university of groningen / faculty of science and engineering / bernoulli institute

Master Thesis

Egocentric camera-based fall detection system using rotation, motion, HOG and LBP

Author:
Hichem Bouakaz

Supervisors:
Dr. George Azzopardi
Estefanía Talavera Martínez

A thesis submitted in fulfilment of the requirements for the degree of Master

in

Computing Science

October 30, 2020

LIST OF ABBREVIATIONS

ADL	**A**ctivities of **D**aily **L**iving
AUC	**A**rea **U**nder **C**urve
CBS	**C**entraal of **B**ureau voor de **S**tatistiek
EO	**E**dge **O**rientation
ES	**E**dge **S**trength
FPR	**F**alse **P**ositive **R**ate
HOG	**H**istogram of **O**riented **G**radients
HOF	**H**istogram of **O**ptical **F**low
GLBP	**G**radient **L**ocal **B**inary **P**attern
FP	**F**alse **P**ositives
FN	**F**alse **N**egatives
KNN	**K** **N**earest **N**eighbour
LBP	**L**ocal **B**inary **P**attern
PCA	**P**rincipal **C**omponent **A**nalysis
RF	**R**andom **F**orests
ROC	**R**eceiver **O**perating **C**haracteristic
TN	**T**rue **N**egatives
TP	**T**rue **P**ositives
TPR	**T**rue **P**ositive **R**ate
SVM	**S**upport **V**ector **M**achines
WHO	**W**orld **H**ealth **O**rganization

ABSTRACT

Elderly people are the fastest-growing segment of the population in the Netherlands and the world. According to the World Health Organization (WHO), falls - after road accidents- are the second leading cause of unintentional injury deaths worldwide.

This work presents a fall detection system, based on egocentric cameras, to assist the living of the elderly both in indoor and outdoor environments. During this research, a dataset containing 1459 pre-recorded video sequences of falls and non-fall activities was used. The videos were recorded using one camera mounted on the waist and one on the neck. Several methods to detect falls were proposed, namely LBP, HOG, video rotation and camera motion derived from optical flow and Random Forests algorithm is used for classification. Each of the proposed methods was tested and evaluated both separately and together with the other methods. To determine the most suited system for a fall detection which can be employed in real-time to assist the living of the elderly, a comparison is made in terms of reliability and efficiency. The best results in terms of reliability were achieved by combining rotation, LBP and motion with 98.3% mean AUC and 92.6% mean accuracy during cross-validation, 98.8% AUC and 91.8% accuracy on the test set for binary classification, 95.1% micro AUC and 94.7% macro AUC during cross-validation and 95.2% micro AUC and 94.9% macro AUC on the test set. However, considering efficiency LBP is slow, therefore our suggestion is to utilise the model that uses video rotation and camera motion, which is slightly less reliable with 98.1% AUC and 92.4% accuracy during cross-validation and 98.7% AUC and 91.5% accuracy, but considerably faster. The results obtained during this thesis work have shown that the best placement for the camera is the waist. The outcome of the work is promising, not just for fall detection in particular but also for human action recognition in general.

ACKNOWLEDGMENTS

First of all, I would like to thank Dr George Azzopardi for providing excellent feedback and guidance throughout this research. His feedback was always steering me in the right direction, while still allowing enough freedom to develop my scientific skills.

Furthermore, I also like to thank Xueyi Wang for his valuable guidance and feedback. His insight and strong dedication to his work helped me to understand quickly and efficiently in what way I could contribute to the subject of fall detection, which turned out to be an interesting field to explore.

I also like to thank Mark Helmus for our discussions and the exchange of ideas, which allowed me to deepen my knowledge of the subject. Many thanks go out to the volunteer who managed to fall more times than we can count and with her contribution helping us to test our theories.

Finally, I thank my family for giving me space and support to finish my master.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 INTRODUCTION

There is an increase in the population of the elderly in the Netherlands and the world due to advancement in medical care and other factors. Therefore, it is important to pay more attention to the problems facing this category of the population. One of the biggest dangers among the elderly population is falling, and it is one of the major causes of morbidity and mortality. It is considered as the second cause of unintentional deaths after road traffic accidents. Most falls happen in medical health care centres, hospitals and houses, with approximately 30% of falls causing injury which can suddenly degrade the quality of life and the general wellbeing of elderly people. Most people who experience falls need special care in a rehabilitation centre, at home or in the hospital.

The World Health Organisation [1] has defined fall as an event which results in a person coming to rest unintentionally on the ground or floor or other lower level. Fall-related injuries may be fatal or non-fatal.

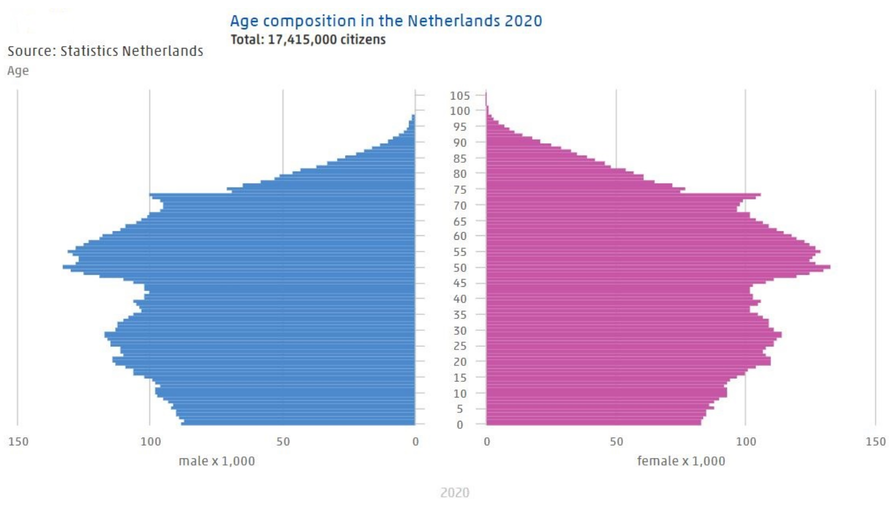


Figure 1.1: Age composition of the Netherlands in 2020, taken from CBS [2]

The current population of the elderly in the Netherlands above 65 years is 3.3 million according to the CBS (Centraal Bureau voor de Statistiek) [3]. Of the whole Dutch population, 19,2% were above the age of 65 in 2019. The number of people that suffer from falling every year is over 15 thousand. Falls cause major injuries, the most common injuries are hip fractures (15%), mild brain damage (11%) and wrist

(9%). More than three-quarters of them are 75 years or older. After falling 40% of elderly cannot return to their homes and need to go to an elderly care facility. Only one-quarter of elderly are open to take advice on how to take measurements to prevent falling. [4]

VeiligheidNL, an organisation specialised in monitoring and reducing the number of accidents in the Netherlands, does research on a regular basis concerning fall prevention within elderly people. In the most recent report from 2018, it is mentioned that every 5 minutes an elderly end up in the emergency room after a fall, which amounts to 108 thousands of treatments a year. Of these cases, 74.900 patients require additional diagnoses due to i.e. fractures and brain injury. The death-rate caused by falling among the elderly is 12 people every day, this amounts to 4.396 deaths on a yearly basis. Besides this, it is estimated that there has been an increase of 6% of serious injury between 2009 and 2018.

In view of the ageing population, these numbers will only increase in the coming years. The prognosis is that the number of first-aid visits after a fall accident will increase by 47% until 2050. The injury sustained by a fall accident has a major impact on the self-reliance of the elderly, the ability to live at home longer, and the quality of life [5].

The Dutch ministry of health, welfare and Sport has recently issued a new ‘National Memorandum on Health Policy 2020-2024’ which has the ambition to do more on local level to prevent falling. In this Memorandum, the central government encourages municipalities to find new ways to ensure less elderly people ending up in the ER [6].

Falls among the elderly also have an economic impact. In 2017 the medical cost for treating elderly patients in the ER or admittance in the hospital due to falling was €837 million. Most of these costs were caused by falling of the elderly above 75 years (85%, €749 million) [6]. A reduction of fall incidents with 5% will save the health care sector in The Netherlands around 40 million a year. On top of this, there is a program, approved by RIVM (Rijksinstituut voor Volksgezondheid en Milieu -Centre for Healthy living), that has already been set in a place called ‘In Balans’. This course aims to teach elderly people the skills they need to prevent themselves from falling. Among elderly who participated in this training, there has been found that accidents due to falling have decreased by 61% and the fear of falling has been reduced by 37% [7].

Most elderly people spend a lot of time at home, however, there are studies in which researchers have found that more than 50% of falls occur when elderly interact outside their homes. The health condition difference between elderly is that the ones which fall in the home, are more fragile and might already suffer from underlying conditions. The elderly that fall outside the homes are often very active. Most studies combine the numbers and do not differentiate between falling inside and outside of the home environment. [8]

1.2 RESEARCH CHALLENGES

Fall detection is a very important and popular research topic. However, there are many challenges when it comes to developing a fall detection system. One of the main challenges is that there are many human actions in the daily life that are very similar to falling, which results in many false positives in the existing systems. Thus, the challenges are:

Reliability: the main challenge is to build an appropriate system that is not only able to detect true positive falls, but also able to correctly separate between the non falls similar to falls and the actual falls which will reduce the number of false alarms. This system will be using only videos taken from wearable egocentric cameras.

Efficiency: for the fall detection system to usable in real time it has to be effective, which means finding an efficient algorithm that gives the ability to detect falls in real time is very important.

Elderly privacy is another challenge specially with vision-based fall detection system. However, using egocentric cameras mitigate the privacy issue, because as opposed to ambient cameras that watch the subject himself.

Lack of public datasets which makes it difficult to compare our work with previous works since papers use their own simulated falls dataset, The size of the collected dataset is also small, and has limited variation in the types of activities of daily life.

1.3 AIMS AND OBJECTIVES

Our research is mainly focused on developing an autonomous system which detects human falls through egocentric camera video input data. In this research, it is important that we develop an efficient algorithm that can be implemented in real time, while at the same time it can distinguish between falls and other ADL activities.

To be able to achieve our objective we will:

- Conduct extensive research in the literature to get a general view on existing techniques used for fall detection in particular and

video action classification in general.
Over the last years there have been many researches conducted in the area of fall detection. Our goal is to first research and compare existing methods used for fall detection, identify their strengths and weaknesses, understand their limitations, and compare the results they obtained to get informed about the state-of-art techniques used for fall detection.

- Pre-process the data obtained by volunteers
- Implement different methods for fall detection that can advance the state-of-art limitations. This is done by using traditional methods.

1.4 SCOPE AND MAJOR CONTRIBUTIONS OF THE THESIS

The scope of this study consists of designing a fall detection algorithm which employs different methods namely local binary patterns (LBP), histogram of oriented gradients (HOG), camera motion derived from Gunnar Farneback optical flow, and video rotation obtained from Lucas Kanade optical flow. Since there are no available public datasets, we use the dataset we collected. After implementing the aforementioned methods, we evaluate the performance of each one on fall detection in terms of reliability and efficiency to determine the best single method or combination of methods for fall detection, in terms of reliability and efficiency. Furthermore, we also conduct a non-binary classification to evaluate the performance of the chosen methods on different activities of daily living (ADLs). Finally, we compare the performance of each camera separately i.e., neck mounted camera versus waist mounted-camera to find the ideal mounting point of the camera. This study does not provide a solution to the privacy issue mentioned in Section 1.2.

1.5 THESIS OUTLINE:

This thesis consists of the chapters that are summarised as follows:
Chapter 2: Background and Literature Review. This chapter gives a review of the relevant literature in the field of Assisted Living (AL) technologies which are used in human activity recognition and fall detection domains. These technologies can be classified into wearable sensor-based methods, ambient sensor-based method and vision-based methods. In particular, the literature review focus on the use of threshold-based methods and machine learning methods for fall detection of older adults to support them to live independently in their own homes.

Chapter 3: Methodology. This chapter goes in details describing the

Dataset used in this project, then gives a detailed view about the theory behind the used methods namely local binary patterns, histogram of oriented gradients, camera motion obtained from Gunnar Farneback optical flow method, and finally rotation obtained from video trajectories using the pyramidal Lucas-Kanade method. Finally, we describe the Random Forest algorithm.

Chapter 4: Experiments and results, The chapter contains our conducted experiments, and presents the results obtained from each experiment, and concludes with a comparison between the methods derived from the obtained results to determine which methods are best suited for fall detection.

Chapter 5: Conclusion. This chapter summarises the research as a whole, restating the problem definition and the answer to the research question, challenge and limitation of the research, and suggestion for future improvement.

2 BACKGROUND AND LITERATURE REVIEW

2.1 INTRODUCTION

This chapter is structured as follows: first we explore the different types of falls and the major risks of falling, and we emphasise the need for a fall detection system that can monitor the elderly both indoors and outdoors in Section 2.2. Secondly a general review of fall detection literature is given in Section 2.3. Thirdly we investigate previous researches in the fall detection domain and highlight the techniques and algorithms used to solve the problem of fall detection. Finally, we explore the problems they have encountered and how they manage to overcome those problems, and we give an overview about the state-of-art classification methods used for detecting and classifying human actions.

2.2 BACKGROUND

To fully understand the problem of falling among the elderly, we study different fall types, the factors surrounding it and the risks which result in falling.

2.2.1 *Fall types:*

The most common fall types depending on orientation, amplitude and initial position are: [9]

2.2.2 *Fall types according to the orientation of the fall:*

- **Frontal fall:** An elderly falls forwards in which case the face hits the surface.
- **Backward fall:** An elderly falls backwards; in most cases the back of the head hits the surface.
- **Side fall:** An elderly falls sideways either left or right.

2.2.3 *Fall types according to the amplitude*

- **Fast fall:** An elderly falls fast, causing a high amplitude of the body movement; this type of fall lasts from one to two seconds.
- **Slow fall:** An elderly falls slowly, the amplitude of the body movement is comparatively small and the duration is comparatively long.

2.2.4 *Fall types based on the initial position*

- **Fall from standing:** This type of fall starts from a standing or walking posture, it occurs when an elderly slip or get unconscious. Both the head and centre of gravity move towards one direction and their height reduce (normally to the plane of the ground). Typically, this type of fall is considered a fast fall with large movement amplitude.
- **Fall from sitting:** This type of fall starts sitting position, it occurs when an elderly person slips from a chair due to unconsciousness. Similar to the fall from standing, the head and centre of gravity move towards one direction with reduced height. Compared with the fall from standing, this type of fall has smaller movement amplitude.
- **Fall from lying:** This type of fall starts from a lying position. This type of falls means that an elderly person rolls to the floor from the bed during sleep. The person is initially on the bed when a fall happens and the body reduces its height from the bed to the floor plane, with the final body position being near the bed. This type of fall usually happens when an elderly person sleeps and his/her body rolls out of the bed while the person remains unconscious.
- **Falling from other positions:** A person falls from an initial bending or crouching or other posture. This type of fall happens for example when the elderly tie their shoelaces or become suddenly unconscious while doing other activities.

2.2.5 *Risk factors of falling*

The main risk factors of falling among the elderly are divided into three categories:

2.2.5.1 *Intrinsic factors:*

The intrinsic factors are related to individual strength and performance levels like age, muscle weakness, heart conditions etc. and age-related medical conditions like Parkinson's, Alzheimer's, sudden blood pressure drops, chronic and acute disorders. The risk of falling also increases with the number of drugs taken. [10] [11]

2.2.5.2 *Extrinsic factors:*

The extrinsic factors are environmental factors. Extrinsic actors can be improved to lower fall incidents. Fall risk becomes higher when the elderly needs greater postural control, this includes poor lightening, slippery floors etc. [10] [11].

2.2.5.3 *Situational factors:*

Such factors refer to activities and situations that increase the chances of falling such as going to the bathroom at night, rushing to open the door, multitasking etc. [10]

Types of fall	**Indoors**		**Outdoors**	
	Falls	Fractures	Falls	Fractures
Slipping	94	12	428	27
Tripping	279	23	203	5
Other extrinsic	89	10	110	4
Intrinsic	762	39	207	12
On stairs	38	6	41	5
From an upper level	248	12	31	3
Non defined	212	18	20	2
Total	**1722**	**120**	**1040**	**58**

Table 2.1: Falls and fractures according to place and type of fall in home dwelling elderly. [12]

The study made by Luukinen et al. [12] in which they have recorded all falls and non-falls related fractures of 980 home dwelling elderly people (70 years and above) over 7 years, has shown that extrinsic causes are falls such as tripping, slipping and falling on stairs, falls occurring from an upper level are stronger than intrinsic falls, which expose the subject to fracture. Another finding made by the study as shown in Table 2.1 that intrinsic factors related falls are more common indoors than outdoors.

2.3 LITERATURE REVIEW

This section gives a review of the existing literature, we discuss state-of-art systems used for human action recognition in general and fall detection in particular, and we provide an overview of the most popular classification methods used in fall detection.

2.3.1 *Technologies used in fall detection*

We can divide the fall detection systems into three major categories: 1) environmental sensing-based systems, 2) wearable sensor-based systems, 3) vision-based systems.

2.3.1.1 *Environmental sensing-based systems*

Environmental sensing systems, also called ambient devices, use event sensing through the examination of the environment to monitor the elderly person's movement. The sensors are usually external and attached to the environment. Ambient devices have an advantage compared to other devices as they are not intrusive and maintain the privacy of elderly people. However, they have many disadvantages including coverage; they can only work indoors and suffer from blind spots and they are affected by the environment such as background noise and ambient noise. Ambient sensors can be divided according to the sensor type: acoustic sensor, vibration sensors and pressure sensors. [13]

Taramasco et al. [14] proposed a non-intrusive fall detection system for controlled environments. Their system measures the ambient temperature through low-resolution thermal sensors. The sensors detect the body-heat of the subject of interest without having to be worn. The sensors can detect the body-heat from a maximum distance of 4 meters.

Wang et al. [15] use existing wireless infrastructure for detecting falls by using the channel state information (CSI) in a given area from the WiFi deployments. The major advantage of this system is its non-intrusiveness since it does not expect the subject of interest to be wearing or carrying any device. However, it has the disadvantage of having false positives because of the impact of other ambient parameters like heat.

Litvak et al. [16] combined microphone measurements with the extracted features from the floor accelerometer.

Device-free fall detection has been the focus of many studies. However, it has many shortcomings, such as the need to install a lot of sensors on the wall or house, and it is limited to indoors. [17]

2.3.1.2 *Wearable sensor-based systems*

Wearable devices contain a wide range of sensors such as accelerometers and gyroscopes. Such sensors are employed to capture human movement and detect potential falls. In this sub-section, we will discuss the literature of systems that are based on such wearable sensors.

In their paper Abbate et al. [18] proposed a system that uses accelerometers found in smartphones, where they measure the orientation based on the accelerometer x- and y-axis. They have demonstrated that the recognition of fall-like activities of activity daily living (ADL) on the daily basis can significantly reduce the number of false fall alarms.

Crispim-Junior et al. [19] suggested combining a video camera with an accelerometer device for fall detection. the system combines the elderly's acceleration with visual information. Compared to the same system using just video data combining the video data with the accelerometer data has improved the event detection performance of fall detection.

Shi et al. [20] developed a system based on inertial Micro electromechanical systems (MEMS) which can detect falls in real-time. This system can be used as a wearable device and it contains three decision algorithms, in which J48 decision tree algorithm are used. Through training and evaluating the classifier, the results showed that the best location for detecting the fall events was the waist where the sensitivity was 95.5%, the specificity 98.8% and the overall accuracy 97.792%. However, combining the waist and feet to perform the classification gives better performance.

Lee et al. [21] applied a two thresholding method to analyse the data acquired from a smartphone and an accelerometer, to identify the movements and the different simulated falls. The technique of thresholding has some limitations; mainly because it generates many false alarms since it is not able to accurately distinguish between the fall and ADL movements.

2.3.1.3 *Vision-based systems*

The computer vision-based methods use camera as input. The cameras can be either from a third-person perspective or from an egocentric perspective. The input can be either from a single camera or multiple cameras, or using depth cameras like Microsoft Kinect cameras. [22] In this section we discuss techniques used for video classification in general and fall detection in particular. These techniques are commonly used on video action recognition. The third-person perspective cameras focus on the body shape to detect falls, while the egocentric cameras focus on the change in scenery to detect falls.

Vision-based action recognition uses many techniques to extract important features from the raw videos. We can summarise the most popular techniques into three main categories:

2.3.1.3.1 Human body model based methods

Action recognition is based on the extraction of 2D or 3D information on human body parts, such as body part configuration, body part positions, and movements.

The idea of recognising Human action using the body motion goes back to the experiment of Gunnar Johansson [23], his experiment shows that humans can recognise actions merely from the motion of a few moving light displays (MLDs) attached to the subject's body. The collection of MLDs spots carry only 2D information and no structural information. As they are not connected to each other, their relative movement creates an impression of the person activity walking.

Chen et al. [24] proposed an approach for recognising falls based on the symmetry principle. Their system first extracts the skeleton information of the human body using OpenPose algorithm. Then it identifies the fall through three critical parameters: first the speed of descent at the centre of the hip joint, second the centre-line angle of the body with the ground and third the width-to-height ratio of the human body external rectangular.

2.3.1.3.2 Holistic methods

Human action recognition needs the extraction of information on people localisation in videos, and a global representation of human body structure, shape and movements is used for action recognition. Holistic techniques do not use information on human body parts. Holistic approaches can be divided into two main categories; 1) Based on shape masks or silhouette information, using background subtraction or difference images, to represent actions.2) Based on shape and optical flow information.

Weinland et al. [25] introduced a time-invariant representation for action recognition by using a set of silhouette exemplar-based embedding, The embedding represents a sequence via its minimum distances to a set of prototypes, and the action sequences are represented as vectors of minimum distance between silhouettes. The classification is done using Naive Bayes classifier with Gaussians to model action classes.

Abobakr et al. [26] proposed a vision-based skeleton-free fall detection system using Kinect-like sensor, the input depth frames goes through several processing modules. First, foreground segmentation is performed to subtract the background. After an RDF which is trained and evaluated using synthetic datasets is used to identify the current articulated posture in the frame. Finally, SVM is used to detect falls.

Holistic approaches have proven to be suitable for action recognition. However, holistic representations are in general not invariant to the camera point of view, to take different camera angles into account larger amount of training data will be needed.

2.3.1.3.3 Local feature methods

Action recognition based on local features is one of the most active research topics. Local features based methods have an advantage since no information on human body model or detection of people is required. We will focus in this section on the main local features methods. The pipeline of local features methods has the following steps: 1) interest point detection 2) local descriptor extraction 3) aggregation of local descriptors. [27]

2.3.1.3.4 Local feature detectors

We can divide local feature detectors into two categories: spatio-temporal interest point detectors and trajectory detectors.

2.3.1.3.5 Spatio temporal (detector) interest points

Proposed by Laptev et al. [28] it was developed by extending the notion of spatial interest points into spatio-temporal domain, built on the idea of Harris and Forstner interest points operators through detecting significant local variations in both space and time. The Harris 3D interest points are detected as local positive spatio-temporal maxima. Moreover, the detected points have to be normalised using spatio-temporal Laplace operator.

Dollar et al. [29] proposed a new spatio-temporal interest point detector. Their detector showed how the use of cuboid prototypes paved the way to an efficient and robust behaviour descriptor.

Instead of detecting interest points over the entire volume, Wong et al. [30] used a new method for extracting spatio-temporal features using global information. Their method is based on the extraction of dynamic textures. The extracted locations are sparse and detect salient motion patterns, only a sparse set of features is needed for action recognition. However, all input videos need to be pre-processed into samples containing one iteration of the action each.

Willems et al. [31] introduced Hessian3D interest point detector as spatio-temporal extension of the Hessian saliency measure using the determinant of the 3D Hessian matrix. They have combined point localisation and scale-selection in a direct way. The authors aimed to use a dense and scale-invariant interest point detector.

2.3.1.3.6 Trajectory detectors:

Trajectory detectors employ the motion information of trajectories. It is obtained by either tracking techniques based on the KLT tracker (Lucas and Kanade 1981) or using a SIFT descriptor between consecutive frames, or by combining both approaches. [32]

Matikainen et al. [33] introduced spatio-temporal interest points that are both scale-invariant spatially and temporally, and densely cover the video content which allows for efficient computing of features, applying scale-space theory using the determinant of the Hessian as the saliency measure. Computations are made faster through the use of approximative box-filter operations on an integral video structure.

Messing et al.[34] proposed a feature based on velocity history of tracked keypoints. Using a generative mixture model (GMM) for video sequences. Using this feature, the proposed feature performs comparably to local spatio-temporal features on the KTH activity recognition dataset.

Another way to extract trajectories is to use Dense Trajectories proposed by [35] which is an approach to model videos by combining dense sampling with feature tracking. This approach is more robust than other descriptors as its ability to capture the motion information in the videos efficiently. It also helps in removing camera motion by computing the motion boundaries descriptors along the dense trajectories.

2.3.1.3.7 Bag of features

The bag-of-features is a popular representation based on local features, it was mostly used in Natural Language Processing and was originally proposed for document retrieval, where text is represented as a bag-of-words, it encodes global statistics of local features through computing a spatial histogram of local feature that occurs in a video sequence. [36] However recent approaches try to retain more information about the local features either by having features represented as a combination of visual words or by representing the differences between features as visual words.

Fisher vector encoding [37], [38]. The Fisher Kernel is a generic framework which uses generative and discriminative approaches, it extends the bag of visual-words beyond count statistics [39]. In the context of image classification, Fisher Encoding represents differences between features and visual words; this is done by creating a vocabulary through clustering local features. The clustering is done using Gaussian Mixture Model and after this, it captures the first and second-order differences between local features and visual vocabulary. [36]

A variant of Fisher encoding is Super-vector encoding introduced by Zhou et al. [40]. Super-vector encoding uses local visual descriptors. The pipeline has three steps: First, perform a nonlinear feature transformation on descriptors, secondly aggregate the results together to form image-level representations and finally apply a classification model. There are two variants of the support vector encoding when it comes to assigning the local features to the visual words; first assignment of local features to the nearest visual word and second soft assignment of local features to several nearest visual words. [41]

Rodriguez et al. [42] propose an approach using a template-based method or recognising human actions through flow features called Maximum Average Correlation Height (MARCH). In which filter spatio-temporal regularity flow information is used as feature type. They claim that regularity flows is an improvement over optical flow, because it globally minimises the overall sum of gradients in the sequence. The cuboid templates are learned through aligning training samples using correlation. Then test sequences get correlated with the learned template via generalized Fourier transform that allows for vectorial values. Results are demonstrated on the KTH dataset, for facial expressions, as well as on custom movie and sports actions.

Shieh et al. [43] have proposed a human-shape based fall detection algorithm from multiple cameras, the algorithm uses different angles to fetch the images then uses a falling pattern-recognition based on the body posture to determine if it was a fall or not, their multi-threaded pipeline contains: image fetching, image processing, human-shape generation and pattern recognition, they have reached the precision rate of 92.3% on their dataset.

A powerful tool for texture description is Local Binary Pattern (LBP) operator [44]. LBP operator has powerful discrimination while being computationally simple. LBP works originally using 3x3 pixel blocks. However, it can be extended to neighbour pixels hood with different sizes. [44]

Mattivi and Shao. [45] have proposed using Local Binary Pattern on Three Orthogonal Planes, where each video is described as a collection of spatial-temporal words, their method has shown that (LBP-TOP) is a promising descriptor for human action recognition.

Similarly, Bulbul et al. [46] system uses a feature extracting scheme which they called for the real-time human recognition first by forming three Depth Motion Maps (DMMs) from the depth video, After they calculate the local binary patterns within the overlapping blocks to capture the texture information, third they use Edge orientation histograms on the non-overlapping blocks.

Suad et al. [47] used an efficient approach for fall detection. Their approach is based on a combination of timed motion and change in human shape, which offers crucial information about the activity of the person in the video. By implementing timed motion history image (tMHI) which is a technique to extract motion from a video sequence.

2.3.1.3.8 Egocentric cameras systems

There are few vision-based methods that utilised egocentric cameras in which the subject of interest wears a camera around a chosen part of his body. The benefit of such methods that unlike fixed cameras methods where the fall or activity detection is limited to the field of view of camera, egocentric cameras tracks the activity of the subject of interest everywhere. In addition, the egocentric camera does not film the subject which provides more privacy.

Mauricio et al. [48] were the first to have built a system for detecting falls by a wearable embedded smart camera, their system could work with limited memory and processing power. The detection method uses modified HOG where they build separate histograms for gradient orientations and gradient strengths, and to detect falls they use a manual threshold.

Ozcan et al. [49] introduced a threshold-based system using egocentric cameras that is mounted on the waist. Besides detecting falls, their system is able to detect other activities such as laying down and sitting. In their system, they make use of a modified version of the histogram of oriented gradients algorithm that was first proposed by [50]. The main idea behind their algorithm is based on the observation that edge orientations in a frame vary drastically and extremely fast during falls, which makes subsequent frames blurry. They calculated dissimilarity distance between consecutive frames and if the dissimilarity is greater than a predefined threshold a fall is detected.

In a later study Ozcan et al. [51] proposed a new way of detecting falls through two stages; first detecting the event, then decide whether the detected event is a fall or not. This is done by introducing Gradient local binary patterns GLBP, compared to HOG that uses only 4 neighbours, GLBP uses eight neighbours, another benefit of using GLBP that it uses only uniform. The authors also make use of Edge orientation. and Edge strength. Their obtained result showed the superiority of their method to HOG and GLBP based methods.

Another egocentric camera based fall detection is the work of Boudouane et al. [52] their system employ the original HOG version combined with the optical flow by calculating $\Delta X/\Delta Y$ which is derived from the average flow of pixels along the x-axis and the y-axis to improve the performance. The fall is detected if both the dissimilarity distance of two consecutive HOG histograms together with ratio $(\Delta Y/\Delta X)$ remains above a certain threshold.

2.3.2 *Classification methods:*

2.3.2.1 *Threshold-based methods:*

One of the simple methods to classify falls and videos in general is using a threshold, threshold-based fall detection systems are widely used and are typically designed to minimise computational overhead. Threshold-based systems accuracy, however, may heavily depend on the placement of the sensors or camera [53]. Many studies applied threshold to detect falls. Bashir et al. [54] proposed a simple threshold-based fall detection system that uses wireless body area network. Their system can distinguish between three activities; fall, ADL, and sleep through calculating the posture angle, angular velocity, and acceleration to determine if a fall has occurred or not.

Anania et al. [55] proposed a threshold-based fall detection algorithm based on 3D acceleration. A Kalman filter was used to separate the signal component due to gravity from acceleration data, and then the trunk inclination angle was computed. The system uses predefined two thresholds; one for the subject's tilt angle and the second for the rate of change of tilt angle. If the tilt angle is greater than the first threshold and the change in the tilt angle over a short period is greater than the second threshold a fall is detected, since the thresholds are set manually the algorithm does not generalise well for unseen subjects. Another drawback of the system that it only two postures are considered as corresponding to falls.

2.3.2.2 *Machine learning based methods:*

2.3.2.2.1 Support Vector Machines:

Support Vector Machines (SVMs) is a supervised classier derived from statistical learning theory and was introduced by Vapnik [56]. SVMs need both positive and negative training set, the sets are needed to seek for the decision surface that best separates the positive from the negative data by finding the linear separation which maximises the margins between the positive and negative data in the n-dimensional space (hyperplane).

SVMs advantages can be summarized into the following: First, it is based on a well-established theory. Second, it desires only tens of training specimens, and it performs well non-linear relationship between the input and output features exists, [57]. However, SVMs are time-consuming and use high memory compared to other classifiers such as logistic regression because of more parameters, demands more computation.

2.3.2.2.2 Random Forests

Random forests (RF) were introduced by Breiman in 2001 [58]. RF make use of an ensemble of classifiers such as binary decision trees (DT) which are learned in a supervised way through randomly picking several descriptor dimensions at each node, choosing the node with the highest entropy gain. RF has low bias and variance performances. which makes make use of DT advantages but at the same time, avoiding their disadvantages such as their sensitivity for noise. RF does not have very high computational cost and work with limited resources. RF are generally popular for solving classification and regression problems. RF have proved to be very accurate but at the same time simple and fast, when compared to other machine learning techniques [59] [60] [61].

In their paper Kim et al. [61] used two different types of features extracted from 3-axis accelerometer and depth sensors, To efficiently detect falls they integrated the sensor information in the framework of random forest classifier to detect falls. the input features in the training phase were randomly selected repeatedly to learn each decision tree and multiple (200) decision trees were combined together, each tree had 25 depth.

2.3.2.2.3 K nearest neighbours

k Nearest Neighbor classification (kNN) is a supervised machine learning algorithm which can be seen as a direct classification method. It is based on localising a group of k objects in a training case that has the closest proximity (distance) to the new test object we want to classify. After, the new observation will be assigned to the most common class through a majority vote of its k nearest neighbours. [62] [63]

KNN was first used by Foerster et al. to classify 9 different human activities by using time-domain features obtained from three uni-axial accelerometers. Foerster et al. have combined kNN with a hierarchical decision approach. It has shown to be more efficient, in terms of classification accuracy, it also has demonstrated a high level of accuracy and satisfactory segmentation results [64].

Sani et al. [65] presented an analysis of different feature representation approaches for recognising human activity using kNN. they have used three categories of feature representations, namely: handcrafted, frequency transform and deep features. Their evaluation shows kNN to be very effective at using deep features, even when a minimum amount of time spent in training these deep features.

2.3.2.2.4 Artificial Neural Networks

Artificial Neural Networks (ANNs) emerged as a powerful technique for modeling general relationships. They have been used in many areas and have proven to be effective in classification problems [66]. ANNs consist of three types of layers: 1) input layer with input units which receive information to be processed, 2) output layer with output units which give the result of the algorithm, 3) hidden layers with hidden units which process the data.

Harrou et al. [67] proposed a statistical approach to detect human falls based on both video data and acceleration data. Video data were collected via camera and accelerometer named X-IMU inertial sensor proposed a statistical approach to detect human falls based on both video data and acceleration data. A Shewhart control chart is used to detect a fall by using the accelerometer data. Features are extracted from images which contain a human body silhouette. The silhouette is divided into five areas. The set of ratios that are computed for each frame are then computed to form the feature vector. The features are used as input data to Neural Network. The experiments were conducted on UR Fall Detection dataset. Their system achieves an accuracy of 96.67%, sensitivity of 100% and specificity of 93.4%.

2.3.2.2.5 Deep learning Methods

The recent advancements in deep learning have changed the landscape of computer vision, it has improved the results in many tasks, such as object recognition, segmentation, and image captioning. [68] The most popular neural network architectures are CNNs. CNNs are nets inspired by human visual perception and mainly applied for image processing.[69]

Kong et al. [70] used three-stream CNN as an event classifier. where they used the Silhouettes motion history images as input for the first two streams, and dynamic images are used as input for the third stream. The final classification uses voting on the results of event classification to perform multi-camera fall detection.

Espinosa et al. [71] proposed a fall detection system based on a 2D CNN inference method and multiple cameras. Their approach analyses images in fixed time windows and extracts features using an optical flow method that obtains information on the relative motion between two consecutive images.

Martinez et al. [69] proposed a multi-modal fall detection system based on wearable sensors, and combined long short-term memory networks (LSTMs) and convolutional neural networks (CNN). The authors concluded that they were able to extract features from raw data, and the suggested method is suited for real-time detection. The results represented an improvement in precision, recall and F1 score over using only LSTM or CNN networks for fall detection.

3 METHODOLOGY

In this chapter, we describe the dataset in Section 3.1 and we discuss the methods of designing and implementing the fall detection system. In Section Section 3.1, we give an overview of the dataset; we give its size and discuss the way it was processed. In Section 3.2, we discuss in details the algorithms that are used for feature extraction from the videos in Section 3.2, then we finally discuss the classification method used to detect falls in Section 3.3.

3.1 DATASET DESCRIPTION

The Dataset is obtained from a volunteer who wears two cameras as shown in Figure 3.1 :
A camera fixed on the neck; the volunteer wears this camera on her neck, and another camera is fixed around the volunteer's waist. The videos collected from the neck camera are expected to have more movement than the videos collected from the waist camera since anatomically the waist is a more stable part of the body.

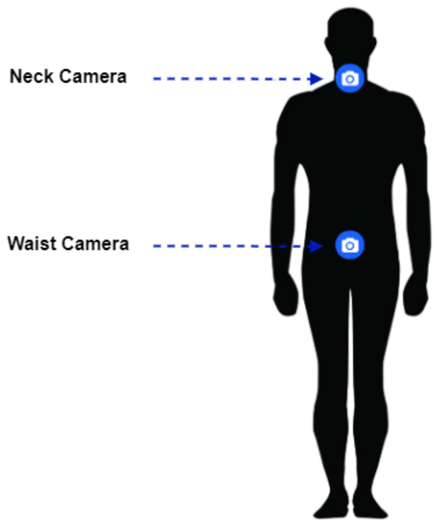


Figure 3.1: Camera placement

After obtaining the videos from the volunteer, a segmentation for the videos is needed since one video contains multiple activities. To ensure the quality of the dataset, we have decided to segment the videos manually by first splitting the video into several video segments. Each video segment contains a single daily activity. Afterwards, we needed to trim the videos to shorten the duration and to remove segments when there is a high luminosity or when there is a hand in front of the camera. We then stored the videos in a database containing the following information: file path, file name, camera, place, number of frames, action type, action. In the end, we have collected 1459 videos. The daily activities collected are divided into the following:

3.1.0.1 *types of Falls:*

- **Back left** Back let falls happen when the subject falls towards the back and ends up on the left side of the body. This fall action takes from 2 to 4 seconds.
- **Back right** similar to back left falls but the end of fall happens on the right side.
- **Back lying** This action happens when the subject falls towards the back and ends up lying on his/her back. this type of action takes from 2 to 4 seconds.
- **Down syncope** The Down Syncope falls are falls that happen from standing. This type of fall action takes from 3 to 5 seconds.
- **Down Syncope Wall** Those falls happen when a person falls from standing, but he or she touches the wall while falling. This fall action takes from 3 to 5 seconds.
- **Front falls** This action happens when the subject falls from standing posture directly towards the floor. this action takes from 3 to 5 seconds.
- **Front knees falls** This type of falls is when the subject falls forward on his knees to the floor.
- **Front left** This type of falls happens when the subject falls on the left side from a standing position. The action takes from 3 to 5 seconds.
- **Front right** this action is similar to 'Front left falls' the only difference is that the subject falls on the right side.
- **Lateral Right side** this type of falls happens when the subject falls on the right side from a standing posture, but there is more movement involved in this type of fall. It takes from 2 to 3 seconds.
- **Lateral left side** Similar to 'Lateral Right side falls' with the only difference that the fall happens towards the left side.

We have also collected Non-falls that are similar to falls to avoid false positives, in case the fall detection system is applied in real life.

3.1.0.2 *Types of Non-Falls:*

- **Bending** This type of action is when the subject bends his knees and looks towards the floor, usually to pick up something. The action takes from 2 to 3 seconds.
- **Lying in bed** This action is when a person goes from standing to lying in bed, these actions are very similar to falls. The action takes from 3 to 5 seconds.
- **Rising from bed:** This action is when a person goes from the position of lying in bed to the position of standing up. This type of actions takes from 3 to 6 seconds.
- **Limping** Limping is defined as the action of walking with difficulty, typically because of a damaged or stiff leg or foot. This action can be continuous.
- **Sitting:** This action is when the subject goes from standing posture to sitting posture. This type of actions takes between 1 and 3 seconds.
- **Sitting on chair:** This action happens when the person goes from the standing position to a sitting position. The action takes from 1 to 3 seconds.
- **Squatting down:** this happens when the subject lowers himself to the ground while balancing on his feet with the legs bent. This action takes between 2 and 4 seconds.
- **Stumble:** This action happens when the subject trips or momentarily loses balance when walking and tripping over something. This action takes between 1 and 2 seconds.
- **Walking:** walking is moving at a regular pace by lifting and setting down each foot in turn. This action is continuous.

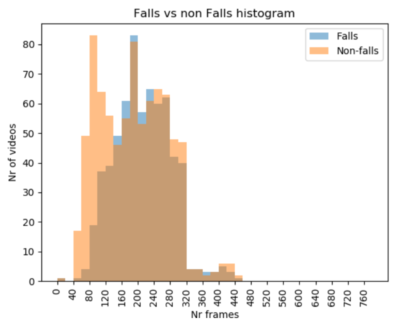


Figure 3.2: Histogram of the number of frames per video of falls and non-falls videos

While segmenting the videos, we also made sure that the number of frames is equally distributed between falls and non-falls as shown in Figure 3.2.

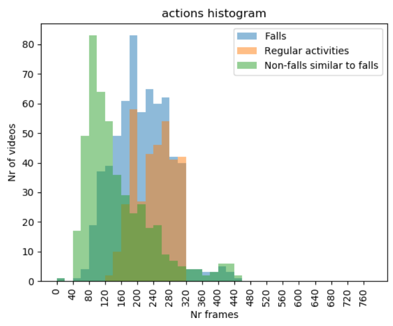


Figure 3.3: Histogram of the number of frames per video by each action type

We also ensured that the number of frames is equally distributed for the action types i.e.: falls, non-falls similar to falls, and regular activities as shown in Figure 3.2. The number of frames range from short videos ≥ 40 frames to long videos ≤ 460 frames. The frame rate of the videos is 30 frames per second.

Action	**Action type**	**Number of files**
Fall	Back left	30
	Back lying	36
	Back right	45
	Down-syncope	39
	Down-syncope-wall	59
	Front falls	78
	Front knees falls	68
	Front left	76
	Front right	81
	Lateral-Right-side	65
	Lateral-left-side	66
	Total Falls	**643**
Non-Fall	Bending	82
	Limp	80
	Lying-bed	84
	Rising-bed	39
	Sitting	313
	Sitting-chair	78
	Squatting-down	75
	Stumble	29
	Walking	36
Total	**Total Non-Falls**	**816**
		1459

Table 3.1: Falls and non-falls actions distribution

Table 3.1 Shows a detailed view of the number of actions (falls vs non-falls). The number of falls videos is 643 which is 44% of the whole dataset. Most of the fall types are equally distributed with small differences, however, the number of 'Sitting videos' is much larger than the other action types 21% of the full dataset.

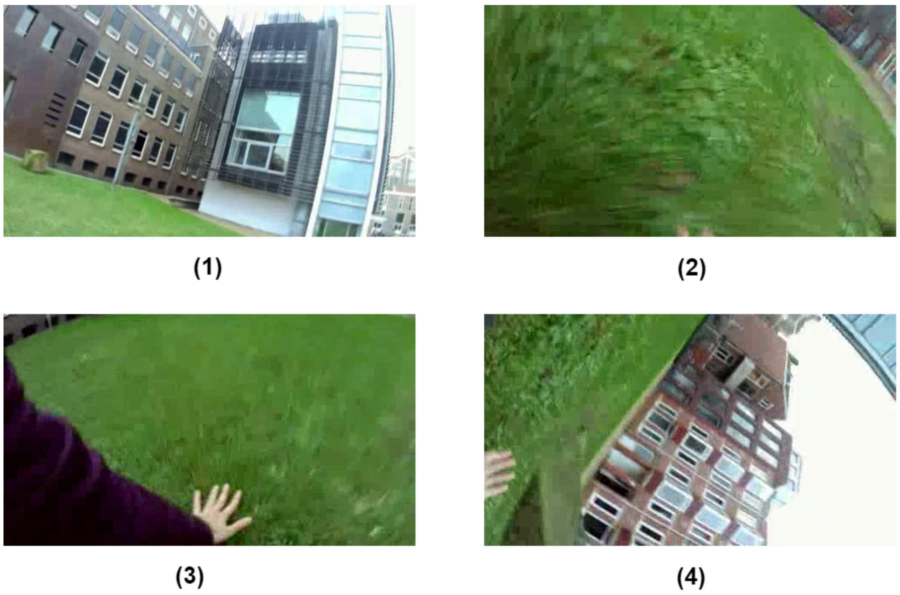


Figure 3.4: Example images sequence from a fall video

Figure 3.4 shows an example of frames captured from a video containing a fall event. In the sequence we can see that the video starts by a frame that indicates that subject of interest is in an upright position and the buildings are oriented in a normal angle. When the fall starts (frame 2 and 3), there is a significant change in the scene, in which we see the grass instead of the buildings. In frame 4 we can see the hand of the person on the grass; the frame is rotated almost 90 degrees compared to the frame captured at the start of the video. It also shows that in typical outdoor falls, the sky takes a significant part of the frame.

3.2 METHODS

In this section we go in details about the theory of the methods used to extract features from the videos in our fall detection system, we first talk about the cosine similarity used as a similarity distance measure during our work, then we talk about HOG in details, then LPB then pyramidal Lucas-Kanade descriptor.

3.2.1 *Cosine Similarity*

We use cosine similarity for comparing the two vectors,
Cosine similarity measures the cosine of the angle between two vectors in a multi-dimensional space. In this context, the two vectors are the normalised histograms.

$$\cos(\theta) = \frac{\vec{a}.\vec{b}}{\|\vec{a}\|\|\vec{b}\|} = \frac{\sum_{i=1}^{n} \mathbf{a}_i \mathbf{b}_i}{\sqrt{\sum_{i=1}^{n} (\mathbf{a}_i)^2}\sqrt{\sum_{i=1}^{n} (\mathbf{b}_i)^2}} \tag{3.1}$$

Equation 3.1 shows how the cosine similarity is calculated between two vectors $\vec{a}$ and $\vec{b}$ Where $\vec{a}.\vec{b} = \sum_{i=1}^{n} \mathbf{a}_i\mathbf{b}_i = \mathbf{a}_1\mathbf{b}_1 + \mathbf{a}_2\mathbf{b}_2 + ... + \mathbf{a}_n\mathbf{b}_n$ is the dot product of two vectors a and b. The resulting similarity ranges from 1 denoting the two vectors are opposite, to 1 meaning exactly the same.

3.2.2 *Local descriptors*

3.2.2.1 *Histogram of oriented gradients*

The HOG descriptor algorithm for each frame has the following steps:

1. Divide image into small connected cells
2. For each cell compute the histogram of edge orientations for the pixels within the cell as shown in Figure 3.5 and Equation 3.2
3. Each cell gets discretized into angular bins according to the gradient orientation.
4. The cell's pixel contributes with the weighted gradient to the corresponding angular bin.
5. Each group of adjacent cells is considered as spatial regions. The grouping of the cells into a block is the basis for grouping and normalisation of histograms.
6. Normalise the group of histograms represents the block histogram. The set of these block histograms represents the descriptor. [72]

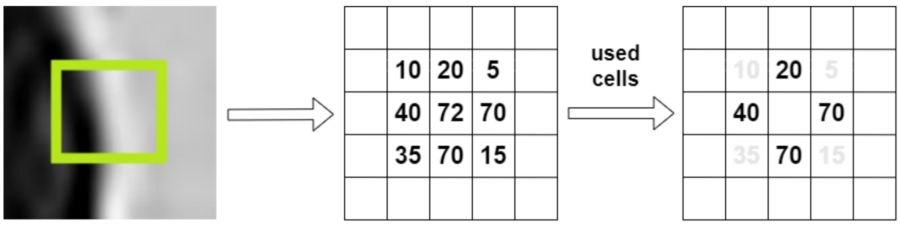


Figure 3.5: HOG calculation

$$
\begin{aligned}
X\,direction &= |40 - 70| = 30 \\
Y\,direction &= |20 - 70| = 50 \\
Gradient\ Magnitude &= \sqrt{30^2 + 50^2} = 58 \\
Gradient\ Direction &= tan^{-1}(\frac{30}{50}) = 30\,^{\circ}
\end{aligned}
\tag{3.2}
$$

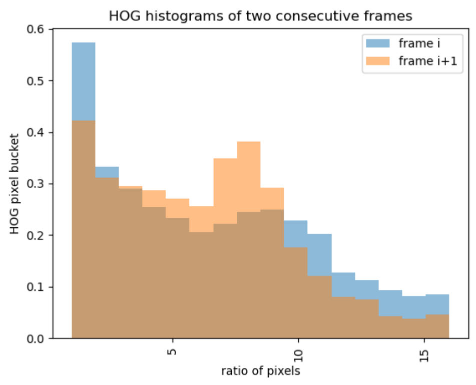


Figure 3.6: HOG histograms of two consecutive frames

After obtaining the histogram of each frame, we compare it to the previous frame histogram. Figure 3.6 shows the HOG histogram of two consecutive frames where the number of bins is 16. We can see that the two histograms do not overlap perfectly. We compute the cosine dissimilarity of the two frames using the equation Equation 3.1 and save it into a one-dimensional array till there are no more frames to process. The result will be a vector of length $nframes - 1$.

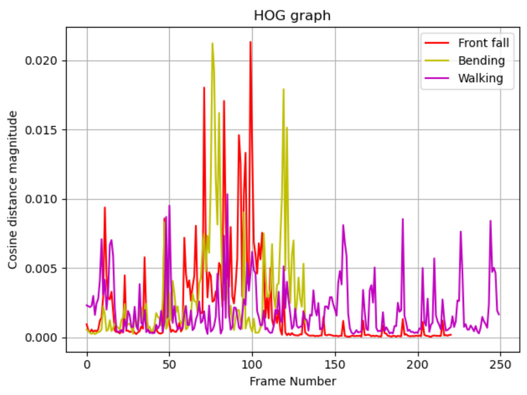


Figure 3.7: HOG graph

Figure 3.7 shows a plot of three hog vectors obtained from three different daily activities videos, namely: front fall, bending and walking. We can see that the cosine distance magnitude of the falling is much higher than the magnitude of the other two daily activities.

3.2.3 *Local Binary Pattern*

The LBP operator was originally developed by Ojala et al. [73]. The main idea for developing the LBP operator is to describe two-dimensional surface textures by using two complementary measures: local spatial patterns and gray scale contrast.

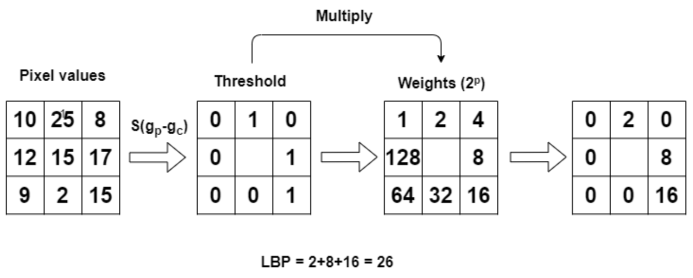


Figure 3.8: LBP calculation example

$$LBP_{p,r} = \sum_{p-1}^{i=0} S(gi - gc)2^i, S(x) = \begin{cases} 1 & if \qquad x \geq 0, \\ 0 & otherwise \end{cases} \tag{3.3}$$

Figure 3.8 shows an example on how LBP values are calculated in a 3 by 3 block.

- **Step one:** calculating threshold: for calculating the threshold we compare each neighbouring pixel with the centre pixel grayscale value. If the value of the neighbouring pixel g_i is greater or equal to the center pixel value g_c of the neighbouring pixel the threshold value of the pixel g_i will be 1; otherwise its 0.
- **Step two:** multiplying the threshold with weights
- **Step three:** summing up the convolved values.

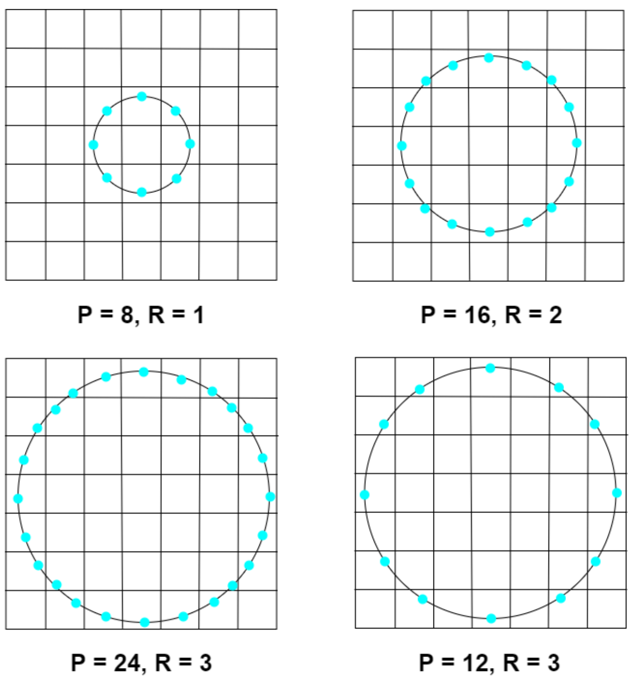


Figure 3.9: Circular multiscale LBP

The multi-scale version of the LBP operator extends the original one. It uses neighbourhoods of different sizes to be able to deal with large-scale structures that might capture more features of some the texture. [74].

There are two variations of multi-scale LBP when it comes to choosing the neighbour pixels; circular and square. Figure 3.9 gives an example of different neighbourhood sizes and different radius values in a circular multi-scale LBP where P represents the number of pixels in the neighbourhood and R represents the radius of the neighbourhood.

After obtaining the histogram of each frame we compare it to the previous frame histogram Figure 3.10 shows the LBP histogram of two consecutive frames where the number of bins is 56. We can see that the two histograms do not overlap perfectly. We compute the cosine dissimilarity of the two frames using the equation Equation 3.1 till there are no more frames to process.

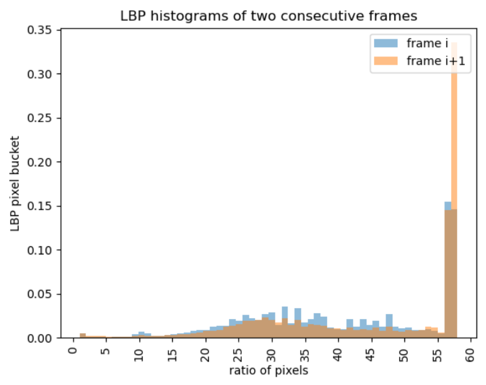


Figure 3.10: LBP histograms of two consecutive frames

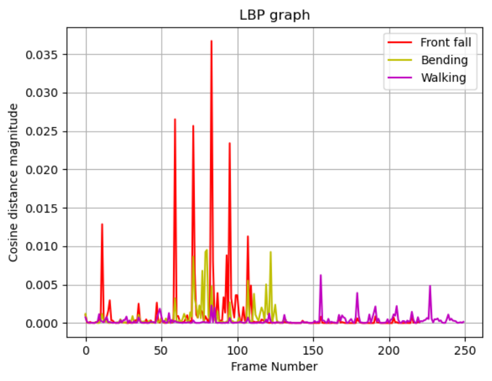


Figure 3.11: LBP graph of three different videos

In Figure 3.11 we see an example plot of the obtained LBP graphs of three different daily activities namely: front fall, bending and walking.

We can see that the cosine distance magnitude of the falling is much higher than the magnitude of the other two daily activities.

3.2.4 *Optical flow estimation*

The algorithms of optical flow estimate the deformations between two images. It tracks image pixels from an image to another as shown in Figure 3.12. The optical flow calculation works on the assumption that pixel intensity is conserved, which means that the intensity or colour of the objects has not changed significantly between the two images. Based on this idea, we have the following assumption:

- Spatial coherence: Nearby points in the frame plane move in a similar manner all the time (velocity smoothness constraint).
- Brightness constancy: projection of the same point looks the same in every frame.
- Small motion: points do not move very far in the next frame, in other words the vector $[u, v]$ is very small.

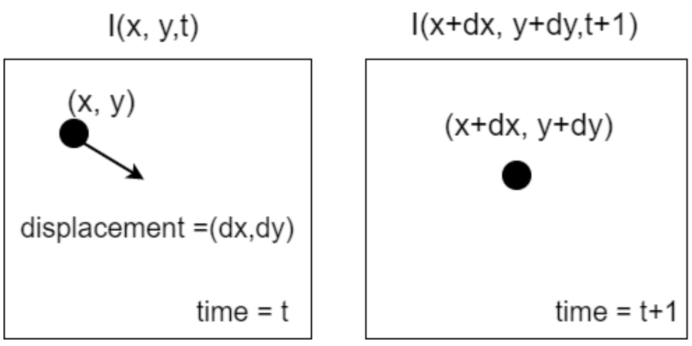


Figure 3.12: Pixel displacement between two consecutive images

$$I(x, y, t) = I(x + \delta x, y + \delta y, t + 1) \tag{3.4}$$

Where $\vec{V} = (\delta x, \delta y)$ is the vector of velocity. Then, by derivation, we obtain the well known optical flow constraint equation

$$\vec{\nabla} I . \vec{V} + \frac{\partial I}{\partial t} = 0 \tag{3.5}$$

Where $\vec{\nabla} I = (\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})$ is the gradient of the image.
However, the optical flow Equation 3.5 contains two unknown variables, namely V_x and V_y and cannot be solved by having only one constraint equation. Therefore, we cannot observe the optical flow using only one observation. This is known as the aperture problem of the optical flow as shown in Figure 3.13 which shows that the change

in the perceived texture in the aperture where all the motions a,b and c can be possible motions.

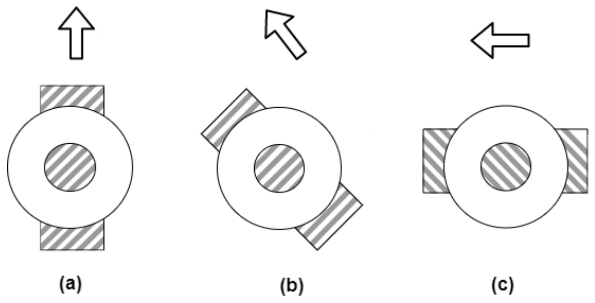


Figure 3.13: Illustration of aperture problem

To be able to solve the equation, additional constraints must be added. One of the most popular methods to solve the aperture problem is the Lucas Kanade-method.

3.2.4.1 *Lucas-Kanade Method*

Lucas-Kanade algorithm can be applied to sparse scenes (sparse optical-flow), it relies on some small windows around our point of interest. The approach of Lucas-kanade method of solving the optical flow equation is based on estimating the displacement at one pixel using only on the information around that specific pixel (inside the window). The Lucas-Kanade algorithm makes the following assumptions:

- The time separating the two is small and increment by Δt. Therefore, the algorithm works best when objects are slow moving.
- The images depict a natural scene containing textured objects exhibiting different gradient values which change smoothly [75].

The equation Equation 3.5 is solved for a pixel x 2 A based on the assumption that the motion field is constant in the neighbourhood $N(x)$ of x, in other words the field is moving in a similar manner around x. An over determined system of equations is obtained and the flow can be computed by minimising the least-squared errors by solving equation Equation 3.8. Note that the equation is solvable only if $A^T A$ is invertible and also not too small.

$$A = \begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_n) & I_y(p_n) \end{bmatrix} v = \begin{bmatrix} V_x \\ V_y \end{bmatrix} b = \begin{bmatrix} -I_t(p_1) \\ -I_t(p_2) \\ \vdots \\ -I_t(p_n) \end{bmatrix} \quad (3.6)$$

$$A = \underbrace{\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix}}_{A^T A} \begin{bmatrix} u \\ v \end{bmatrix} = - \underbrace{\begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}}_{A^T b} \tag{3.7}$$

$$A^T A v = A^T b \quad or \qquad v = (A^T A)^{-1} A^T b \tag{3.8}$$

Algorithm 1 Pseudocode of Lucas Kanade [76]

```
1: $t \leftarrow 0$
2: for $i \leftarrow 0, maxX$ do
3:     for $j \leftarrow 0, maxY$ do
4:         $I_x(i,j,t) \leftarrow CalculateGradientX(j,j,t)$
5:         $I_y(i,j,t) \leftarrow CalculateGradientY(j,j,t)$
6:         $I_t(i,j,t) \leftarrow CalculateGradientT(j,j,t)$
7: $S \leftarrow CreateMatrix(25,2)$
8: $S \leftarrow CreateMatrix(25,2)$
9: $S \leftarrow CreateMatrix(25,2)$
10: for $i \leftarrow 0, maxX$ do
11:     for $j \leftarrow 0, maxY$ do
12:         for $n \leftarrow 1,5$ do
13:             for $m \leftarrow 1,5$ do
14:                 $S((n-1)*5+m,1) \leftarrow I_x(i+(n-3), j+(m-3), t)$
15:                 $S((n-1)*5+m,2) \leftarrow I_y(i+(n-3), j+(m-3), t)$
16:                 $T((n-1)*5+m,1) \leftarrow I_t(i+(n-3), j+(m-3), t)$
17:         $\vec{u} \leftarrow (S^T S)^{-1} S^T(-T)$
18:         $print(''X - vector\ on'', i, j, ''is'', \vec{u}(1,1))$
19:         $print(''Y - vector\ on'', i, j, ''is'', \vec{u}(2,1))$
```

Pseudocode 1 shows the steps taken to calculate the Lucas Kanade optical flow, the algorithms uses two consecutive frames where maxX is the width of the frame and maxY, the height, and t is the time.

3.2.4.2 *Pyramid representation of images*

The pyramid representation of a an image I of size $n_w n_h$. Having the original image level $I^0 = I$ this image has the highest resolution. The image width and height at that level are defined as $n_w^0 = n_w$ and $n_h^0 = n_h$. The rest of the pyramidal representation of the image are calculated recursively, and each image will have $n_w^i = n_w^{i-1}/2$ and $n_h^i = n_h^{i-1}/2$ [77].

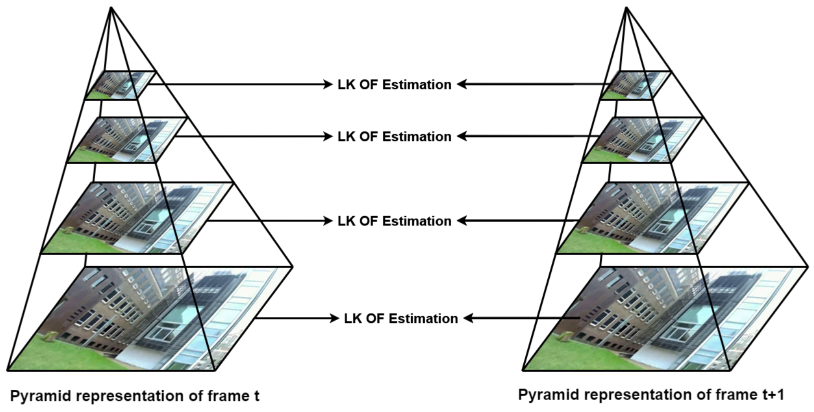

LK OF Estimation
LK OF Estimation
LK OF Estimation
LK OF Estimation
Pyramid representation of frame t
Pyramid representation of frame t+1

Figure 3.14: Lucas-Kanade optical flow pyramid

3.2.4.3 *Pyramidal Lucas-Kanade*

As mentioned in Section 3.2.4.1 one of the limitations of Lucas-Kanade is that it cannot track pixels with large motion amplitude because they end outside the local window. This problem led to the development of the pyramidal LK algorithm.
The pyramid optical-flow tracking procedure is shown in Figure 3.14 demonstrates three-level pyramids of two frames. For a given feature point u on frame t, its corresponding location $v = u + d$ is found on frame $t + 1$, where $d = [dx\, dy]$ is the displacement from u to v. The local affine transformation matrix between frame t and frame $t + 1$ is then created from the vicinity of u and v, respectively. Let L be the level of pyramid image. For m levels, $L = 0, ..., Lm$, define $u^L = [u_L^x\, u_L^y]$, and then the corresponding coordinates of the point u on the pyramidal image I^L is computed as [77]:
$u^L = \frac{u}{2^L}$

Algorithm 2 Pseudocode of Pyramidal Lucas-Kanade algorithm [78]

Input: img1,img2, pyramid level L.
Output: Optical flow field of.

```
1: Generate Gaussian pyramids for $img1$ and $img2$
2: for $i = L \rightarrow 2$ do
3:     Compute the optical flow $f_i$ on pyramid level $i$ using
       iterative $Lucas - Kanade$ method with an initial $guess = f$
4:     $2X$ bilinear interpolate $f_i$ in both height and width
       and store the result in $f$
5: iterative $Lucas - Kanade$ method with an initial
6: $f = f_1$
```

3.2.4.4 *Motion model*

Under the assumption that the motions between video frames follow the similarity model,where the similarity transformation includes a combination of translation, rotation, and isotropic scaling. The scaling is parameterized by the factor s and the rotation The rotation is parameterized by the angle θ [79].

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} s\,cos(\theta) & -s\,sin(\theta) & d_x \\ s\,sin(\theta) & s\,cos(\theta) & d_y \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

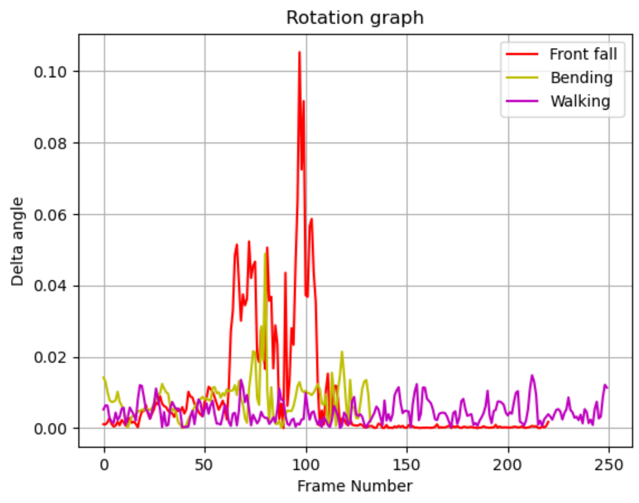


Figure 3.15: Rotation plot

Figure 3.15 shows the unsigned rotation in radiant of three different videos: front fall, bending and walking, the rotation angle magnitude gets bigger when the fall happens, while in the walking and bending the rotation angle magnitude does not go very high.

3.2.4.5 *Motion estimation with optical Gunnar Farneback optical flow*

To estimate the camera motion we have used the method proposed by Gunnar Farneback in his paper [80]. The Gunnar Farneback optical flow is an effective technique to estimate the motion of interesting features by comparing two consecutive frames based on polynomial expansions to estimate the optical flow at every pixel location i.e. dense optical flow.

Instead of solving the optical flow equation, Farneback approximates the intensity information of a neighbourhood of both frames at time t_1 and t_2 with a quadratic polynomial through polynomial expansion transform:

$$f_1(x) = x^T A_1 x + b_1^T x + c_1$$

where x is the pixel coordinate vector in a local coordinate system, A_1 is a symmetric matrix, b_1 is a vector and c_1 is a scalar.

After a global shift d, we construct a new signal f_2 to obtain the shifted neighbourhood as depicted in Equation 3.9.

$$\begin{aligned} f_2(x) &= x^T A_2 x + b_2^T x + c_2 \\ &= f1(x-d) \\ &= (x-d)^T A(x-d) + b_1^T(x-d) + c_1 \\ &= x^T A_1 x + (b1 - 2A_1 d)^T x + d^T A_1 d - b_1^T d + c_1 \end{aligned} \tag{3.9}$$

by equating the coefficients of x the shift can be solved: $A_2 = A_1$, $b_2 = b_1 - 2A_1\,d$ and $d = -\frac{1}{2}A_1^{-1}(b_1 - b_2)$

Next step is to replace the global polynomial in Equation 3.9 with local polynomial approximations.

$$A(x) = \frac{A_1(x) + A_2(x)}{2} \tag{3.10}$$

and introducing

$$\Delta\, b(x) = -\frac{1}{2}(b_2(x) - b_1(x)) \tag{3.11}$$

to obtain

$$A(x)d(x) = \Delta b(x) \tag{3.12}$$

Because solving Equation 3.12 pointwise is too noisy, Farneback made the assumption that the displacement is slow-varying and satisfy a neighbourhood W of x and try to find $d(x)$ satisfying Equation 3.12 as accurate as possible over a neighborhood I of x through the equation Equation 3.13.

$$\sum_{\Delta x \in N} w(\Delta x)||A(x+\Delta x)d(x) - \Delta b(x+\Delta x)||^2 \tag{3.13}$$

where $w\Delta(x)$ is the weight function for the points in the neighbourhood. After obtaining the Farneback optical flow, we calculate the translations for which the length of the vector representing the displacement of each pixel is calculated. However, in practice and to speed up the calculation of the translations we did not use all the frame pixels but every 8^{th} pixel.
To make the motion more accurate and robust, we have removed the outliers before calculating the mean of the translations.

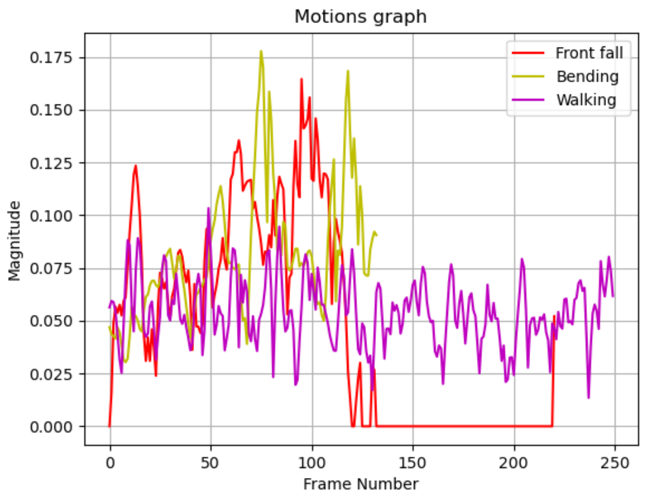


Figure 3.16: Camera motion plot

Figure 3.16 shows the resulting motion graph of three different activities.

3.2.5 *Resampling and aggregating features*

Since the videos have different lengths, we had to resample the length of the extracted features to have the same length for all the extracted features. We have used two ways of resampling the features: 1) by interpolating a new graph of the desired length n from the old array, Figure 3.17 shows an example of resampling a rotation graph to a new graph of length 50 , 2) by taking a sub-array of length n from the original array, to make sure the action is included we take the maximum value of the array as the centre of the new array, Figure 3.18 shows an example of sub-sampling where the new graph will be the graph highlighted in red.

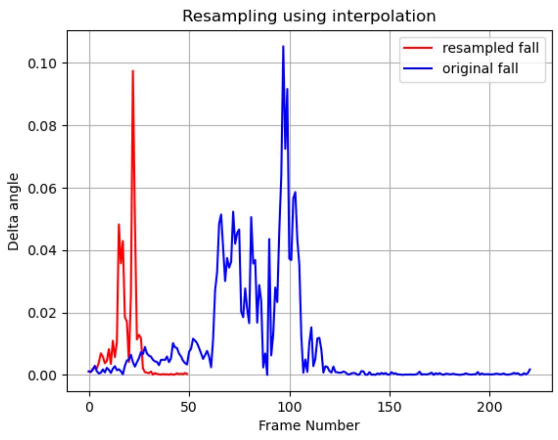


Figure 3.17: Resampling of rotation graph using interpolation

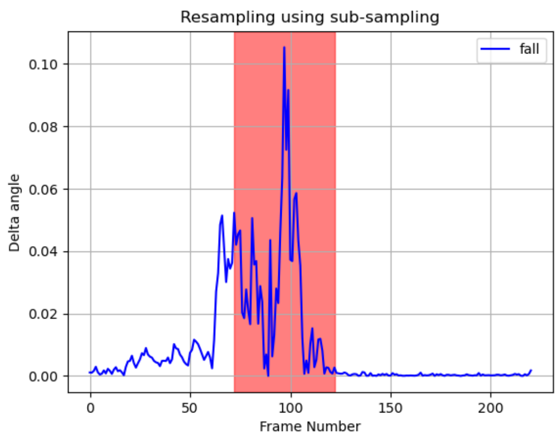


Figure 3.18: Resampling of rotation graph using sub-sampling

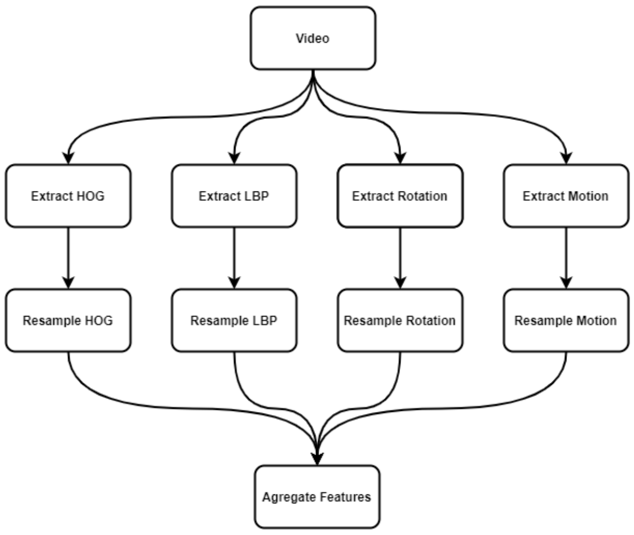


Figure 3.19: Feature extraction, resampling and aggregation

In the case of combining methods, the features will be aggregated, Figure 3.19 shows the steps involved in features extraction and aggregation.

3.3 CLASSIFICATION METHOD

In this section we give a brief explanation of the theory behind the used classification algorithm which is Random Forests. Random Forests were developed by Leo Breiman [58] and they combine the idea of bagging [81] together with a random feature selection. RF combines several individual classification trees like shown in the pseudo-code of the algorithm 3

Algorithm 3 Pseudocode of random Forest algorithm [82]

```
1:  Generate c classifiers:
2:  for $i = 1 \rightarrow c$ do
3:      Randomly sample training data D with replacement to
    produce $D_i$
4:      Create a root node, $N_i$ containing $D_i$
5:      $BuildTree(N_i)$
6:  $BuildTree(N)$ :
7:  if $N$ contains instances of only one class then
8:      return
9:  else
10:     Randomly select $x\%$ of the possible splitting features in $N$
11:     Select the feature $F$ with the highest information gain to split
    on
12:     Create f child nodes of $N, N_1, ..., N_f$ where F has possible values
    $(F_1, F_2..., F_f)$
13:     for $i = 1 \rightarrow f$ do
14:         Set the contents of $N_i$ to $D_i$, where $D_i$ is all instances in $N$
    that matche $F_i$
15:         $BuildTree(N_i)$
```

Random Forest has a high classification accuracy and can deal with large data sets for multiple classes with outstanding time efficiency.

4 EXPERIMENTS AND RESULTS

4.1 INTRODUCTION

In this chapter, we describe the evaluation measures used for the experiments in Section 4.2. In section Section 4.3 we discuss the conducted experiments and their results. We have conducted the following experiments:

Experiment 1: Comparing the performance of the different methods on the binary classification: fall versus non-fall.
Experiment 2: Evaluating the performance of the proposed methods on the multi-class classification of the different daily activities.
Experiment 3: Determining which camera placement has better performance; we compared the results of the methods on binary classification using data collected from the neck camera versus data collected from the waist camera.
Experiment 4: Measuring the computation time of each method.
Experiment 5: Investigating the effect of down-scaling videos on the performance of the suggested methods.

4.2 PERFORMANCE EVALUATION METHODS

For evaluating the conducted experiments we used the following measures:

4.2.1 *Sensitivity and specificity*

Sensitivity: also called true positive rate (TPR) is the portion of falls that are detected correctly as falls with the algorithm under evaluation. Specificity: also called false positive rate (FPR) is the portion of the non-falls that are classified as non-falls in the algorithm under evaluation. Sensitivity and specificity are calculated using Equation 4.1:

$$Sensitivity = \frac{TP}{TP + FN} \qquad Specificity = \frac{TN}{TN + FP} \tag{4.1}$$

- **TP**: true positive: the algorithm classifies a fall activity correctly as a fall.
- **FP:** false positive: the algorithm classifies a non-fall activity falsely as a fall.

- **TN:** true negative: the algorithm classifies a non-fall activity correctly as a non-fall.
- **FN:** false negative: the algorithm classifies a non-fall activity correctly as a non-fall

Sensitivity and specificity are inversely proportional, which means that when the sensitivity increases, the specificity decreases and vice versa.

4.2.2 *ROC-curve*

ROC-curve stands for Receiver Operating Characteristic; it is used to measure classification problems at various thresholds settings and it is calculated following Equation 4.2. The area under the ROC curve (AUC) measures the whole two-dimensional area underneath the ROC curve. AUC is used as the measure of a diagnostic test's discriminatory power [83]. It shows the model's capability to distinguish between classes. The higher the AUC, the better the model. AUC is one of the most commonly used metrics to evaluate model performance for the following two reasons:

- AUC is scale-invariant; it measures how well predictions are ranked, rather than their absolute values.
- AUC is classification-threshold-invariant; it measures the quality of the model's predictions regardless of what classification threshold is chosen.

$$AUC = \int_{1}^{0} TPR(FPR)d(FPR) \tag{4.2}$$

where FPR is true Positive rate and FPR is false positive rate.

4.2.3 *Cross-validation*

Cross-validation is defined as the process for creating a distribution of pairs of training and test sets out of a single dataset [84], there are various methods used for cross-validation. Commonly used methods are listed below.

- **Holdout cross-validation:** In this method, test data is held out during the training phase, therefore, there will be no overlapping between the training and testing datasets. In this method the validation data is not used during the training phase and the system performance is dependent on the choice of the training and testing subsets.

- **K-Fold cross-validation:** The dataset in this method is divided into k equally sized subsets. The training and validation are performed in k iterations. During each iteration, one model is trained on $k-1$ subsets. The left-out subset is used to test the model. The advantage of K-Fold cross-validation is that all data samples are used for both training and testing.

- **Leave one-out cross-validation:** In this method all data samples except one observation are used for training and one instance of data is used for testing. It is a special case of K-fold cross-validation [84] [85].

- **Repeated K-fold cross-validation:** the K-fold cross-validation is executed many times [85].

In the learning process, our data is split into two subsets: training and testing data. The training data is 90% of the full data and the hold out data is 10%. The training data is used for 10-fold cross-validation. The training data set is divided into 10 subsets, and the holdout method is repeated 10 times. Each time, one of the subsets is used as the test set and the other subsets are put together to form a training set. Then the average AUC is computed across all 10 trials.

The advantage of using cross-validation is that the model does not get effected much by the way the dataset is divided. Every file in the dataset will be in the test once, and gets to be in a training set $k-1$ times. The variance of the resulting estimate is reduced as k is increased.

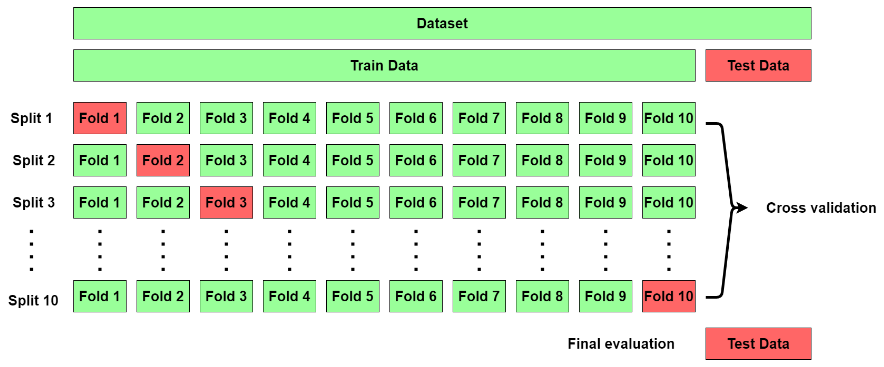


Figure 4.1: 10 folds cross-validation with holdout

4.3 PERFORMANCE EVALUATION

In this section, we compare the performance of the feature extraction algorithms on the binary classification (falls versus non-falls). To evaluate our work, we first test each feature extraction algorithm performance alone. After, we combine algorithms to evaluate the performance when the algorithms are combined.

For evaluation we applied cross-validation explained in Section 4.2.3 then we evaluate the algorithm on the test set. The measures used to evaluate this work are ROC curves, AUC, accuracy, precision and recall curves.

4.3.1 *Binary classification*

We first evaluate the methods on binary classification, i.e. fall versus non-fall classification. We test the performance every method separately, then we test the performance of the possible combination of the methods.

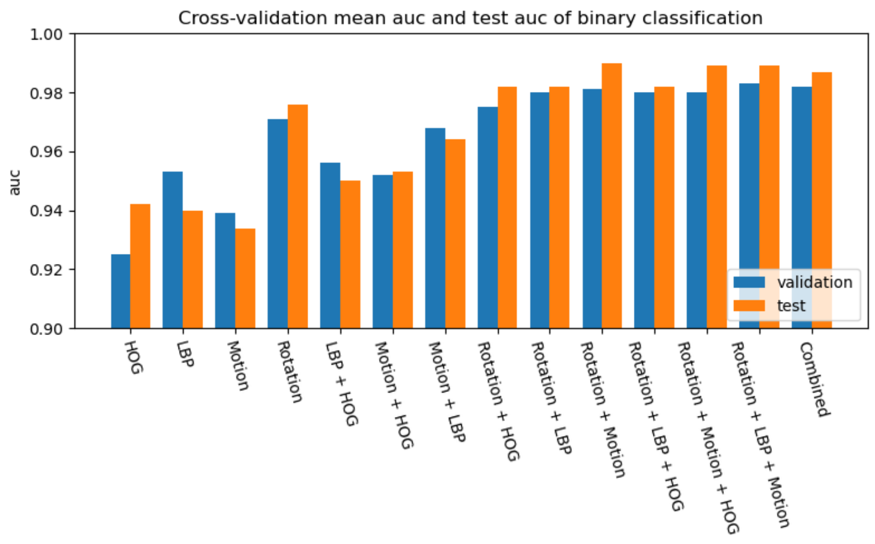


Figure 4.2: Binary classification AUC results

	Validation data		Test data	
Method	**AUC**	**Accuracy**	**AUC**	**Accuracy**
HOG	0.925	0.839	0.942	0.884
LBP	0.953	0.877	0.940	0.884
Motion	0.939	0.855	0.934	0.856
Rotation	0.971	0.910	0.976	0.911
LBP + HOG	0.956	0.896	0.950	0.884
Motion + HOG	0.952	0.880	0.953	0.870
Motion + LBP	0.968	0.906	0.964	0.897
Rotation + HOG	0.975	0.915	0.982	0.911
Rotation + LBP	0.980	0.928	0.982	0.918
Rotation + Motion	0.981	0.924	0.987	0.915
Rotation + LBP + HOG	0.980	0.921	0.982	0.918
Rotation + Motion + HOG	0.980	0.918	0.988	0.918
Rotation + LBP + Motion	**0.983**	**0.926**	**0.988**	**0.918**
Combined	0.982	0.922	0.987	0.925

Table 4.1: Methods evaluation on binary classification

Figure 4.2 and Table 4.1 presents the performance results for the different methods (HOG, LBP, Rotation, Motion). We give results obtained with each method separately, but also for all possible combinations.

From the results, we can observe that HOG method has the lowest performance with only 0.925 average AUC and $0,839$ average accuracy during cross-validation, 0.942 AUC and 0.884 accuracy on the test set. Rotation alone has a better performance than LBP, Motion and HOG (0.971 AUC, 0.910 accuracy) during cross-validation, 0.976 AUC and 0.911 accuracy on the test set.

The best performance was by combining rotation, LBP and motion, resulting in an AUC of 0.983 and accuracy 0.926 during cross-validation, 0.988 AUC and 0.918 accuracy on the test set.

4.3.2 *Multiclass classification*

To test the performance of our method for multiclass classification we have done two experiments:

1. Testing the methods on all the 20 action types presented in Table 3.1,

2. Grouping all the fall action sub-types into one fall action type, resulting in 10 action types in total.

The obtained results for 20 classes classification are presented in table Table 4.2; the method used to obtain the results is Rotation + Motion + LBP, and we have applied resampling. We notice that the algorithm has a good performance on the non-fall ADLs such as **Rising-bed**, **Limp**, **Walking**, **Sitting**. However, the algorithm has difficulty classifying fall types such as **Front fall** with average AUC value of 0.672 on cross-validation, and 0.722 on the test set and **Front-knees fall** with 0.68 average AUC on cross-validation and 0.547 on test set.

After further investigation, we found that the algorithm falsely classify falls that are similar, like **Front-knee fall** and **Front-fall**. The overall performance can be seen from the AUC with 0.852 micro, 0.826 macro on average validation and 0.826 micro, 0.790 on the test set.

Action type	Validation	Test
Back left fall	0.685	0.782
Back right fall	0.707	0.696
Back lying fall	0.797	0.637
Bending fall	0.891	0.948
Down-syncope fall	0.770	0.467
Down-syncope-wall fall	0.774	0.665
Front fall	0.672	0.722
Front knees fall	0.680	0.547
Front left fall	0.684	0.478
Front right fall	0.729	0.786
Lateral-Right-side fall	0.833	0.733
Lateral-left-side fall	0.707	0.718
Limp	0.913	0.921
Lying-bed	0.922	0.846
Rising-bed	0.950	0.965
Sitting	0.990	0.999
Sitting-chair	0.895	0.954
Squatting-down	0.960	0.899
Stumble	0.974	0.967
Walking	0.982	0.986
Micro	0.852	0.821
Macro	0.826	0.790

Table 4.2: AUC of multiclass classification of 20 action types using rotation, motion and LBP methods

To improve the results of fall types, we have grouped all the fall types as one action type (fall). Figure 4.3 and Table 4.3 presents the obtained results; we can notice that the fall results when grouped has improved; the mean AUC during cross validation is 0.94 and 0.957.

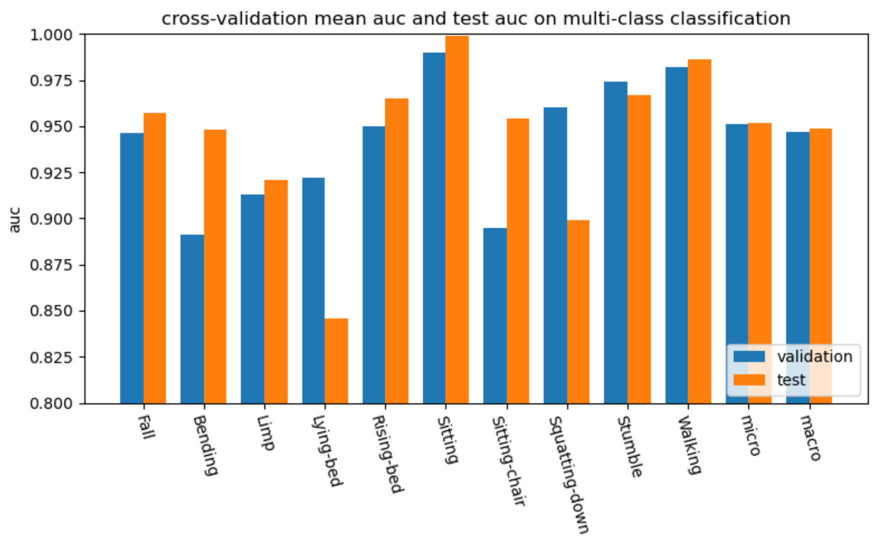


Figure 4.3: Binary classification AUC results

Action type	validation	test
Fall	0.946	0.957
Bending	0.891	0.948
Limp	0.913	0.921
Lying-bed	0.922	0.846
Rising-bed	0.950	0.965
Sitting	0.990	0.999
Sitting-chair	0.895	0.954
Squatting-down	0.960	0.899
Stumble	0.974	0.967
Walking	0.982	0.986
micro	0.951	0.952
macro	0.947	0.949

Table 4.3: Multiclass AUC of 10 action types using rotation, motion and LBP method

4.4 CAMERA LOCATION COMPARISON

To test which camera placement is best, we had to test the algorithms using data only from one type of camera. The dataset size becomes half the size but it is good to find out which camera placement is best for fall detection. Results as shown in Figure 4.4 and Figure 4.5 suggest that the performance of the camera fixed on the waist is often superior to the camera fixed on the neck, both during cross-validation and test. Investigating results in Table 4.5 and Table 4.4 we can observe that the combined features (Rotation + LBP + Motion) gave the best result with average 0.982 AUC and 0.933 accuracy during cross-validation and 1 AUC and 0.986 accuracy on test data. The best results for the neck camera is obtained from combining all the methods, which gave 0.980 AUC and 0.937 accuracy average of cross-validation and 0.984 AUC 0.945 accuracy on test data.

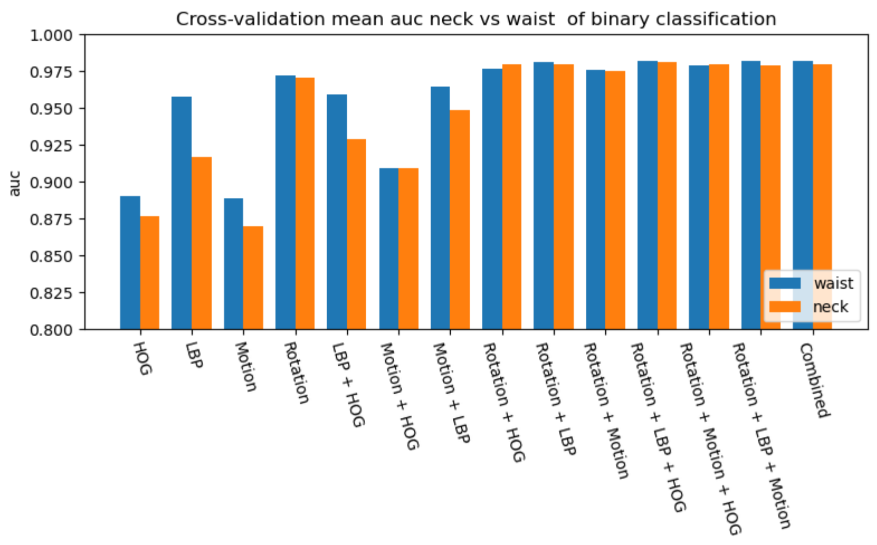


Figure 4.4: Binary classification cross-validation mean AUC of waist camera versus neck camera

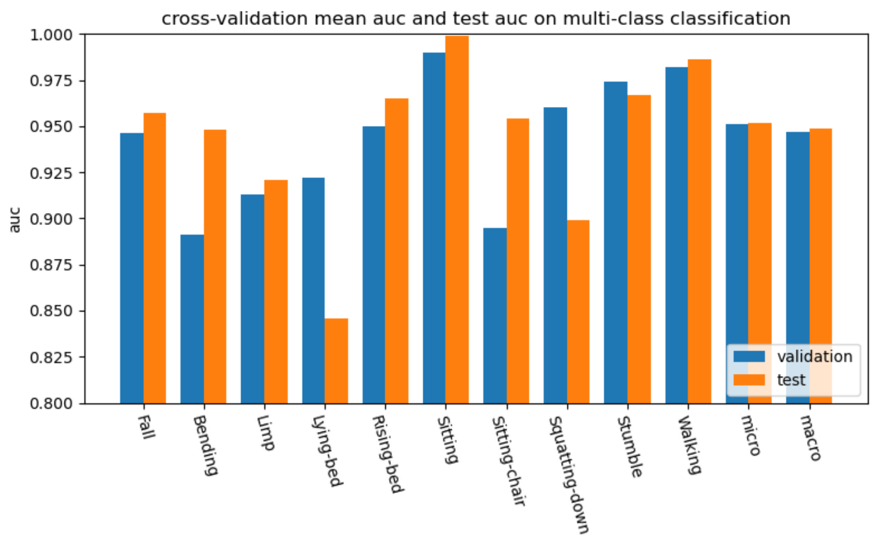


Figure 4.5: Binary classification test AUC of waist versus waist versus neck camera

	Validation data		Test data	
Method	**AUC**	**Accuracy**	**AUC**	**Accuracy**
HOG	0.890	0.808	0.957	0.932
LBP	0.958	0.909	0.992	0.946
Motion	0.889	0.800	0.953	0.865
Rotation	0.972	0.918	0.992	0.946
LBP + HOG	0.959	0.911	0.995	0.959
Motion + HOG	0.909	0.837	0.962	0.878
Motion + LBP	0.965	0.917	0.995	0.959
Rotation + HOG	0.977	0.924	0.996	0.959
Rotation + LBP	0.981	0.929	1.000	1.000
Rotation + Motion	0.976	0.918	0.994	0.932
Rotation + LBP + HOG	0.982	0.932	1.000	1.000
Rotation + Motion + HOG	0.979	0.923	0.997	0.946
Rotation + LBP + Motion	0.982	0.933	1.000	0.986
Combined	0.982	0.927	1.000	0.986

Table 4.4: Data using only waist camera

	Validation data		Test data	
Method	**AUC**	**Accuracy**	**AUC**	**Accuracy**
HOG	0.877	0.791	0.875	0.822
LBP	0.917	0.837	0.916	0.808
Motion	0.870	0.791	0.864	0.808
Rotation	0.971	0.908	0.964	0.863
LBP + HOG	0.929	0.862	0.932	0.822
Motion + HOG	0.909	0.822	0.898	0.849
Motion + LBP	0.949	0.883	0.930	0.849
Rotation + HOG	0.980	0.929	0.981	0.904
Rotation + LBP	0.980	0.937	0.983	0.932
Rotation + Motion	0.975	0.920	0.966	0.863
Rotation + LBP + HOG	0.981	0.934	0.981	0.945
Rotation + Motion + HOG	0.980	0.926	0.978	0.863
Rotation + LBP + Motion	0.979	0.929	0.976	0.904
Combined	**0.980**	**0.937**	**0.984**	**0.945**

Table 4.5: Data using only neck camera

4.4.1 *Computation time*

The computation time test was conducted using Intel Core i5-4690 CPU @ 3.5GHz with 16 GB RAM and NVIDIA GeForce GTX970. The test was done on 10 videos of 10 seconds length (300 frames), which amounted in 100 seconds and 3000 frames. The speed results are presented in Table 4.6:

Method	Speed in seconds	Frames per second
HOG	19.44	154.32
Rotation	23.44	127.98
Motion	323.03	9.28
LBP	1098.16	2.73

Table 4.6: Computation time results on 600*x*400 pixels resolution

From the results we can see that LBP has the longest time, 1098.16 sec., because LBP computation uses the multi-scale method -which calculates 56 neighbouring points for each pixel- which makes it time costly. Motion takes 323.09 sec. Rotation takes 23.44 sec. and HOG 19.44 sec.

Method	Speed in seconds	Frames per second
HOG	6.04	496.68
Rotation	6.23	481.54
Motion	60.06	49.95
LBP	269.29	11.14

Table 4.7: Computation time results on 300*x*200 pixels resolution

After resizing the videos to 300*x*200 pixels resolution the computation time has dropped considerably

4.4.2 *Down-scaling videos*

We evaluate the performance of the suggested methods on smaller video size, by down-scaling the videos from 600*x*400 to 300*x*200.

Figure 4.6 and Figure 4.7 provide a comparison of AUC values between the down-scaled videos and the 600*x*400 pixels videos during cross-validation and testing. From the results we can see that the down-scaled videos results are inferior. However, the difference is not big. The values of down-scaled AUC values are depicted in Table 4.8, the best AUC was obtained by combining all methods with AUC of 0.980 during cross-validation and 0.984 during testing.

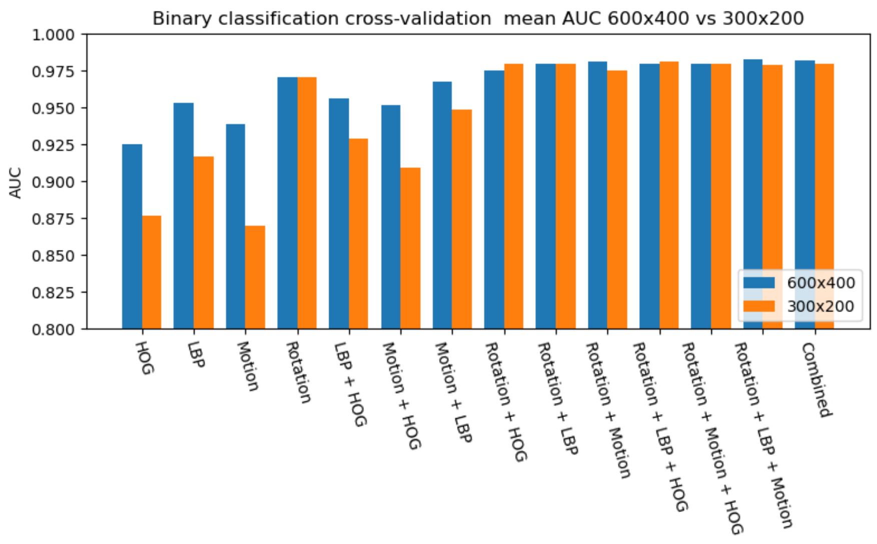


Figure 4.6: Comparing AUC results after down-scaling from 600*x*400 pixels resolution to 300*x*200 pixels

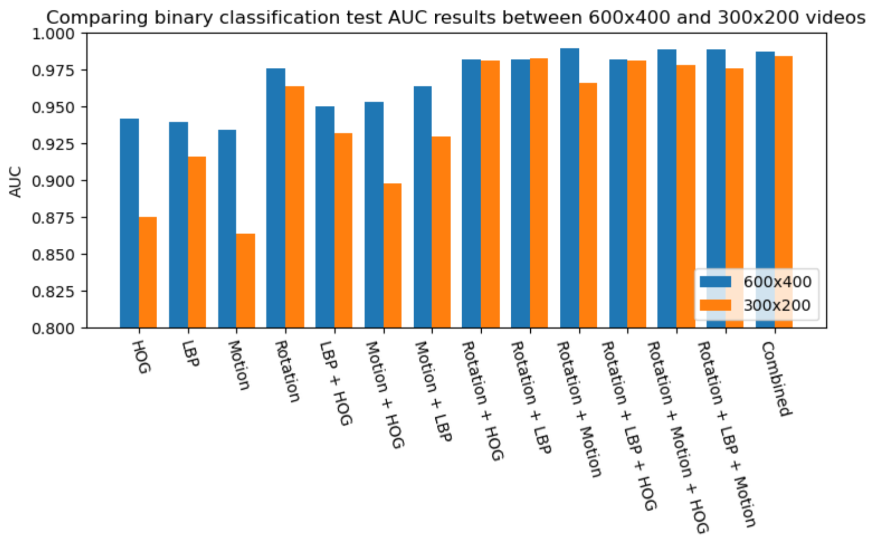


Figure 4.7: Binary classification test AUC of neck versus waist camera

	Validation data		Test data	
Method	**AUC**	**Accuracy**	**AUC**	**Accuracy**
HOG	0.877	0.791	0.875	0.822
LBP	0.917	0.837	0.916	0.808
Motion	0.870	0.791	0.864	0.808
Rotation	0.971	0.908	0.964	0.863
LBP + HOG	0.929	0.862	0.932	0.822
Motion + HOG	0.909	0.822	0.898	0.849
Motion + LBP	0.949	0.883	0.930	0.849
Rotation + HOG	0.980	0.929	0.981	0.904
Rotation + LBP	0.980	0.937	0.983	0.932
Rotation + Motion	0.975	0.920	0.966	0.863
Rotation + LBP + HOG	0.981	0.934	0.981	0.945
Rotation + Motion + HOG	0.980	0.926	0.978	0.863
Rotation + LBP + Motion	0.979	0.929	0.976	0.904
Combined	0.980	0.937	0.984	0.945

Table 4.8: Binary classification AUC and accuracy results on 300x200 pixels videos

4.5 DISCUSSION

As discussed in the previous chapters, the videos were down-scaled to a resolution of 600x400 pixels to speed up the processing. For the classification algorithms, we have tested various algorithms and selected RandomForests as the best algorithm for both classifications.

The first experiment Section 4.3.1. The results of using different combinations of features are given in Table 4.1. The best results for the binary classification were obtained by combining rotation, LBP and motion with an average of 98.3% AUC, and 92,6% accuracy during cross-validation. The AUC on test data was 98,8% AUC and the accuracy was 91,8%. However, if we consider speed performance in Table 4.6 we can see that calculating LBP is costly. This introduces a speed versus accuracy trade-off, which makes us consider other models that do not include LBP and have slightly worse results. The best result that does not include LBP method is obtained by combining Rotation with motion which have an average AUC of 98.1%, and average accuracy of 92,4% during cross-validation and 98.7% AUC and 91.5% accuracy during test.

The second experiment Section 4.3.2 was to check if our models can classify different ADLs. We first used all the Daily ADLs including the different types of falls without considering all the sub-types of falls as one ADL. The results in Table 4.2 have demonstrated that while performing well on the non-fall ADLs, the model performance on the different types of falls was not as good. This is because most falls share many similarities. After grouping all the falls sub-types into one fall class, the results obtained in Table 4.3 show that the model performance on the fall class has improved.

The third experiment was to investigate camera placement effect on the models' performance, the results have suggested fixing the camera on the waist of the elderly gives better results.

The fourth experiment aimed at comparing the methods performance in terms of speed. The first finding was the results showing that LBP is slow especially on the $600x400$ pixels resolution. The second finding was that down-scaling the videos has a significant speeding of the algorithms; 3 times for HOG and rotation, 5 times for motion and 4 times for LBP.

The fifth experiment was to test the effect of down-scaling the videos to $300x200$ pixels on the performance of the methods. The results showed that the methods still perform well after the down-scaling, with only slight inferior results, which suggests that down-scaling videos is an option to take into consideration, especially because the speed of the algorithms is considerably improved.

Comparing the overall results of the algorithms in the first experiment, it can be seen that the majority of the chosen algorithms showed good performance both in the cross-validation and the testing phase. When using a single method to detect falls, video rotation is the best choice since it is the most accurate single method in both binary classification and multi-class classification. It is fast enough to be applied in real-time and can be made faster through down-scaling the videos while maintaining the accuracy. Combining methods showed that the model that combines video rotation, LBP and camera motion, gives the best AUC. However, it has a drawback when it comes to speed since it uses LBP. We therefore suggest to consider models that do not include LBP, such as video rotation and camera motion.

5 CONCLUSION AND FUTURE WORK

This chapter presents a summary of the thesis work and the conclusions drawn from it. In this project, an efficient egocentric camera-based fall detection system has been developed. Firstly, a conclusion of our work in which we summarise our findings and key contributions is given in Section 5.1. Secondly, the limitations of our work are discussed in Section 5.2. Finally, possible future research directions that could be pursued to improve and extend this work is pointed out in Section 5.3.

5.1 CONCLUSION

The thesis project aimed at designing a fall detection system that can be both accurate and fast. During this work, we have implemented and evaluated several methods, for fall detection in particular and action recognition in general. We conclude our work pointing out the key contributions and limitations.

5.1.1 *Key contributions*

During this thesis work, we contributed to fall-detection of the elderly in particular and human action recognition in general in the following directions:

Manual preprocessing of the collected data; we segmented the videos so that each video contains one action, and trimmed the segmented videos with the purpose of obtaining a homogeneous dataset.

Reviewing and analysing popular methods used for fall-detection in particular and vision-based action recognition in general and pointing out the strengths and weaknesses of each method.

Implementing different methods and presenting a comprehensive evaluation of each method on both binary and multiclass classification. Moreover, we used a machine learning-based algorithm to classify falls and non-falls. Unlike other studies that used dissimilarity distance between the consecutive frames LBP and HOG histograms, we used the cosine distance since it provides better classification than the dissimilarity distance.

Furthermore, we have investigated the effect of camera placement on the performance of fall-detection and concluded based on the results that the best placement of the camera is the waist.

Finally, we have evaluated the speed of each method to show which methods are more suited for real-time fall detection.

5.2 LIMITATIONS

The limitations of our work can be summarised as follows:

Due to the lack of available egocentric camera fall detection datasets, we had to make use of data collected from a healthy young volunteer and not from elderly people. One of the drawbacks of having a simulated dataset, is that elderly falls might differ from the simulated falls. Another limitation is the small size of the dataset and the limited types of ADLs contained in it. The developed system was not tested on elderly people, but only tested on volunteer data.

5.3 FUTURE WORK

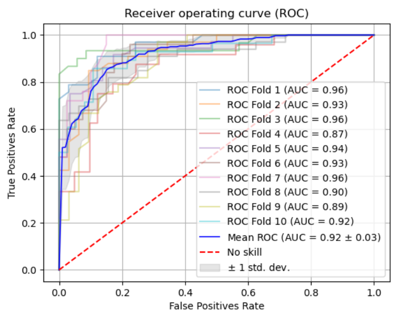
In this section, directions for potential future work that can be carried from this thesis are presented, with the aim to improve the performance of egocentric camera-based fall detection.

- **Dataset improvement:** as mentioned in the limitations, the dataset used in this thesis is relatively small, therefore, enriching the dataset by both increasing the dataset size and including more types of daily activities is recommended. We believe by increasing the dataset size better results could be obtained.
- **D**evelop a classification model that cannot only detect falls but also detect the type of the fall. The type of the fall and the circumstances that led to it, can also help in developing fall-prevention strategies.
- **Implementing deep learning methods** many studies have shown that deep learning techniques have further advanced the domain of computer vision and action recognition in general. Thus, we believe that implementing deep learning-based methods will provide better results, especially when it comes to ADLs classification.
- Finally, the models could be tested in real life on elderly people, to measure its performance in real-life circumstances.

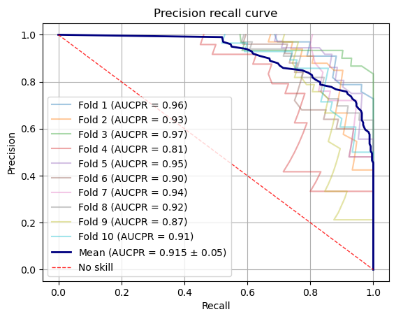
Part I

APPENDIX

A
APPENDIX

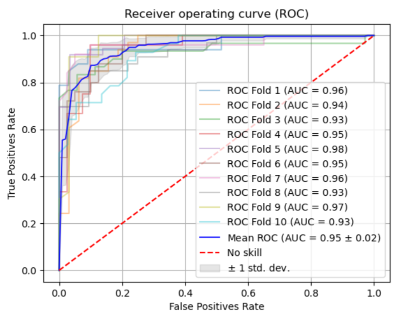


(a) ROC curve.

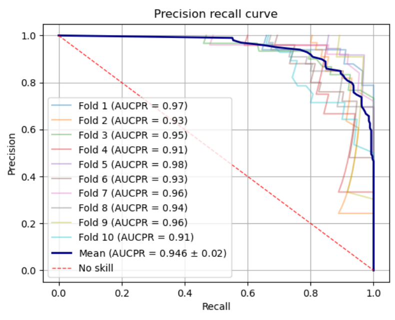


(b) Precision-recall curve.

Figure A.1: ROC and precision-recall curves using HOG.

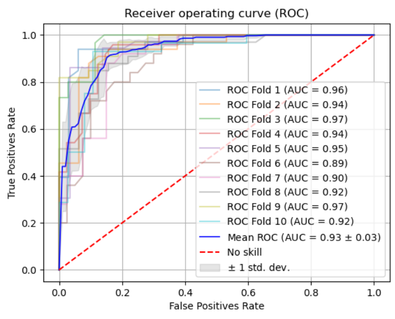


(a) ROC curve.

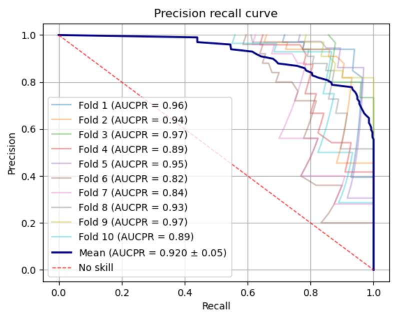


(b) Precision-recall curve.

Figure A.2: ROC and precision-recall curves using LBP.

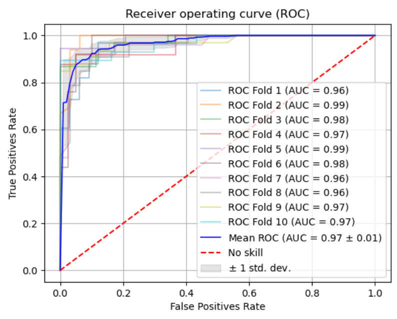


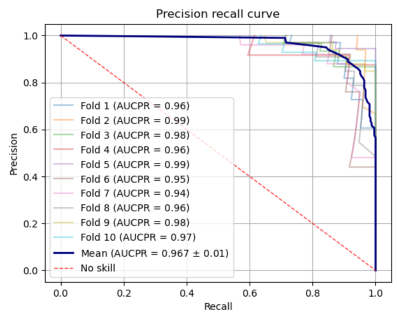
(a) ROC curve.



(b) Precision-recall curve.

Figure A.3: ROC and precision-recall curves using camera motion.

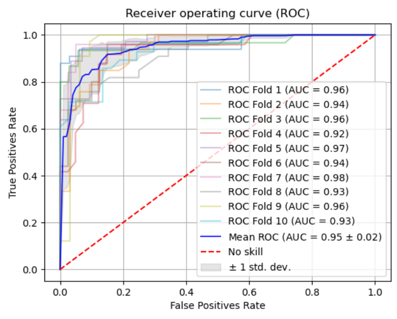


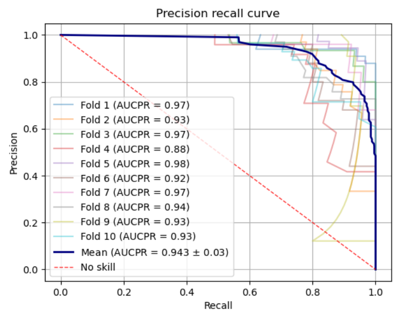


(a) ROC curve.

(b) Precision-recall curve.

Figure A.4: ROC and precision-recall curves using video rotation.

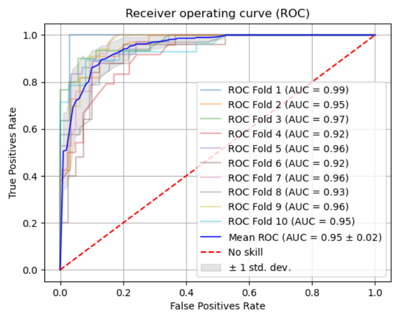


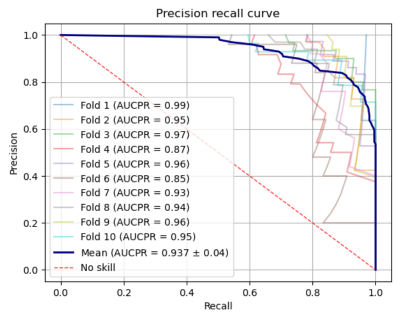


(a) ROC curve.

(b) Precision-recall curve.

Figure A.5: ROC and precision-recall curves using Rotation, LBP and HOG.

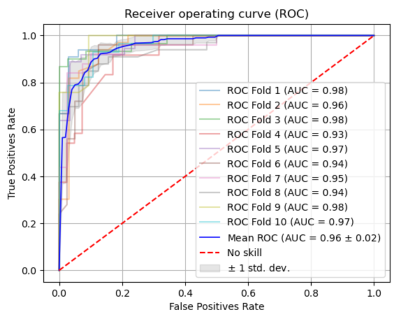




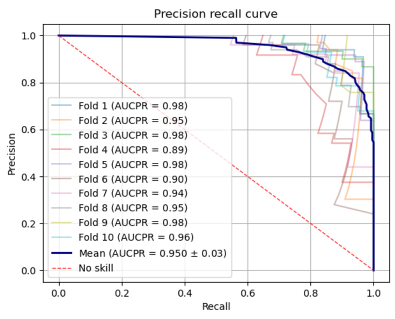
(a) ROC curve.

(b) Precision-recall curve.

Figure A.6: ROC and precision-recall curves using motion and HOG.

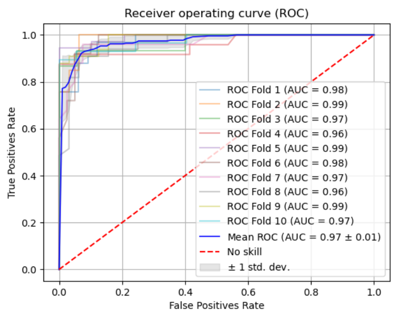


(a) ROC curve.

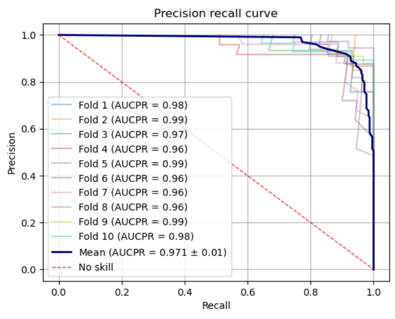


(b) Precision-recall curve.

Figure A.7: ROC and precision-recall curves using motion and LBP.

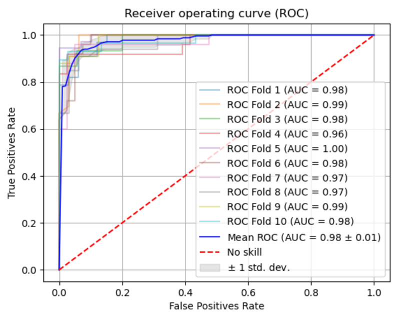


(a) ROC curve.

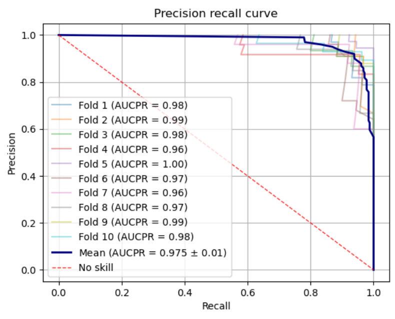


(b) Precision-recall curve.

Figure A.8: ROC and precision-recall curves using rotation and HOG.

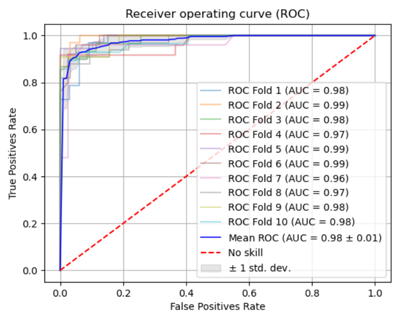


(a) ROC curve.

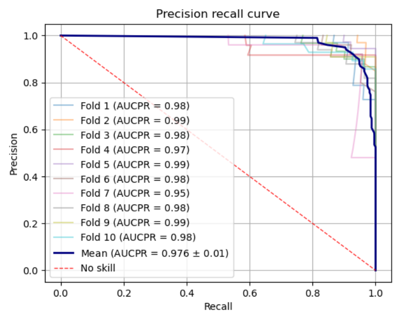


(b) Precision-recall curve.

Figure A.9: ROC and precision-recall curves using Rotation and LBP.

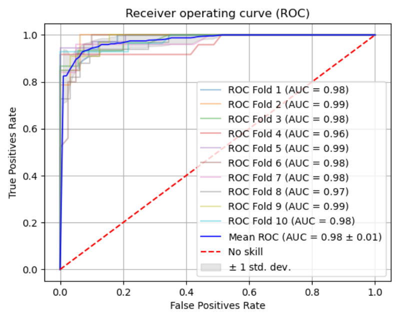


(a) ROC curve.

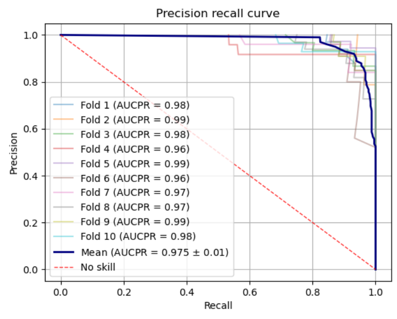


(b) Precision-recall curve.

Figure A.10: ROC and precision-recall curves using rotation and motion.

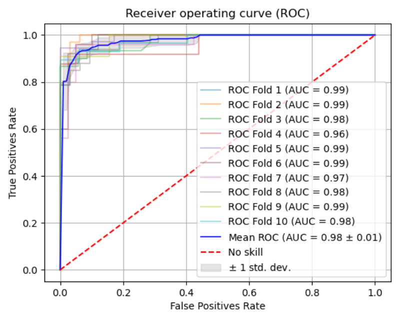


(a) ROC curve.

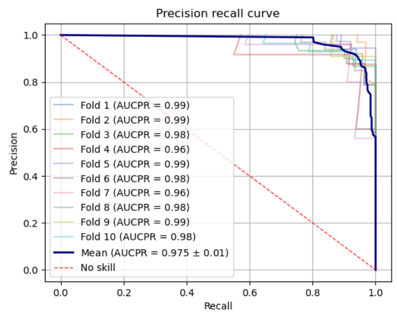


(b) Precision-recall curve.

Figure A.11: ROC and precision-recall curves using rotation, LBP and HOG.

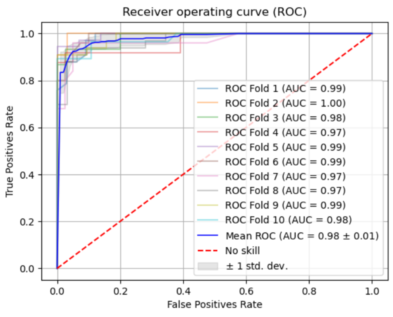


(a) ROC curve.

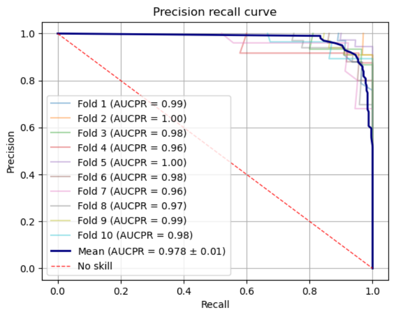


(b) Precision-recall curve.

Figure A.12: ROC and precision-recall curves using rotation, motion and HOG.

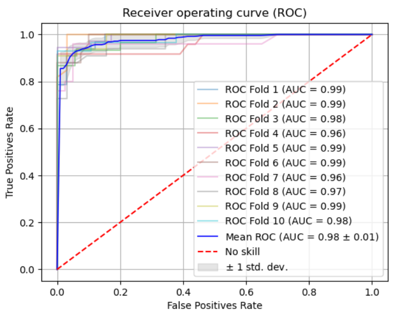


(a) ROC curve.

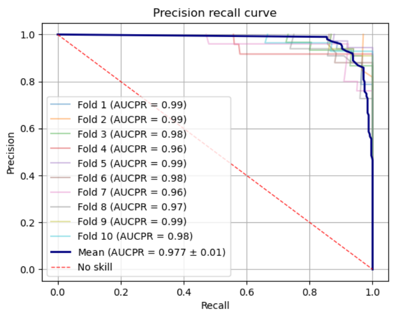


(b) Precision-recall curve.

Figure A.13: ROC and precision-recall curves using rotation, motion and LBP.



(a) ROC curve.



(b) Precision-recall curve.

Figure A.14: ROC and precision-recall curves of all methods combined.

BIBLIOGRAPHY

[1] *The world health organization*. URL: https://www.who.int/news-room/fact-sheets/detail/falls.

[2] Statistics Netherlands. *Population Pyramid*. 2020. URL: https://www.cbs.nl/en-gb/visualisations/population-pyramid.

[3] URL: https://opendata.cbs.nl/#/CBS/nl/dataset/37296ned/table?defaultview&dl=12E27.

[4] Ministerie van Algemene Zaken. *Veilig thuis wonen: in gesprek over valpreventie en brandveiligheid*. 2018. URL: https://www.rijksoverheid.nl/documenten/brochures/2018/03/01/veilig-thuis-wonen-in-gesprek-over-valpreventie-en-brandveiligheid.

[5] *Feiten cijfers*. 2019. URL: https://www.veiligheid.nl/valpreventie/feiten-cijfers.

[6] *Cijfers en feiten valpreventie ouderen*. URL: https://www.loketgezondleven.nl/gezondheidsthema/gezond-en-vitaal-ouder-worden/valpreventie/cijfers-en-feiten-valpreventie-ouderen.

[7] Kennisplein Zorg voor Beter. *Nieuwe vorm financiering*. 2020. URL: https://www.zorgvoorbeter.nl/valpreventie-ouderen/health-impact-bond.

[8] Jennifer L Kelsey, Sarah D Berry, Elizabeth Procter-Gray, Lien Quach, Uyen-Sa DT Nguyen, Wenjun Li, Douglas P Kiel, Lewis A Lipsitz, and Marian T Hannan. "Indoor and outdoor falls in older adults are different: the maintenance of balance, independent living, intellect, and Zest in the Elderly of Boston Study." In: *Journal of the American Geriatrics Society* 58.11 (2010), pp. 2135–2141.

[9] Stefano Abbate, Marco Avvenuti, Paolo Corsini, Janet Light, and Alessio Vecchio. "Monitoring of human movements for fall detection and activities recognition in elderly care using wwireless sensor network: A survey." In: (2010).

[10] Laurence Z Rubenstein. *Falls in Older People - Geriatrics*. URL: https://www.msdmanuals.com/professional/geriatrics/falls-in-older-people/falls-in-older-people?query=fallsinolderpeople.

[11] E Fumio. "Causes of Falls in the Elderly." In: *Journal of the Japan Medical Association* 44.7 (2001), pp. 299–305.

[12] H Luukinen, M Herala, K Koski, R Honkanen, P Laippala, and S-L Kivelä. "Fracture risk associated with a fall according to type of fall among the elderly." In: *Osteoporosis International* 11.7 (2000), pp. 631–634.

[13] Rafael Luque, Eduardo Casilari, María-José Morón, and Gema Redondo. "Comparison and characterization of android-based fall detection systems." In: *Sensors* 14.10 (2014), pp. 18543–18574.

[14] Carla Taramasco, Tomas Rodenas, Felipe Martinez, Paola Fuentes, Roberto Munoz, Rodrigo Olivares, Victor Hugo C De Albuquerque, and Jacques Demongeot. "A novel monitoring system for fall detection in older people." In: *IEEE Access* 6 (2018), pp. 43563–43574.

[15] Yuxi Wang, Kaishun Wu, and Lionel M Ni. "Wifall: Device-free fall detection by wireless networks." In: *IEEE Transactions on Mobile Computing* 16.2 (2016), pp. 581–594.

[16] Dima Litvak, Yaniv Zigel, and Israel Gannot. "Fall detection of elderly through floor vibrations and sound." In: *2008 30th annual international conference of the IEEE engineering in medicine and biology society*. IEEE. 2008, pp. 4632–4635.

[17] Oussama Kerdjidj, Naeem Ramzan, Khalida Ghanem, Abbes Amira, and Fatima Chouireb. "Fall detection and human activity classification using wearable sensors and compressed sensing." In: *Journal of Ambient Intelligence and Humanized Computing* 11.1 (2020), pp. 349–361.

[18] Stefano Abbate, Marco Avvenuti, Francesco Bonatesta, Guglielmo Cola, Paolo Corsini, and Alessio Vecchio. "A smartphone-based fall detection system." In: *Pervasive and Mobile Computing* 8.6 (2012), pp. 883–899.

[19] Carlos Fernando Crispim-Junior, Francois Bremond, and Veronique Joumier. "A multi-sensor approach for activity recognition in older patients." In: 2012.

[20] Guangyi Shi, Jiye Zhang, Chao Dong, Peng Han, Yufeng Jin, and Jack Wang. "Fall detection system based on inertial mems sensors: Analysis design and realization." In: *2015 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*. IEEE. 2015, pp. 1834–1839.

[21] Raymond YW Lee and Alison J Carlisle. "Detection of falls using accelerometers and mobile phone technology." In: *Age and ageing* 40.6 (2011), pp. 690–696.

[22] Zhong Zhang, Christopher Conly, and Vassilis Athitsos. "A survey on vision-based fall detection." In: *Proceedings of the 8th ACM international conference on PErvasive technologies related to assistive environments*. 2015, pp. 1–7.

[23] Gunnar Johansson. "Visual perception of biological motion and a model for its analysis." In: *Perception & psychophysics* 14.2 (1973), pp. 201–211.

[24] Weiming Chen, Zijie Jiang, Hailin Guo, and Xiaoyang Ni. "Fall Detection Based on Key Points of Human-Skeleton Using OpenPose." In: *Symmetry* 12.5 (2020), p. 744.

[25] Daniel Weinland and Edmond Boyer. "Action recognition using exemplar-based embedding." In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. 2008, pp. 1–7.

[26] Ahmed Abobakr, Mohammed Hossny, and Saeid Nahavandi. "A skeleton-free fall detection system from depth images using random decision forest." In: *IEEE Systems Journal* 12.3 (2017), pp. 2994–3005.

[27] Samitha Herath, Mehrtash Harandi, and Fatih Porikli. "Going deeper into action recognition: A survey." In: *Image and vision computing* 60 (2017), pp. 4–21.

[28] Ivan Laptev. "On space-time interest points." In: *International journal of computer vision* 64.2-3 (2005), pp. 107–123.

[29] Piotr Dollár, Vincent Rabaud, Garrison Cottrell, and Serge Belongie. "Behavior recognition via sparse spatio-temporal features." In: *2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance*. IEEE. 2005, pp. 65–72.

[30] Shu-Fai Wong and Roberto Cipolla. "Extracting spatiotemporal interest points using global information." In: *2007 IEEE 11th International Conference on Computer Vision*. IEEE. 2007, pp. 1–8.

[31] Geert Willems, Tinne Tuytelaars, and Luc Van Gool. "An efficient dense and scale-invariant spatio-temporal interest point detector." In: *European conference on computer vision*. Springer. 2008, pp. 650–663.

[32] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. "Dense trajectories and motion boundary descriptors for action recognition." In: *International journal of computer vision* 103.1 (2013), pp. 60–79.

[33] Pyry Matikainen, Martial Hebert, and Rahul Sukthankar. "Trajectons: Action recognition through the motion analysis of tracked features." In: *2009 IEEE 12th international conference on computer vision workshops, ICCV workshops*. IEEE. 2009, pp. 514–521.

[34] Ross Messing, Chris Pal, and Henry Kautz. "Activity recognition using the velocity histories of tracked keypoints." In: *2009 IEEE 12th international conference on computer vision*. IEEE, pp. 104–111.

[35] Heng Wang, Alexander Kläser, Cordelia Schmid, and Cheng-Lin Liu. "Action recognition by dense trajectories." In: *CVPR 2011*. IEEE. 2011, pp. 3169–3176.

[36] Farhood Negin and François Bremond. "Human action recognition in videos: A survey." In: *INRIA Technical Report* (2016).

[37] Florent Perronnin, Jorge Sánchez, and Thomas Mensink. "Improving the fisher kernel for large-scale image classification." In: *European conference on computer vision*. Springer. 2010, pp. 143–156.

[38] Dan Oneata, Jakob Verbeek, and Cordelia Schmid. "Action and event recognition with fisher vectors on a compact feature set." In: *Proceedings of the IEEE international conference on computer vision*. 2013, pp. 1817–1824.

[39] Xiaojiang Peng, Limin Wang, Xingxing Wang, and Yu Qiao. "Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice." In: *Computer Vision and Image Understanding* 150 (2016), pp. 109–125.

[40] Xi Zhou, Kai Yu, Tong Zhang, and Thomas S Huang. "Image classification using super-vector coding of local image descriptors." In: *European conference on computer vision*. Springer. 2010, pp. 141–154.

[41] Hervé Jégou, Matthijs Douze, and Cordelia Schmid. "Improving bag-of-features for large scale image search." In: *International journal of computer vision* 87.3 (2010), pp. 316–336.

[42] Mikel D Rodriguez, Javed Ahmed, and Mubarak Shah. "Action mach a spatio-temporal maximum average correlation height filter for action recognition." In: *2008 IEEE conference on computer vision and pattern recognition*. IEEE. 2008, pp. 1–8.

[43] Wann-Yun Shieh and Ju-Chin Huang. "Speedup the multi-camera video-surveillance system for elder falling detection." In: *2009 International Conference on Embedded Software and Systems*. IEEE. 2009, pp. 350–355.

[44] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns." In: *IEEE Transactions on pattern analysis and machine intelligence* 24.7 (2002), pp. 971–987.

[45] Riccardo Mattivi and Ling Shao. "Human action recognition using LBP-TOP as sparse spatio-temporal feature descriptor." In: *International Conference on Computer Analysis of Images and Patterns*. Springer. 2009, pp. 740–747.

[46] Mohammad Farhad Bulbul, Yunsheng Jiang, and Jinwen Ma. "DMMs-based multiple features fusion for human action recognition." In: *International Journal of Multimedia Data Engineering and Management (IJMDEM)* 6.4 (2015), pp. 23–39.

[47] Suad Albawendi, Kofi Appiah, Heather Powell, and Ahmad Lotfi. "Video based fall detection with enhanced motion history images." In: *Proceedings of the 9th ACM International Conference on PErvasive Technologies Related to Assistive Environments*. 2016, pp. 1–7.

[48] Mauricio Casares, Koray Ozcan, Akhan Almagambetov, and Senem Velipasalar. "Automatic fall detection by a wearable embedded smart camera." In: *2012 Sixth International Conference on Distributed Smart Cameras (ICDSC)*. IEEE. 2012, pp. 1–6.

[49] Koray Ozcan, Anvith Katte Mahabalagiri, Mauricio Casares, and Senem Velipasalar. "Automatic fall detection and activity classification by a wearable embedded smart camera." In: *IEEE journal on emerging and selected topics in circuits and systems* 3.2 (2013), pp. 125–136.

[50] Navneet Dalal and Bill Triggs. "Histograms of oriented gradients for human detection." In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*. Vol. 1. IEEE. 2005, pp. 886–893.

[51] Koray Ozcan, Senem Velipasalar, and Pramod K Varshney. "Autonomous fall detection with wearable cameras by using relative entropy distance measure." In: *IEEE Transactions on Human-Machine Systems* 47.1 (2016), pp. 31–39.

[52] Isma Boudouane, Amina Makhlouf, Nadia Saadia, and Amar Ramdane-Cherif. "Wearable camera for fall detection embedded system." In: *Proceedings of the 4th International Conference on Smart City Applications*. 2019, pp. 1–6.

[53] Aihua Mao, Xuedong Ma, Yinan He, and Jie Luo. "Highly portable, sensor-based system for human fall monitoring." In: *Sensors* 17.9 (2017), p. 2096.

[54] Woon-Sung Baek, Dong-Min Kim, Faisal Bashir, and Jae-Young Pyun. "Real life applicable fall detection system based on wireless body area network." In: *2013 IEEE 10th Consumer Communications and Networking Conference (CCNC)*. IEEE. 2013, pp. 62–67.

[55] Gaetano Anania, Alessandro Tognetti, Nicola Carbonaro, Mario Tesconi, Fabrizio Cutolo, Giuseppe Zupone, and Danilo De Rossi. "Development of a novel algorithm for human fall detection using wearable sensors." In: *SENSORS, 2008 IEEE*. IEEE. 2008, pp. 1336–1339.

[56] Vladimir N Vapnik. "Introduction: Four periods in the research of the learning problem." In: *The Nature of Statistical Learning Theory*. Springer, 2000, pp. 1–15.

[57] Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. "Supervised machine learning: A review of classification techniques." In: *Emerging artificial intelligence applications in computer engineering* 160.1 (2007), pp. 3–24.

[58] Leo Breiman. "Random forests." In: *Machine learning* 45.1 (2001), pp. 5–32.

[59] Urbano Miguel Nunes, Diego R Faria, and Paulo Peixoto. "A human activity recognition framework using max-min features and key poses with differential evolution random forests classifier." In: *Pattern Recognition Letters* 99 (2017), pp. 21–31.

[60] Pierre Geurts, Damien Ernst, and Louis Wehenkel. "Extremely randomized trees." In: *Machine learning* 63.1 (2006), pp. 3–42.

[61] Kijung Kim, Guhnoo Yun, Sung-Kee Park, and Dong Hwan Kim. "Fall detection for the elderly based on 3-axis accelerometer and depth sensor fusion with random forest classifier." In: *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE. 2019, pp. 4611–4614.

[62] Xindong Wu, Vipin Kumar, J Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J McLachlan, Angus Ng, Bing Liu, S Yu Philip, et al. "Top 10 algorithms in data mining." In: *Knowledge and information systems* 14.1 (2008), pp. 1–37.

[63] Dorra Trabelsi, Samer Mohammed, Faicel Chamroukhi, Latifa Oukhellou, and Yacine Amirat. "An unsupervised approach for automatic activity recognition based on hidden Markov model regression." In: *IEEE Transactions on automation science and engineering* 10.3 (2013), pp. 829–835.

[64] Ferhat Attal, Samer Mohammed, Mariam Dedabrishvili, Faicel Chamroukhi, Latifa Oukhellou, and Yacine Amirat. "Physical human activity recognition using wearable sensors." In: *Sensors* 15.12 (2015), pp. 31314–31338.

[65] Sadiq Sani, Nirmalie Wiratunga, and Stewart Massie. "Learning deep features for kNN-based human activity recognition." In: CEUR Workshop Proceedings. 2017.

[66] Jamal Deen. *Advances in Imaging and Electron Physics: Silicon-Based Millimetre-wave Technology*. Academic Press, 2012.

[67] Fouzi Harrou, Nabil Zerrouki, Ying Sun, and Amrane Houacine. "Vision-based fall detection system for improving safety of elderly people." In: *IEEE Instrumentation & Measurement Magazine* 20.6 (2017), pp. 49–55.

[68] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." In: *nature* 521.7553 (2015), pp. 436–444.

[69] Lourdes Martínez-Villaseñor, Hiram Ponce, and Karina Perez-Daniel. "Deep Learning for Multimodal Fall Detection." In: *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*. IEEE. 2019, pp. 3422–3429.

[70] Yongqiang Kong, Jianhui Huang, Shanshan Huang, Zhengang Wei, and Shengke Wang. "Learning spatiotemporal representations for human fall detection in surveillance video." In: *Journal of Visual Communication and Image Representation* 59 (2019), pp. 215–230.

[71] Ricardo Espinosa, Hiram Ponce, Sebastián Gutiérrez, Lourdes Martínez-Villaseñor, Jorge Brieva, and Ernesto Moya-Albor. "A vision-based approach for fall detection using multiple cameras and convolutional neural networks: A case study using the UP-Fall detection dataset." In: *Computers in biology and medicine* 115 (2019), p. 103520.

[72] *Histogram of Oriented Gradients (HOG) Descriptor*. URL: https://scc.ustc.edu.cn/zlsc/tc4600/intel/2017.0.098/ipp/common/ipp_manual/GUID-83B0EE35-E5EF-4C23-96A4-F0918DFA826B.htm#:~:text=Discretizeeachcellintoangular,asspatialregionscalledblocks..

[73] Timo Ojala, Matti Pietikäinen, and David Harwood. "A comparative study of texture measures with classification based on featured distributions." In: *Pattern recognition* 29.1 (1996), pp. 51–59.

[74] Abdallah A Mohamed and Roman V Yampolskiy. "Wavelet-based multiscale adaptive LBP with directional statistical features for recognizing artificial faces." In: *International Scholarly Research Notices* 2012 (2012).

[75] Raúl Rojas. "Lucas-kanade in a nutshell." In: *Freie Universit at Berlinn, Dept. of Computer Science, Tech. Rep* (2010).

[76] H Top. "Optical flow en bewegingillusies." MA thesis. Rijkuniversiteit Groningen, 2007.

[77] Jean-Yves Bouguet et al. "Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm." In: *Intel corporation* 5.1-10 (2001), p. 4.

[78] Fan Zhang, Yang Gao, and Jason D Bakos. "Lucas-kanade optical flow estimation on the ti c66x digital signal processor." In: *2014 IEEE High Performance Extreme Computing Conference (HPEC)*. IEEE. 2014, pp. 1–6.

[79] Qiming Luo and Taghi M Khoshgoftaar. "An empirical study on estimating motions in video stabilization." In: *2007 IEEE International Conference on Information Reuse and Integration*. IEEE. 2007, pp. 360–366.

[80] Gunnar Farnebäck. "Two-frame motion estimation based on polynomial expansion." In: *Scandinavian conference on Image analysis*. Springer. 2003, pp. 363–370.

[81] Leo Breiman. "Bagging predictors." In: *Machine learning* 24.2 (1996), pp. 123–140.

[82] Naphaporn Sirikulviriya and Sukree Sinthupinyo. "Integration of rules from a random forest." In: *International Conference on Information and Electronics Engineering*. Vol. 6. 2011, pp. 194–198.

[83] Jerome Fan, Suneel Upadhye, and Andrew Worster. "Understanding receiver operating characteristic (ROC) curves." In: *Canadian Journal of Emergency Medicine* 8.1 (2006), pp. 19–20.

[84] "Encyclopedia of Machine Learning." In: *Encyclopedia of Machine Learning*. Ed. by Claude Sammut and Geoffrey I. Webb. Boston, MA: Springer US, 2010, pp. 249,600,601. ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_469.

[85] Payam Refaeilzadeh, Lei Tang, and Huan Liu. "Cross-Validation." In: *Encyclopedia of database systems* 5 (2009), pp. 532–538.