature scopes to a fall scenario because the person is sitting down. The likelihood of the no fall HMM is still higher but the difference to the probability of the fall HMM is small. The output of the complex MLP is nearly zero while the simple MLP yields 0.5 as output. That means, the complex MLP classifies the no fall sample correctly while the simple MLP classifies the sample only correctly if the threshold for falls is specified higher than 0.5.



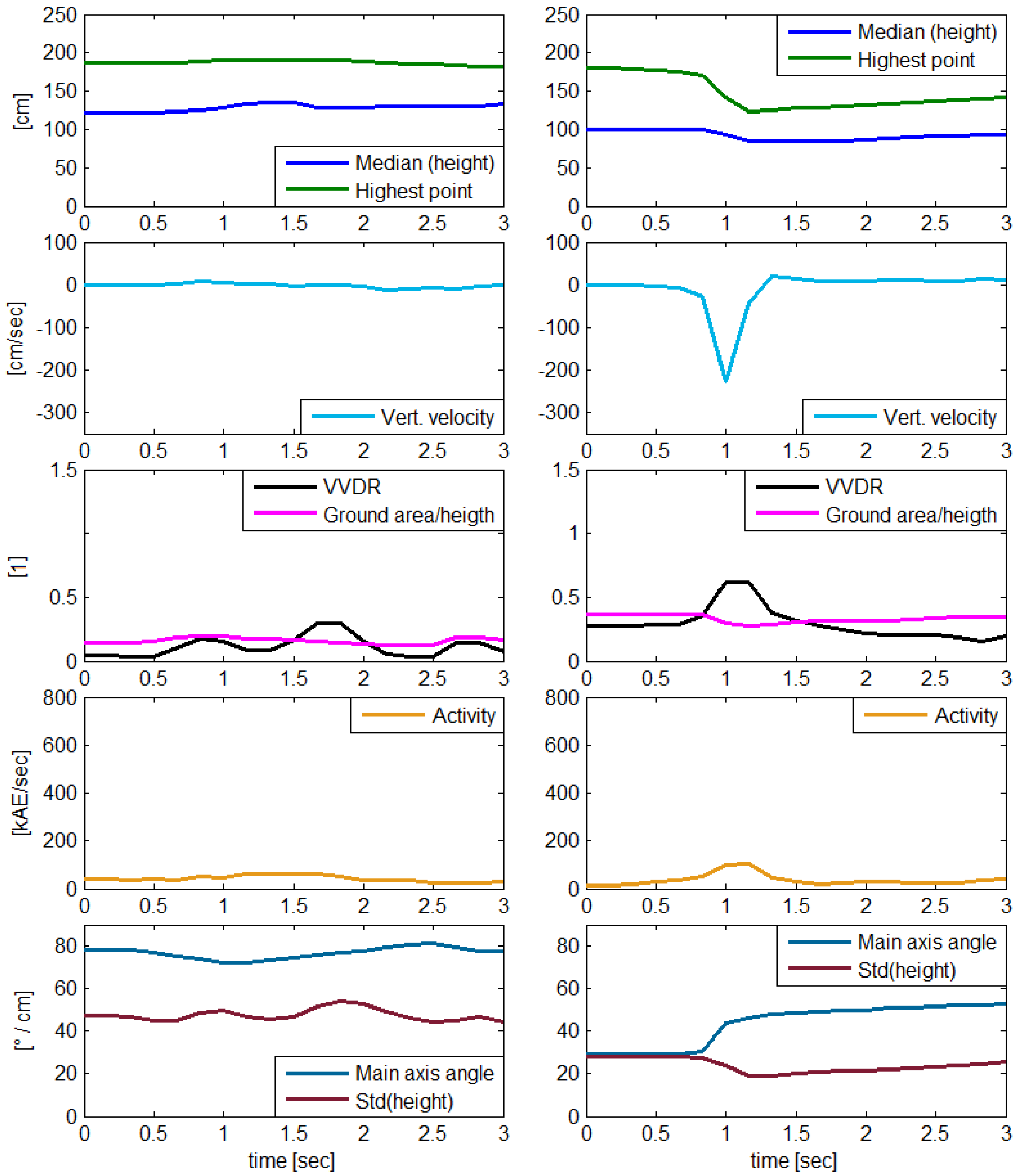
(a) Fall 1 (b) Fall 2 (c) Walk (d) Sit down

Figure 4.9: Classification results for different scenarios. Each column contains the results of the models for one scenario.



(a) Fall 1 (b) Fall 2

Figure 4.10: Features for two fall scenarios. Each column shows the features for one fall scenario.



(a) Walk (b) Sit down

Figure 4.11: Features for two no fall scenarios. Each column shows the features for one scenario.

Summary

In this chapter first the computed features were preliminarily investigated with respect to the significance for the fall detection. Therefore the alternation of the features was computed for the time windows and visualized in histograms. To quantify the significance, a characteristic number, the decrease of entropy was introduced.

As next step the set of time windows with associated features was used to train HMM and MLP models. The optimal parameter settings for both models were investigated with the help of cross validation on the sample set. For each model a 10-fold cross validation was performed with different parameter settings. In the case of the HMM, two parameters, the numbers of hidden states and Gaussian mixtures, were optimized. For the MLP the optimal number of hidden neurons had to be found within the parameter evaluation.

Then the classification performance of the optimized models was analyzed by quality measuring numbers and ROC curves obtained from the cross validation.

The influence of the features was investigated by measuring the classification performance when leaving out particular features. The results were compared to the preliminary entropy based feature evaluation.

Finally the results of the trained models for two fall and two no fall scenarios were demonstrated. For each scenario the features computed which were fed into the models and the according classification outputs were depicted.

CHAPTER 5

Conclusion

This thesis presents a methodology for fall detection using an automatically learned classifier based on the 3D data of a dynamic stereo vision sensor. Within this thesis a database of fall scenarios and normal movements is created with the sensor. Thereby it is important that, the database contains not only different kinds of falls but also a diversity of normal movements, which have a high risk of being misclassified as falls e.g. sitting down or lying down. To investigate the fall scenarios, it is necessary that the falls are acted as realistic as possible which means that they happen during every day movements like bending down, standing up from a chair etc. The database is the basis of the proposed method for fall detection because with its help the learning models are trained and evaluated.

The proposed features include as much information as possible about the pose and motion of the body. An aim of the thesis is also to investigate whether the theoretically promising features are actual significant in the specific application. For example the VVDR might be on the first thoughts a significant feature because it describes the vertical distribution of the person related data points. Analysing the scope of this feature, it can be recognized that it increases during falls because at the end of the fall the person is nearer to the ground plane. But on the other hand, it was found that the feature also alternates relatively high during normal movements like walking or bending down. So it is less significant for fall detection than previously assumed. Therefore the alternation of the features is investigated for two different classes of time samples containing falls and normal movements. Within the feature evaluation, the alternation of the features is visualized in form of histograms for every feature. The results of this evaluation showed that during different fall samples, the features have a similar alternation and form a cluster in the histogram. An ideal feature would form a cluster of fall time samples, well distinguishable from the no fall samples. Investigating the histograms of the presented features, no feature is significant enough to distinguish falls from no falls sufficiently by its own.

To investigate the actual feature influence on the classification performance of the two classifiers, they were cross validated with different subsets of features. Leaving out every feature and comparing the performances is another form of feature evaluation. It is assumed that if the left out feature is higher significant, the performance of the classifier decreases more than for the case of low significant features. To take up the above example of the VVDR, the histogram based feature evaluation assigns this feature to the lower significant group. Leaving out this feature for the classification, influences the performance of the HMM and the simple MLP. So it can be assumed that the VVDR is not as significant as the remaining ones. On the other hand the exact ranking of the features is not similar within the different evaluations. This shows that the significance of the features depends strongly on the learning approach and the representation of the features. Using the alternation descriptors in the simple HMM, only the difference of the magnitude between the end and the beginning of the time windows is the input for the classifier, while with the help of the DCT it is possible to capture more details of the feature scope. So it is explainable that in the different approaches the features have different significances. However in the case of the activity a striking difference between the histogram based feature evaluation (Table 4.1) and the results of the MLP evaluation (Table 4.13 and Table 4.14) arises. The preliminary feature evaluation ranked the activity as least significant feature while leaving it out for the complex as well as the simple MLP classifier leads to a relative high performance decrease. Considering that the simple MLP uses the same alternation descriptors as the histogram based feature evaluation, the strongly divergent significance results for the activity, suggest that also the combination of the features plays a decisive role for the significance. While within the preliminary feature evaluation only one particular feature is used for a classification and the resulting decrease of entropy is computed as characteristic number, in the MLP, the combination of all features except the particular one is used for classification. It is possible that, a relatively low distinctive feature like the activity, can still strongly improve the classification performance when it is combined with other features.

Analysing the feature evaluation with the classifiers shows that even if some features are less significant or contain similar information (e.g. the highest point and the median), using all features results still in the best classification performance.

Although the computations within the scope of this thesis are performed in Matlab on a PC, the fall detection method is done under the motivation that it may be implemented on the board of the stereo vision sensor which has less capacity for the computation. Therefore the feature evaluation can be used to find a selection of features which can be disclaimed without decreasing the classification performance too much.

Comparing the results of the different learning approaches shows that in this case using the DCT coefficients is the most effective way to represent the feature scope. With DCT, the HMM brings the best classification performance but the MLP differs only in 2 false negatives and 3 false positives. Although the simple MLP with one hidden neuron and feature alternations as input, classifies the falls better than the other models, there occur more than twice as much false positives. This suggests that the simple alternation of the features during a time window is not sufficient to model the high variety of movements.

Disadvantages

In the results it was shown that the classifiers trained yield a relatively high sensitivity and specificity. Since the scenario of a fall is even in elderly life the exceptional case it occurs rarely. So the acceptance of the fall detection method in real life will mainly depend on the number of false alarms per day. Summing up the length of all recorded scenarios used as basis for this thesis, results in a total length 54 minutes. With a 3 second sliding window of 0.4 seconds overlapping, these 54 minutes yield to about 7600 time windows as samples to be classified. In the case of the HMM, the number of false positives is 100, which is a small amount in relation to 7600 input samples but quoting it with regard to the time length, arises in 100 false alarms in 54 minutes, which is not acceptable. Therefore the ROC curve of the classifier shows whether the sensitivity can be decreased without loosing too much specificity. The scope of the ROC curve of the HMM (Figure 4.5) implies that adjusting the threshold for the difference of the likelihoods, the classifier can be tuned to lower sensitivity without a strong specificity decrease. The ROC curves of the complex MLP (Figure 4.8, green) shows a similar behaviour while the simple MLP yields a more flat ROC curve and so decreasing the sensitivity here, leads to a higher decrease of specificity.

The characteristic of the dynamic stereo vision sensor is that it generates data asynchronously, that means if less illumination changes are occurring, less data is available and therefore the features are less reliable. With the help of the Kalman filter, the features can be estimated during low activity periods but still there is an uncertainty in the feature computation when the person moves little or is further away from the sensor. For the case of the recorded scenarios, the couch in the laboratory environment (Figure 3.6) is about 6 meters away from the sensor. Since the activity decreases with distance, it was recognized that 6 meters are the limit where the activity is sufficient enough to gain reliable information about the pose of the person.

Another disadvantage of the method might be, that even if no image is obtained by the sensor, customers may feel observed because of the affinity of the sensor with common surveillance equipment.

Advantages

The promising results of the classifiers show that the models can be adjusted and tuned to obtain a lower false positive rate without a significant loss of specificity. The classification outputs provide a confidence measure which can be used for further verification of fall alarms.

The proposed method can be compared to other vision based approaches for fall detection in existing works (see Section 2.3). The majority uses conventional cameras, and proposes methods to calculate 3D features based on the 2D images, which is error-prone and therefore the actual pose of the body can be estimated only inaccurately. In opposite here the features are extracted from 3D point clouds obtained by a stereo vision sensor, which makes the features more robust. The 3D data enables to compute the persons pose and movements in the world coordinate system which is necessary to model the characteristic patterns of motions in a comparable way independently from the camera view and position.

The advantage of the learning approach is that no thresholds for the features are introduced and so the classifier learns the specific patterns of the different classes. Although the human movements are of high variety, sufficient classifiers with only two classes (fall and no fall) can be obtained by the proposed methodology.

Future Work

To improve the fall detection method and reduce the number of classification samples, a preceding simple classifier can be used to investigate the time windows preliminary whether they are potential fall candidates. Therefore the simple MLP or the height alternation can be used to sort out time periods where the MLP yields an output near 0 or the height difference is positive. Time windows which yield a high probability for a fall in the first simple classification can be fed into the HMM or the complex MLP to be further investigated. For the alarm handling a verification method can be introduced which analyses mainly the activity but also the remaining features after a potential fall, to find out if the person is recovering. For further verification, the fall detection can also be combined with statistical behaviour models (see Section 2.3) which learn the individual behaviour patterns of persons in specific rooms to detect abnormalities.

For a more significant evaluation and classification the data base of about 200 scenarios has to be extended to get more balance between the amount of fall and no fall samples. With a bigger database it would also be possible to build up a similar but extended classifier with more than one different class for falls and normal movements. Especially the normal movements contains a variety of different motion patterns. By introducing more classes e.g. for sitting down or bending down, the variety of each class can be minimized and so the pattern of each class can be better recognized. However a high amount of training samples has to be available for each introduced class.

While extending the database of movements, the scenarios can be recorded simultaneously with different 3D sensors like the Kinect [14] or Time of flight cameras to adjust the fall detection method on other types of data. As further improvement of the evaluation during acting the scenarios, wearable devices can be tested to compare their results with the proposed fall detection method.

List of acronyms

AAL	Ambient Assisted Living
AE	Address Event
AIT	Austrian Institute of Technology GmbH
CCD	Charge Coupled Device
CMOS	Complementary Metal Oxide Semiconductor
DCT	Discrete Cosine Transform
FIFO	First In First Out
FPGA	Field Programmable Gate Array
HMM	Hidden Markov Model
MLP	Multilayer Perceptron
PCA	Principal Component Analysis
ROC	Receiver Operating Characteristic
VVDR	Vertical Volume Distribution Ratio

Bibliography

[1] N. Ahmed, T. Natarajan, and K. R. Rao. Discrete cosine transfom. *IEEE Transactions on Computers*, C-23(1):90–93, 1974.

[2] O. Almeida, M. Zhang, and J. Liu. Dynamic fall detection and pace measurement in walking sticks. In *Proceedings of the 2007 Joint Workshop on High Confidence Medical Devices, Software, and Systems and Medical Device Plug-and-Play Interoperability*, pages 204–206, Washington, DC, USA, 2007. IEEE Computer Society.

[3] 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 *2nd Conf. on Information and Communication Technologies, 2006. ICTTA '06.*, volume 1, pages 1003–1007, 2006.

[4] AAL Association. Ambient assisted living joint programme. `http://www.aal-europe.eu/`. access date: 01.11.2011.

[5] E. Auvinet, L. Reveret, A. St-Arnaud, J. Rousseau, and J. Meunier. Fall detection using multiple cameras. In *30th Annual International Conf. of the IEEE in Engineering in Medicine and Biology Society, 2008. EMBS 2008.*, pages 2554–2557, 2008.

[6] M. Hudson Beale, M. T. Hagan, and H. B. Demuth. Neural network toolbox 7 users guide. `http://www.mathworks.de/help/toolbox/nnet/index.html`. access date: 01.11.2011.

[7] A. N. Belbachir. Project care. safe private homes for elderly persons. `http://care-aal.eu/en`. access date: 02.09.2011.

[8] A. N. Belbachir, A. Nowakowska, S. Schraml, G. Wiesmann, and R. Sablatnig. Event-driven feature analysis in a 4d spatiotemporal representation for ambient assisted living. In *The 3rd International Workshop on Video Event Categorization, Tagging and Retrieval for Real-World Applications (VECTaR2011) In Conjunction with ICCV 2011*, pages 1570–1577, nov 2011.

[9] FIZ CHEMIE Berlin. Chemgapedia: Varianz/kovarianzmatrix. `http://www.chemgapedia.de/vsengine/vlu/vsc/de/ch/13/vlu/daten/multivariate_datenanalyse_allg/multivar_datenanalyse_allg.vlu/Page/vsc/de/ch/13/anc/daten/multivar_datenanalyse_allg/varianz_kovarianzmatrix.vscml.html`. access date: 01.11.2011.

[10] A.K. Bourke and G.M. Lyons. A threshold-based fall-detection algorithm using a bi-axial gyroscope sensor. *Medical Engineering Physics*, 30(1):84 – 90, 2008.

[11] P. A. Bromiley, P. Courtney, and N. A. Thacker. Design of a visual system for detecting natural events by the use of an independent visual estimate: A human fall detector. *Empirical evaluation methods in computer vision, Series in machine perception artificial intelligence Volume 50*.

[12] Yie-Tarng Chen, Yu-Ching Lin, and Wen-Hsien Fang. A hybrid human fall detection scheme. In *17th IEEE International Conference on Image Processing (ICIP), 2010*, pages 3485 –3488, sept. 2010.

[13] R. Collins, A. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, and O. Hasegawa. A system for video surveillance and monitoring. Technical Report CMU-RI-TR-00-12, Robotics Institute, Pittsburgh, PA, May 2000.

[14] Microsoft Corporation. kinect for windows sdk beta. `http://research.microsoft.com/en-us/um/redmond/projects/kinectsdk/`. access date: 01.11.2011.

[15] R. Cucchiara, A. Prati, and R. Vezzani. Vezzani: A multi-camera vision system for fall detection and alarm generation. In *Expert Systems, Volume 24 Issue 5*, pages 334–345.

[16] G. Diraco, A. Leone, and P. Siciliano. An active vision system for fall detection and posture recognition in elderly healthcare. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2010*, pages 1536 –1541, march 2010.

[17] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2000.

[18] Sensetec EAZ. Sensetec eaz gmbh. `http://www.senstec.at/uber-uns/`, 2011. access date: 01.11.2011.

[19] Koninklijke Philips Electronics. Philips sense and simplicity. `http://www.lifelinesys.com/content/lifeline-products/auto-alert`. access date: 01.11.2011.

[20] H. Foroughi, B. S. Aski, and H. Pourreza. Intelligent video surveillance for monitoring fall detection of elderly in home environments. In *11th International Conference on Computer and Information Technology, 2008. ICCIT 2008.*, pages 219–224, dec. 2008.

[21] H. Foroughi, A. Naseri, A. Saberi, and H. S. Yazdi. An eigenspace-based approach for human fall detection using integrated time motion image and neural network. In *9th International Conference on Signal Processing, 2008. ICSP 2008.*, pages 1499 –1503, oct. 2008.

[22] H. Foroughi, A. Rezvanian, and A. Paziraee. Robust fall detection using human shape and multi-class support vector machine. In *Sixth Indian Conference on Computer Vision, Graphics Image Processing, 2008. ICVGIP '08.*, pages 413–420, dec. 2008.

[23] Z. Fu, T. Delbruck, P. Lichtsteiner, and E. Culurciello. An address-event fall detector for assisted living applications. *IEEE Transactions on Biomedical Circuits and Systems*, 2(2):88–96, june 2008.

[24] R. K. Ganti, P. Jayachandran, T. F. Abdelzaher, and J. A. Stankovic. Satire: a software architecture for smart attire. In *Proceedings of the 4th international conference on Mobile systems, applications and services*, MobiSys '06, pages 110–123, New York, NY, USA, 2006. ACM.

[25] O. Goddard. Cebm centre for evidence based medicine. `http://www.cebm.net/index.aspx?o=1042`. access date: 01.11.2011.

[26] S. Günter and H. Bunke. Optimizing the number of states, training iterations and gaussians in an hmm-based handwritten word recognizer. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition - Volume 1*, ICDAR '03, pages 472–476, Washington, DC, USA, 2003. IEEE Computer Society.

[27] Tunstall Healthcare. Tunstall all the reassurance you need. `http://www.tunstall.co.uk/products.aspx?PageID=152`. access date: 01.11.2011.

[28] G.C. Holst. *CCD arrays, cameras, and displays*. JCD Pub., 1996.

[29] J. Hu, M. Brown, and W. Turin. Hmm based on-line handwriting recognition. In *IEEE Transactions on pattern analysis and machine intelligence*, volume 18, pages 1039–1045. IEEE CS Press, 1996.

[30] D. Ivan, A. Johansson Augier, V. Lang, and J. Piirto. *Europe in figures Eurostat yearbook 2010*. Eurostat, Statistical office of the European Union, 2010.

[31] B. Jansen and R. Deklerck. Context aware inactivity recognition for visual fall detection. In *Pervasive Health Conference and Workshops, 2006*, pages 1 –4, 29 2006-dec. 1 2006.

[32] I. T. Jolliffe. Principal component analysis. *Applied Optics*, 44(30):6486, 2010.

[33] J. Lember, K. Kuljus, and A. Koloydenko. *Hidden Markov Models, Theory and Applications: Theory of Segmentation*, pages 51–84. Bioinformatics. InTech, 2011.

[34] Q. Li, J. A. Stankovic, M. A. Hanson, A. T. Barth, J. Lach, and G. Zhou. Accurate, fast fall detection using gyroscopes and accelerometer-derived posture information. *International Workshop on Wearable and Implantable Body Sensor Networks*, pages 138–143, 2009.

[35] U. Lindemann, A. Hock, M. Stuber, W. Keck, and C. Becker. Evaluation of a fall detector based on accelerometers: A pilot study. *Medical and Biological Engineering and Computing*, 43:548–551, 2005. 10.1007/BF02351026.

[36] M. Lueder, G. Bieber, and R. Salomon. Air pressure- and acceleration-based fall detector. In *Ambient Assisted Living 2009. 2. Deutscher AAL-Kongress mit Ausstellung. CD-ROM : Technologien - Anwendungen, 27. - 28. Januar 2009 in Berlin, Tagungsbandbeiträge Berlin: VDE Verlag*, Berlin, 2009. VDE Verlag.

[37] M. J. Mathie, J. Basilakis, and B. G. Celler. A system for monitoring posture and physical activity using accelerometers. In *Proceedings of the 23rd Annual International Conference of the IEEE on Engineering in Medicine and Biology Society, 2001.*, volume 4, pages 3654–3657, 2001.

[38] MathWorks. R2011b documentation. `http://www.mathworks.de/help/techdoc/index.html`. access date: 01.11.2011.

[39] DTREG Software For Predictive Modeling and Forecasting. Multilayer perceptron neural networks. `http://www.dtreg.com/mlfn.htm`. access date: 01.11.2011.

[40] Kevin Murphy. Hidden markov model (hmm) toolbox for matlab, distributed under the mit license. `http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html`. access date: 01.11.2011.

[41] H. Nait-Charif and S. J. McKenna. Activity summarisation and fall detection in a supportive home environment. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 4, pages 323–326, aug. 2004.

[42] N. Noury, P. Barralon, G. Virone, P. Boissy, M. Hamel, and P. Rumeau. A smart sensor based on rules and its evaluation in daily routines. In *Proceedings of the 25th Annual International Conference of the IEEE on Engineering in Medicine and Biology Society, 2003.*, volume 4, pages 3286–3289, sept. 2003.

[43] A. Nowakowska. Projektbericht wahlpflicht-projekt: Mathematik und simulation in der biologie 101.388, tu vienna. 2011.

[44] Pan Pantziarka. Techbookreport standard deviation in 30 seconds. `http://www.techbookreport.com/tutorials/stddev-30-secs.html`. access date: 19.11.2011.

[45] C. Posch, D. Matolin, and R. Wohlgenannt. A qvga 143db dynamic range asynchronous address-event pwm dynamic image sensor with lossless pixel-level video compression. In *IEEE International Conference Digest of Technical Papers on Solid-State Circuits, (ISSCC) 2010.*, pages 400–401, 2010.

[46] L. 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.

[47] Barbara Resch. Mixtures of gaussians, a tutorial for the course computational intelligence. `http://www.igi.tugraz.at/lehre/CI`. access date: 01.11.2011.

[48] R. Rojas. The kalman filter. `http://robocup.mi.fu-berlin.de/buch/kalman.pdf`. Projektgruppe RoboCup - Freie Universität Berlin. access date: 01.11.2011.

[49] F. Rosenblatt. *Principles of neurodynamics: perceptrons and the theory of brain mechanisms*. Report (Cornell Aeronautical Laboratory). Spartan Books, 1962.

[50] C. Rougier, J. Meunier, A. St-Arnaud, and J. Rousseau. Monocular 3d head tracking to detect falls of elderly people. In *28th Annual International Conference of the IEEE on Engineering in Medicine and Biology Society, 2006. EMBS '06.*, pages 6384–6387, aug 2006.

[51] A. Sankar. Experiments with a gaussian merging-splitting algorithm for hmm training for speech recognition. In *In Proceedings of the Broadcast News Transcription and Understanding Workshop*, pages 99–104, 1998.

[52] C. E. Shannon. A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5:3–55, January 2001.

[53] Nikolai Shokhirev. Hidden markov models. `http://www.shokhirev.com/nikolai/abc/alg/hmm/hmm.html`. access date: 01.11.2011.

[54] A. Sixsmith, N. Johnson, and R. Whatmore. Pyroelectric ir sensor arrays for fall detection in the older population. *Journal De Physique Iv*, 128:153–160, 2005.

[55] D. M. Skapura. *Building neural networks*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1995.

[56] Skyguard. Skyguard your back your backup. `http://www.skyguardgroup.com/products/Gemshield.aspx`. access date: 01.11.2011.

[57] K. Sokolova and Barak Shilo. Experiments in stereo vision computer science 570, final project. `http://disparity.wikidot.com/`. access date: 01.11.2011.

[58] M. Sokolova, N. Japkowicz, , and S. Szpakowicz. Beyond accuracy, f-score and roc: A family of discriminant measures for performance evaluation. *AI 2006: Advances in Artificial Intelligence*, pages 1015–1021, 2006.

[59] A. Srp and F. Vajda. Possible techniques and issues in fall detection using asynchronous temporal-contrast sensors. *e & i Elektrotechnik und Informationstechnik*, 127:223–229, 2010.

[60] T. Starner and A. Pentland. Real-time american sign language recognition from video using hidden markov models. Technical report, 1995.

[61] H. Steg, H. Strese, C. Loroff, J. Hull, and S. Schmidt. Europe is facing a demographic challenge ambient assisted living offers solutions. Technical report, 2006.

[62] N. Thome and S. Miguet. A hhmm-based approach for robust fall detection. In *9th International Conference on Control, Automation, Robotics and Vision, 2006. ICARCV '06.*, pages 1–8, dec. 2006.

[63] B. U. Toereyin, Y. Dedeoglu, and A. E. Cetin. Hmm based falling person detection using both audio and video. In *IEEE 14th Signal Processing and Communications Applications Conference, 2006*, pages 1–4, april 2006.

[64] Tynetec. Tynetec trusted technology. caring for people. `http://www.tynetec.co.uk/page/home`. access date: 01.11.2011.

[65] F. Wanlass and C. Sah. *Nanowatt logic using field-effect metal-oxide semiconductor triodes*, pages 32–33. Institute of Electrical and Electronics Engineers, 1963.

[66] G. Williams, K. Doughty, K. Cameron, and D. A. Bradley. A smart fall and activity monitor for telecare applications. In *Proceedings of the 20th Annual International Conference of the IEEE on Engineering in Medicine and Biology Society, 1998.*, volume 3, pages 1151 –1154 vol.3, oct-1 nov 1998.

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

[68] Xinguo Yu. Approaches and principles of fall detection for elderly and patient. In *10th International Conference on e-health Networking, Applications and Services, 2008. HealthCom 2008.*, pages 42 –47, july 2008.

[69] A. Zweng, S. Zambanini, and M. Kampel. Introducing a statistical behavior model into camera-based fall detection. In *Proceedings of the 6th international conference on Advances in visual computing - Volume Part I*, ISVC'10, pages 163–172, Berlin, Heidelberg, 2010. Springer-Verlag.