Tampere University

Vili Sipilä

AUTOMATIC FALL DETECTION BASED ON DATA FROM A WRISTBAND

Master of Science Thesis
Faculty of Medicine and Health Technology
Examiners: Jari Viik
Timo Erkkilä
Prof. Antti Vehkaoja
June 2022

ABSTRACT

Vili Sipilä: Automatic Fall Detection Based on Data from a Wristband
Master of Science Thesis
Tampere University
Master's Programme in Biotechnology and Biomedical Engineering
June 2022

The aim of the thesis was to create a proof-of-concept fall detection system that utilises data produced by a wristband. The types of data that were studied were triaxial accelerometer data and position data. The position data was gathered with ultra-wideband radio frequency signals.

The thesis started with a literature analysis about the currently available fall detection methods. Fall detection can be done by many different means, such as wearable sensors, ambient sensors and camera-based systems. Utilising wearable sensors in fall detection means that the users wear a device that collects data that is then used to detect the possible falls of the user. The detection is often done by using algorithms.

In the stage of practical tests, data from falls and activities of daily life was collected from 5 test subjects. The subjects wore a prototype wristband on their wrist and conducted falls in four directions: to the back, to the front, to the left and to the right. The activities of daily life that were conducted in data collection included walking, sitting down and getting back up and shaking the wrist. The data was saved for later analysis. A threshold-based algorithm that utilised only accelerometry data was developed because the position data proved to be inaccurate in the height dimension. The features that the algorithm based the fall detection on were the height and width of acceleration peaks and the distance between the peaks.

Based on the collected dataset and by using the algorithm that was developed during the thesis work, the fall detection system detected falls with 85.2% sensitivity, 91.1% specificity and 88.6% classification accuracy. The error rate was 11.4%, the positive predictive value 88.1% and the negative predictive value 88.9%. This performance was reached with the acceleration threshold of 2500 mg.

The thesis work succeeded in creating a proof-of-concept fall detection system. The system utilised acceleration data that was sent wirelessly from a wristband to a microcomputer for conducting the fall detection. The performance of the algorithm could still be optimised further by more throrough testing. If accurate indoor positioning data could be collected and then utilised in the fall detection algorithm, the performance of the fall detection could possibly improve.

Keywords: fall detection, wearable sensors, wearable technology, algorithm development, embedded systems

TIIVISTELMÄ

Vili Sipilä: Automaattinen kaatumisentunnistus rannekkeesta saadun datan perusteella
Diplomityö
Tampereen yliopisto
Bioteknologian ja biolääketieteen tekniikan DI-tutkinto-ohjelma
Kesäkuu 2022

Tämän diplomityön päämääränä oli luoda kaatumisentunnistusjärjestelmä, joka hyödyntää rannekkeesta kerättävää dataa. Järjestelmän tavoitteena oli todistaa konseptin toimivuus ja toimia prototyyppinä. Tutkittuja datatyyppejä olivat kiihtyvyys- ja sijaintidata. Sijaintidata kerättiin ultralaajakaista-alueen radiotaajuussignaaleilla.

Työ alkoi kirjallisuuskatsauksen tekemisellä saatavilla olevista kaatumisentunnistusmenetelmistä. Kaatumisentunnistusta voidaan tehdä monella eri menetelmällä, kuten puettavien antureiden, ympäristössä olevien antureiden tai kameroiden avulla. Puettavia antureita käytettäessä käyttäjällä on puettava laite, joka kerää dataa, jota käytetään tunnistamaan käyttäjän mahdolliset kaatumiset. Kaatumisten tunnistaminen tapahtuu usein algoritmien avulla.

Käytännön kokeiden vaiheessa kerättiin dataa kaatumisista ja päivittäisaktiviteeteista viideltä koehenkilöltä. Koehenkilöt pukivat ranteeseensa prototyyppirannekkeen ja tekivät kaatumiskokeita neljään eri suuntaan: taakse, eteen, vasemmalle ja oikealle. Päivittäisaktiviteetit, joista kerättiin dataa, olivat kävely, istuminen alas ja nouseminen ylös sekä ranteen ravistaminen. Kerätty data tallennettiin myöhempää analysointia varten. Työn aikana kehitettiin kynnysarvoihin perustuva algoritmi, joka hyödynsi ainoastaan kiihtyvyysdataa, koska sijaintidata osoittautui epätarkaksi korkeussuunnassa. Algoritmin kaatumisentunnistus perustui kiihtyvyyspiikkien korkeuteen ja leveyteen sekä piikkien väliseen etäisyyteen aikatasossa.

Kerätyn datasetin ja työssä kehitetyn algoritmin perusteella kaatumisia tunnistettiin 85,2 %:n herkkyydellä, 91,1 %:n spesifisyydellä ja 88,6 %:n luokittelutarkkuudella. Virheprosentti oli 11,4 %, positiivinen ennustearvo 88,1 % ja negatiivinen ennustearvo 88,9 %. Nämä arvot saavutettiin käyttämällä kiihtyvyyden kynnysarvoa 2500 mg.

Diplomityön aikana luotiin onnistuneesti rannekkeen keräämään dataan perustuva kaatumisentunnistusjärjestelmä, jolla saatiin osoitettua konseptin toimivuus. Ranneke lähetti keräämänsä kiihtyvyysdataa langattomasti mikrotietokoneelle kaatumisentunnistuksen tekemiseksi. Algoritmin suorituskykyä voitaisiin vielä optimoida perusteellisemmalla testauksella. Jos tarkkaa sisätilapaikannusdataa saataisiin kerättyä ja sen jälkeen hyödynnettyä kaatumisentunnistusalgoritmissa, kaatumisentunnistuksen laatu voisi mahdollisesti parantua.

Avainsanat: kaatumisentunnistus, puettavat anturit, puettava teknologia, algoritmikehitys, sulautetut järjestelmät

PREFACE

This thesis work would not have been possible without the help of many people. First off, I want to express my gratitude towards Veracell for hiring me to carry out my thesis work. I would like to thank Timo Erkkilä, the CEO of Veracell, for acting both as a supervisor and an examiner of my thesis. Thank you to Sergei Häyrynen for spotting my post on LinkedIn and inviting me to an interview. Jari Viik deserves a big thank you for acting as the supervisor from the university as well as an examiner. Thank you for providing me with valuable feedback and guidance throughout the thesis process. I would also like to thank Antti Vehkaoja for agreeing to be one the examiners.

Eemeli Kallio and Antti-Jussi Mäkipää provided me with tons of help in the practical part of the work. Their help in answering my numerous questions saved me a big amount of time. The thesis was a big learning experience for me, and now I know a lot more about embedded devices and their communications than I did when I started the thesis. Thank you also to all the rest of the employees at Veracell for providing me with general tips on useful software development practices. Thank you to all the employees who acted as the test subjects for the fall tests.

in Tampere, 10th June 2022

Vili Sipilä

CONTENTS

LIST OF FIGURES

ABBREVIATIONS AND MARKINGS

3D	Three-dimensional
$\mathbf{a}$	Acceleration
ADL	Activities of Daily Living
AI	Artificial Intelligence
CPU	Central Processing Unit
CSV	Comma-separated Values
$\mathbf{F}$	Force
g	Gravitional Acceleration of the Earth, 9.81 m/s^2
GPS	Global Positioning System
IC	Integrated Circuit
IDE	Integrated Development Environment
ITU	International Telecommunication Union
ITU-R	ITU Radiocommunication Sector
m	Mass
MEMS	Microelectromechanical Systems
NPV	Negative Predictive Value
PPV	Positive Predictive Value
SVM	Support-vector Machine
TNR	True Negative Rate, also known as specificity
TPR	True Positive Rate, also known as sensitivity
UART	Universal Asynchronous Receiver Transmitter
UWB	Ultra-wideband
WHO	World Health Organization

1. INTRODUCTION

According to the World Health Organization (WHO), falls cause the second-highest number of deaths due to unintentional injury in the world[1]. Especially for elderly people, falls are often dangerous and cause severe injuries and even deaths. In Finland, unintentional falls are the leading cause of fatal accidents in the Finnish elderly population[2]. According to Statistics Finland, accidental falls caused 1151 deaths in 2020. Out of these 1151 deaths, 1048 deaths were in the age group of 65 years and older[3].

Multiple factors affect the risk of falling for each individual[2] and these underlying factors cannot necessarily be diminished with technology. Hence, the technological solutions have focused more on detecting falls instead of preventing them. There are also wearable devices on the market that incorporate alarm buttons for the user to press in case of a fall or other emergency. The problem is, however, that in some cases the user is unable to press the button, for example in the case of unconsciousness. Alternatively, the user might not understand the purpose of the button (e.g. if the user has a memory disorders) and not press it when they should or press it constantly when there is no need for an alarm. One additional problem is that the patients do not always want to press the button when needed for example due to embarrassment or avoiding causing trouble to anyone. For these reasons, it is important to have fall detection systems that send an alarm automatically without input from the user.

Fall detection has been studied extensively. There are multiple surveys and comparisons of different approaches and methodologies for fall detection[4]–[6]. One approach is to use wearable sensors and devices, which are the focus of this thesis. A group of researchers led by Maarit Kangas compared the accuracy of low-complexity algorithms for fall detection for accelerometers attached on the body in an article published in 2008[7]. They found that the wrist was not a suitable location for detecting falls, but the waist produced much better results. The best-performing algorithm based on the waist-mounted accelerometer reached a sensitivity of 97% combined with a specificity of 100%. However, the wrist would be a very convenient location for a fall detection device because it offers high usability and could be fitted with other functionalities. Therefore, further research on the suitability of wrist-mounted sensors for fall detection is needed. In the article by Kangas et al. from 2008, the algorithms utilised only accelerometry data[7]. The authors of the article also stated that the conclusions "might be different if more complicated data processing methods were used". Introducing other types of data in addition to accelerometry data, such as three-dimensional (3D) location data, may produce better results in fall detection.

The aim of this thesis was to create a proof-of-concept-level fall detection system that is based on data collected by a wristband. The main types of data that were used were acceleration data from an accelerometer and position data provided by ultra-wideband radio frequency (UWB) communication signals. The main practical tasks were to create a fall detection algorithm based on the collected data. To achieve this, the devices in the system were needed to be programmed to collect, receive and transmit data for further processing. A ready-made positioning algorithm was utilised in the data collection. In the case of a successful proof-of-concept study, the fall detection system could be further developed into a commercial solution that could be used in a large scale in residential homes, hospital settings and possibly private homes to help caretakers in their work. The accuracy of the fall detection system was not aimed at being comparable with the state-of-the-art solutions at the end of the thesis work.

2. BACKGROUND

The goal of fall detection is to notice if a person falls and get help to the person when needed. To decrease the workload of nurses and other caretakers, technological assistance is needed to detect falls accurately. There are multiple different existing methods for fall detection. Usually the methods include an algorithm to detect falls from other activities. Fall detection systems should detect falls with a high sensitivity, which means that no falls or only a very small portion of the falls would go undetected. However, the system should also have a high specificity to keep the number of false positives, in this case events incorrectly classified as falls, low.

According to the Ministry of Social Affairs and Health of Finland, injury prevention can be divided into primary prevention, secondary prevention and tertiary prevention[8]. Providing approriate treatment based on its need, giving first aid and rescuing injury victims are consireded as secondary help. Detecting falls that have already taken place is considered as secondary help. Preventing injuries from happening in the first place is considered as primary prevention. This is the most effective way to decrease the number and severity of injuries and their consequences but, unfortunately in the scope of fall detection, it is a very challenging task.

A major issue in healthcare units is alarm fatigue. The term means a set of factors that make clinicians respond slower to clinical alarms or ignore some of them because alarms occur too often[9]. The number of alarms a caregiver encounters can exceed 1000 in a single work shift. Most of the alarms do not need to be acted upon or are avoidable[10]. This fact can make the caregivers to become less sensitive to the alarms, consequently worsening the response time or rate of the caregivers. The phenomenon is important to take into account, because it can have serious impacts on patient safety, potentially causing life-threatening situations. Fall detection systems that create false alarms could increase alarm fatigue.[9]

A number of examples about the effects of alarm fatigue were reported by Massachusetts Nurses Association in 2011[11]. Between 2005 and 2011, a total of 11 deaths were found to be linked to alarm fatigue in the state of Massachusetts alone. For instance, one patient died because nurses did not respond to an alarm, even though they heard it for approximately 1 hour and 15 minutes. The audible alarm that the nurses heard meant that the battery of the heart monitor of the patient was in need of replacement. When the nurses did not do anything about the alarm, the battery ran out and the alarm stopped. The heart of the patient then stopped and as there was no alarm, the patient succumbed. Another reported case in the same article represents a deadly course of events. The crisis alarm of the heart monitor of an elderly man patient did not sound, due to the fact that the cables of the machine had slipped off the chest of the man almost three hours earlier. This caused another kind of alarm, a "leads-off" alarm, to signal the care staff to put the leads back on the chest of the patient. Instead, someone had turned off the sound of the alarm without fixing the root cause, leaving the leads unconnected. The next morning a nurse found the patient dead in their chair. The two example cases show the worst possible consequences of alarm fatigue.

Detecting the risk of falling or events when a user nearly falls based on sensor data would be more beneficial than only detecting a fall that has already happened. The detection of high-risk behaviour and situations would allow for preventative measures to be carried out. This, however, is much more difficult to achieve than detecting falls, because it would need a vast set of labelled data to actually pinpoint near falls from data. If a person loses balance but manages to stay upright, it does not necessarily cause significant movement or acceleration of the body, even though the user notices a momentary loss of balance. Therefore, it is challenging to create a set of rules to detect events when the user nearly falls. One possible way of creating the labelled data would be to have members of the target group wear the fall detection device for long periods of time and make them press a button after nearly falling. After this it would be easier for the algorithm developers to find the near falls. The main problem is that near falls occur rather rarely and at random, which is why the time frame for gathering data about them needs to span at least days, but most likely weeks. When comparing this to simply conducting simulated falls with the sensor, it is noticeably more time-consuming to gather data about near falls than actual falls.

2.1 Medical Alarms

Different types of medical alarms that healthcare staff encounter are presented in Figure 2.1. The division by Ruskin and Hueske-Kraus[9] divides alarms into two main categories: clinical alarms and technical alarms. Clinical alarms are alarms that are triggered by a change, typically an adverse one, in the physiological state of a patient. Technical alarms, on the other hand, are alarms that mean that the medical device or its related equipment needs to be checked, such as that there are loose connections or that some cords are disconnected. Clinical alarms can be further split into true clinical alarms, false clinical alarms and nuisance alarms.

Nuisance alarms are true alarms, but they do not require any actions from the care staff. For example if the current heart rate of a patient is a little lower than the normal heart rate of

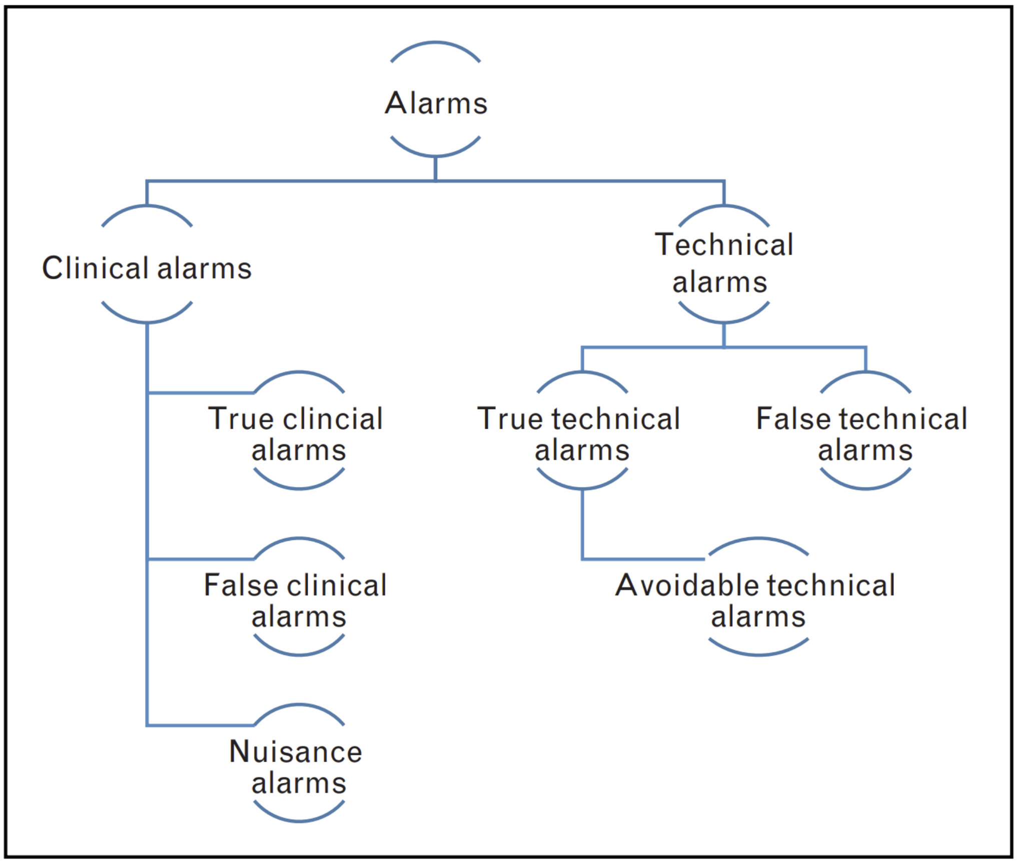


Figure 2.1. *Different types of alarms in a medical setting[9]*

the patient, such as 5–10 beats per minute lower than usual, a heart monitor could trigger an alarm even though the patient would be in good condition at that moment. Alarms of this type are also called nonactionable alarms, because they do not need to be acted upon. True alarms are the alarms that correctly identify when the condition of a patient changes and they need attention. In an ideal situation, the only clinical alarms would be true clinical alarms. False alarms mean that the medical device incorrectly identifies a certain situation and triggers an alarm. For example, if an algorithm that triggers alarms is not functioning as it should, it could classify certain situations incorrectly and cause false alarms. In fall detection this could happen for instance when the patient moves very quickly and a threshold acceleration value is exceeded. The system detects a fall even though no fall happened. Another example could be that a pulse oximeter is giving readings too low about the blood oxygen saturation level of the patient and therefore causing false alarms.[9]

Medical alarm systems are designed to be highly sensitive in order to detect all the important events[10]. The drawback of this choice is that the specificity decreases. The lower specificity means that the number of false alarms in hospital environments is high.

Because of the development of medical technology, the number of monitoring devices and alarm types is increasing, which makes the problem even worse. According to multiple studies, the vast majority of alarms in a hospital environment are often either nuisance alarms or false alarms[12]–[14]. The fall detection system used should not produce daily false alarms and hence increase alarm fatigue, undermining the purpose of getting help to a patient fast in the event of a fall. Having said that, the system still needs to prioritise sensitivity over specificity to not miss falls.

2.2 Motivation for Detecting Falls

Falls are common, and cause devastating results. According to the World Health Organization, a staggering 37.3 million falls cause a need for medical attention every year worldwide[1]. The number of deaths caused by falls is the greatest in adults over the age of 60 years. The World Population Prospects 2019 published by the United Nations Department of Economic and Social Affairs predicts that the worldwide life expectancy will grow rather steadily all the way to year 2100, where the prediction ends. Figure 2.2 shows how the worldwide life expectancy is projected to increase to 77 years by year 2050. This also effectively means that the number of elderly people in the world will increase in the future.

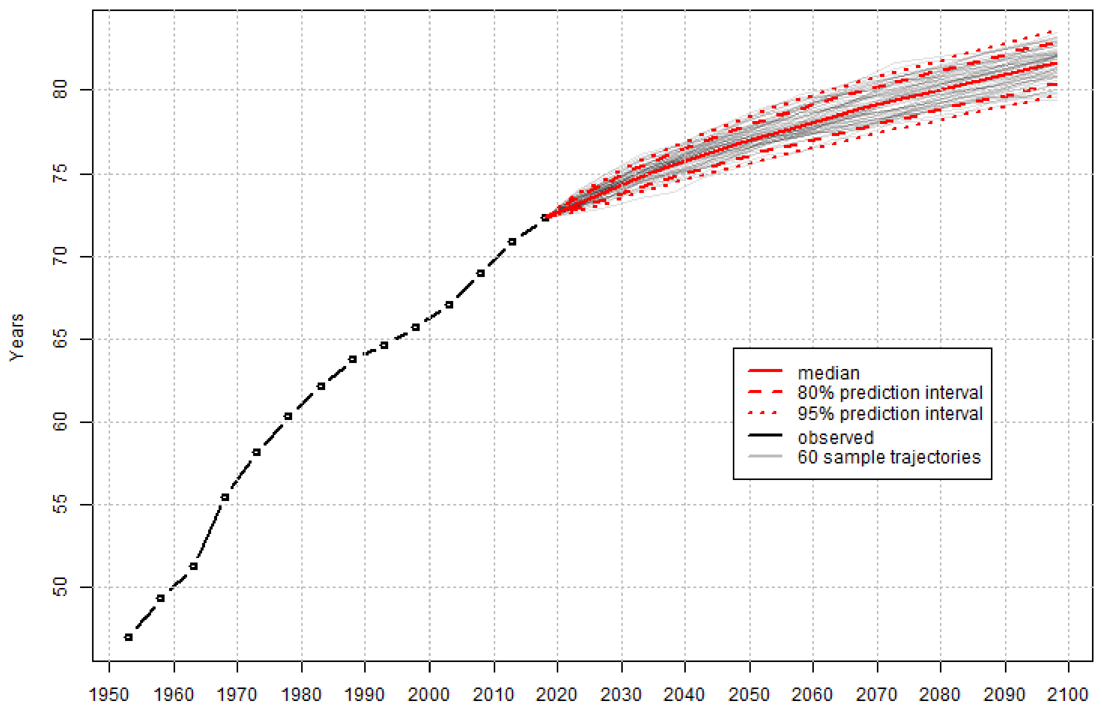


Figure 2.2. *The projected life expectancy over time worldwide. The red lines in the figure represent the estimated human life expectancy in the future using different prediction intervals.[15]*

Detecting falls reliably decreases the time that it takes for help to arrive to the fall victims. Getting the correct treatment quickly can decrease the effect of the sustained injuries and prevent further trauma.

2.3 Fall Detection Approaches

One of the fundamental divisions in fall detection approaches is between using wearable sensors [16], ambient sensors and vision-based systems[6]. Out of these three approaches, using wearable sensors is the only one that requires the user to have anything in contact with themselves. One of the positive attributes of wearable sensors is that the whole fall detection system is rather easy to set up and does not require heavy modifications to the environment of the user. Camera-based approaches can work well with modern advanced signal processing methods, but they pose a greater security risk for privacy compared with other methods without a camera because of the risk of leaking live video data of the patient instead of only sensor data. Even though the communications of the camera can be encrypted, which it often is, the feeling of having surveillance cameras in their home can make people feel uneasy.

2.3.1 Wearable Sensors

Wearable technology is common both in the medical world and as customer products. The development of high-performance, low-cost electronics has enabled manufacturers to continuously add new features on wearable devices. For example, in the past, pedometers that measure the number of steps taken were sold as individual products, but in the current market pedometers are integrated in smart watches, phones or other devices that measure a number of functions. Smart watches often measure heart rate and might have a Global Positioning System (GPS) sensor that measures the location of the user as well. These same features can be used in a medical setting to track the activities of daily living (ADL) of patients.

There are commercial fall detection devices available. Apple Watches incorporate an automatic fall detection feature that the user can choose to activate or disactivate[17]. If the watch detects a fall, it will prompt the user to respond whether they need help or not. In case the user does not react, the device will call the emergency services after 1.5 minutes and notify the emergency contacts of the user after the call. The sensitivity of the fall detection feature of the Apple Watches among wheelchair users was tested by Abou et al. in 2021[18]. They found that the fall detection sensitivity of the Apple Watch was only 4.7% in the fall tests done with a wheelchair. A Finnish company, Suvanto Care, sells a device that the users wear on their neck similarly as a keylace[19]. If the device detects a fall, it will send an alert to a mobile application for the care staff. The users can also press an emergency button on the device to make an alarm. The device has a built-in GPS which allows locating the user after an alarm. Neither Apple nor Suvanto Care disclose the working principle of their fall detection systems, but it is likely that the systems on both devices include at least an accelerometer.

Accelerometers are devices that measure acceleration forces on a given axis or multiple axes. Often an accelerometer has three axes (x, y and z) which are orthogonal. This means that each axis is perpendicular with the others, meaning that the angle between the axes is 90 degrees, as pictured in Figure 2.3. Accelerometers of this type are called triaxial accelerometers. They are convenient in the sense that the combination of three axes makes it possible to detect acceleration in any direction.

Since gravitation is constantly present, it causes a constant force towards the ground. The gravitational acceleration (g) is approximately 9.81 ($\mathrm{m/s^2}$). Because of this force, the orientation of the accelerometer in relationship to the ground can be deducted. If only the gravitational acceleration is present, the absolute value of the total acceleration, as in Equation (2.1), should equal to 1 g[20]. In the equation, x, y and z represent the absolute acceleration values of each axis in g units. The direction of acceleration can be determined as a vector in three-dimensional (3D) space according to Equation (2.2), where x, y and z represent the absolute acceleration values of each axis in g units and i, j and k are

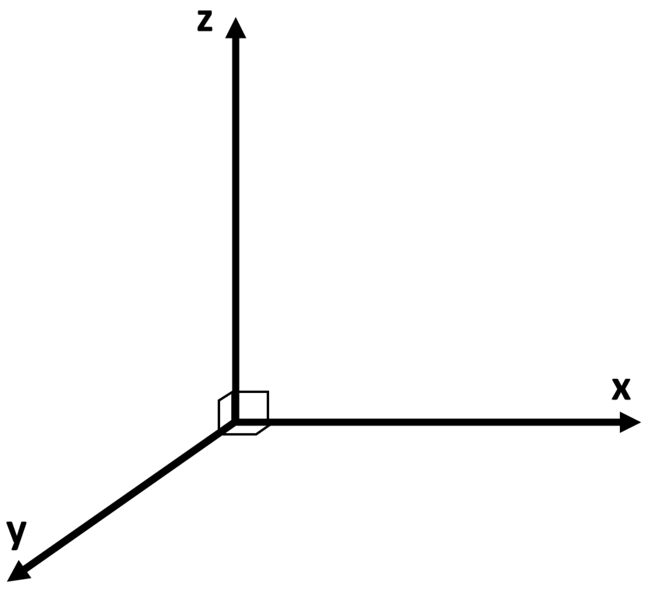


Figure 2.3. *The three axes, x, y and z are orthogonal. The angle between each axis is 90° in respect to the other axes.*

imaginary units. i corresponds to the X-axis, j to the Y-axis and k to the Z-axis. The acceleration can be scaled to mg units by multiplying the values by one thousand.

$$|a_{tot}| = \sqrt[2]{x^2 + y^2 + z^2} \tag{2.1}$$

$$a_{tot} = x \cdot i + y \cdot j + z \cdot k \tag{2.2}$$

Calculating the total acceleration a_{tot} in different points of time and adding the total acceleration vectors up creates information about the changes in the direction and velocity of the accelerometer. However, even small inaccuracies in the acceleration measurements add up over time, and therefore they cause big errors in the estimated position. Due to this error source, there are other methods that are better suited for positioning than accelerometers, such as positioning with radio frequency technology.

Ultra-wideband radio frequency technology (UWB) devices can be utilised in indoor positioning systems. According to the ITU Radiocommunication Sector (ITU-R), a unit of the International Telecommunication Union (ITU), UWB technology is defined as a "technology for short-range radiocommunication, involving the intentional generation and transmission of radio-frequency energy that spreads over a very large frequency range, which may overlap several frequency bands allocated to radiocommunication services. Devices using UWB technology typically have intentional radiation from the antenna with either a –10 dB bandwidth of at least 500 MHz or a –10 dB fractional bandwidth greater than 0.2."[21]

Two main benefits of UWB technology are its high data rate and very low power levels. This combination makes the technology ideal to be applied to systems that need to transmit a high amount of data and that cannot have a high power consumption for example due to a limited battery capacity. Generally UWB technology devices are designed to be robust against interference. The security of UWB signals may also be higher compared to non-UWB signals, because UWB signals can be made to look like noise, the timing of the signals can be randomized and, as mentioned, the bandwidth that the UWB signals cover is large.[21]

Estimating the position of a UWB device requires accurate time measurements and multiple, preferably four or more, UWB devices whose locations are known. In the case of fall detection, the user of a wearable device with UWB technology is the target whose position is not automatically known. There can be a set of UWB devices with known locations fixed to walls. When the device of the user sends a message, the timestamp of the message can be saved. Then, when the stationary devices on the walls receive the message, most likely at different time points, the timestamps of the received messages can be saved. With this information, the time of flight (i.e. the time that it takes for the UWB signal to travel between the devices) can be calculated for each pair of devices separately. To reduce errors, the time of flight can also be calculated as an average time of flight over a few messages. The distance between the devices can be calculated as

$$s = c \cdot t, \tag{2.3}$$

where s is the distance between the devices, c is the speed of light ($c = 299\ 792\ 458\ \mathrm{m/s}$) and t is the time of flight in seconds. When the distances between the device of the user and each of the wall-mounted devices are known, the position of the device of the user can be estimated using trilateration, similarly as in GPS.

Wearable devices have their drawbacks. Sometimes the user does not want to wear the device, maybe because they are ashamed of the device, it is bulky, ugly or uncomfortable, or it might cause skin irritation. Hearing aid usage demonstrates this problem. According to a customer study that was done to hearing aid owners, poor benefit, fit and comfort issues as well as stigma all play a role in not wearing the devices[22]. These issues are also possible in other types of wearable devices, such as wristbands. The device will not serve its purpose if the user is not wearing it. The user might also simply just forget to wear the device. The devices need to be durable, have a decent battery life and still be small enough to be suitable for daily use.

2.3.2 Camera-based Systems

A survey of the principles and approaches of fall detection conducted by Mubashir et al. states that vision-based methods (i.e., using a camera) have a higher accuracy than wearable devices or ambient sensors. The authors also state that by using a camera allows for robust fall detection methods unlike wearable devices and ambient sensors. The downsides are a higher price point and a more difficult setup. The price obviously depends on the camera that is used, and there are also inexpensive cameras widely available. The costs can be decreased if the fall detection algorithms work on a lower resolution or lower frame rate, because these aspects affect the price of cameras significantly.[6]

Computer vision is a key element of camera-based fall detection. The systems utilise advanced image analysis methods to detect and track certain parts of the body or the whole body. In order for accurate camera-based fall detection, the computer vision system needs to detect a person accurately from the video. This is referred to as segmentation. Depending on the approach chosen, the whole body or only part of the body, such as the head, is segmented. One approach of fall detection is to detect changes in the shape of the body, which refers to movement and no change in shape refers to inactivity. Other methods include tracking the posture of the subject or tracking the position of the head in 3D. [6]

2.3.3 Ambient sensors

According to a survey on fall detection, [6], ambient sensors used for fall detection can use audio, video and vibrational data. Over half of the ambient sensor systems track and detect the patient using pressure sensors, often integrated with the floor of the residence or hospital room. Compared to camera-based systems, this type of system can be welcomed better because of its better privacy. Pressure sensors cannot necessarily differentiate which object is causing the pressure. For instance, if the user drops something heavy on the floor, it can cause a high pressure reading from a single location and send out an alarm even though no fall has occurred.

An interesting example of an ambient sensor system setup based on sensor floors was introduced by a reseach group from RTWH Aachen University in 2010[23]. The main idea of the research was that by using floor tiles with piezoelectric force sensors, the movements of a person in a room could be monitored and possible falls detected as well. The authors stated that given all concerns with usability and privacy, the sensor floor approach seems to be the most promising in fall detection in a very private environment. In the study, the researchers created a floor consisting of floor tiles. Underneath the tiles of 0.6 m x 0.6 m, there was a metallic grid forming squares of the same size as the floor tiles. Four piezoelectric elements were installed in each cross point of the grid to support the tiles and measure the force from each corner of each floor tile. The piezoelectric sensor elements were integrated in an acrylic support structure. The basic principle behind piezoelectric measurements is that a piezoelectric element produces a changing voltage according to its deformation. The magnitude of deformation, and thus the produced voltage, depends on the force applied to the element, which makes it possible to use piezoelectric elements as force sensors.

The benefit of using floor sensors is that there is no need for wearable devices that the user would need to actually use and not forget to wear or recharge. Additionally, the system is invisible for the user, and therefore the user does not feel like they are monitored. However, the modifications needed for the floor are comparable to a floor renovation, which would increase the installation costs considerably if the system were to be installed in an existing room instead of to a completely new building. In a further study, the floor presented in [23] was tested with four different test subjects[24]. The researchers found that the accuracy of the floor sensor system in detecting steps taken varied depending on the weight of the test subject. The study still provides a proof-of-concept on this technology. The study did not include fall detection tests, but since the force sensors detect impacts, the technology may be applicable for fall detection as well. A commercial product, Elsi Smart Floor, is available in Finland to track the activities of a person in the room where the system has been installed[25]. The Elsi Smart Floor utilises capacitive sensing and conductance sensing to both track the position of a person in the room and detect falls. The system sends an alarm to the care staff if it detects a fall or if the person is trying to leave their room.

2.4 Fall Detection Datasets

Some publicly available datasets for fall detection exist[26]–[28], but often researchers collect their own datasets for their algorithm development and do not make them publicly available. The biggest limitation of effectively all collected datasets is that their data was gathered with young, healthy subjects using simulated falls. This is understandably the most convenient and ethical way to collect the data, but it poses some problems that need to be taken into account. First of all, simulated falls can not accurately mimic all the details of actual accidental falls and might also include features that are not present in accidental falls. Additionally, young subjects may fall differently than elderly people. For instance, having a better posture and stronger muscles can cause differences in the fall patterns of young people compared to elderly people. Moreover, because falling on a hard surface is dangerous or at least painful also for young and healthy subjects, some kind of pads or mattresses are used in fall testing to soften the impact of the fall. The softened impact affects the gathered fall data by decreasing the acceleration caused by the fall impact as well as by decreasing the fall height.

One of the most comprehensive and recent datasets for fall detection is called "KFall"[28]. To create the KFall dataset, Yu et al. had 32 participants wear an inertial sensor their lower back region. The test subjects then performed 21 different types of daily life activities and 15 types of simulated falls. Yu et al. compared 16 earlier datasets and found that none of them had temporal labels for determining the exact fall time. Hence, to make the dataset more useful, the researchers included time-syncronised videos and temporal labels of the fall times. This allows any researcher to compare actual events with the data that the events produced. Examples of activities of daily life that were performed in the study are standing, picking up an object from the floor, gently jumping to reach an object as well as sitting down to a chair quickly and getting back up quickly. The performed simulated fall types included falls in different directions and in different situations, such as when trying to sit down, get up or slipping. Each subject performed 5 trials of each ADL and fall task except for 4 static tasks where only one trial was tested. The static tasks were standing, sitting on a chair, sitting on a sofa and lying on a bed for 30 s each.

2.5 Fall Detection Algorithms

An algorithm refers to a set of rules to follow to solve a given task or problem. In computing, creating an alogrithm involves creating mathematical formulas that are incorporated in computer programs. In the scope of fall detection algorithms, the input for the algorithm is measurement data, e.g., accelerometry or positioning data, while the output is a prediction whether a fall has taken place or not. Different applications require vastly different algorithms, but there are still similarities. Two main approaches that can be chosen in the development of fall detection algorithms are threshold algorithms and machine-learning algorithms. A study by Yu et al. focused on pre-impact fall detection[28]. In pre-impact fall detection, the fall is detected before the impact with the surface that the person falls on. Therefore the start of the falling motion needs to be detected, which requires accurate ways to differentiate features between normal activities and the start of the falling motion. Yu et al. achieved the highest combination of sensitivity (99.32%) and specificity (99.01%) with a deep learning (a subset of machine learning) algorithm. The next best combination was reached with a support-vector machine (SVM) algorithm, which had a 99.77% sensitivity and a 94.87% specificity. SVM is considered a part of machine learning also. The worst performer of the three tested alternatives was the threshold-based algorithm, with a sensitivity of 95.50% and a specificity value of 83.43%. In the study, the researchers collected their own dataset, KFall, that was used to train and test the algorithms. The choice to focus on pre-impact fall detection may have affected the results, since it is more difficult to develop algorithms for pre-impact fall detection than for the more common post-impact fall detection. It is still important to test pre-impact fall detection, because it could decrease injuries. Helite Hip'Guard is a commercial product that utilises pre-impact fall detection[29]. The device inflates airbags around the hip of the user when an electronic system detects a lateral fall.

Aziz et al.[30] also found in their study that machine learning algorithms performed better than threshold-based algorithms, but the difference between the results was not as clear as in the study by Yu et al. Aziz et al. found that the best-performing machine learning algorithm, which was an SVM algorithm trained by the authors, had a 96% sensitivity and 96% specificity. The best-performing threshold-based algorithm, which was developed by Kangas et al.[7], had a 94% sensitivity and 94% specificity. Aziz et al. also found that near-falls caused a significant number of false positives especially with threshold-based algorithms. The authors acquired their own dataset by using 10 test subjects who were fitted with 7 triaxial accelerometers with a range of $\pm 6\ g$ and a frequency of 128 Hz. The accelerometers were mounted on both ankles, both thighs, waist, sternum and head. Only data from the waist-mounted accelerometer was used in testing the fall detection capabilities of the algorithms.

The main benefit of threshold-based algorithms is that they are more simple than machine-learning algorithms, and they can be modified rather simply. The complexity of machine learning-based fall detection algorithms also means that the algorithms require more computing power than threshold-based algorithms. With modern computing capacity this does not always cause problems, but in an embedded system application the situation is different. In embedded systems both the amount of memory available and processing power can be limited. Also in the case of battery-powered devices the power consumption should be mimimized. Complex calculations consume more power than more simple ones, which favours simple algorithms in battery-powered systems.

2.5.1 Threshold-based Fall Detection Algorithms

Threshold-based algorithms feature a threshold or multiple thresholds for performing fall detection, as the name implies. This means that when a certain measurement exceeds or goes below a certain value based on the application, the system detects a fall. Finding thresholds that combine high sensitivity with high specificity is a difficult task. When using accelerometry data for fall detection, the threshold levels are easy to understand in their most simple form. A simple algorithm could interpret any event where the acceleration value exceeds a predefined threshold value, e.g. $2\ g$, as a fall. This kind of simple system could potentially detect most fall events depending on the chosen threshold because the impact at the end of a fall in general causes an acceleration peak.

A number of events other than falls can also cause an acceleration peak. For this reason, a single threshold may not be enough on its own. Thus, having multiple thresholds measuring different factors of the data can be useful. These thresholds can be combined in the same algorithm to potentially provide more accurate fall detection. To present and visualise the steps of an algorithm, a flowchart can be created. A flowchart is a useful aid

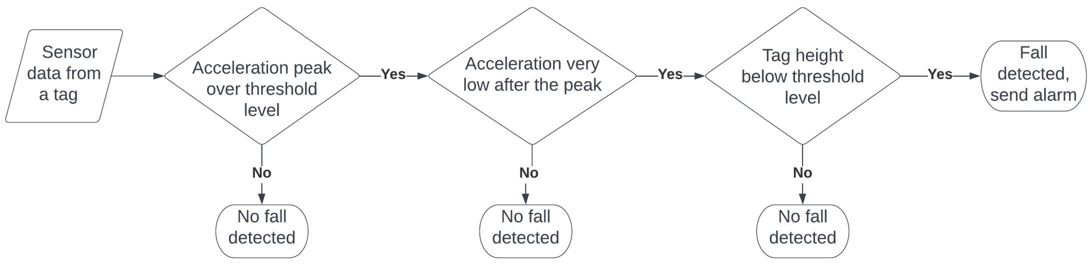

Sensor data from a tag
Acceleration peak over threshold level
Yes
Acceleration very low after the peak
Yes
Tag height below threshold level
Yes
Fall detected, send alarm
No
No fall detected
No
No fall detected
No
No fall detected

Figure 2.4. *Fall detection flowchart example when using a threshold algorithm*

in algorithm development because it visualises the process that should take place and also enables communicating it with others. When the flowchart is done well, it makes it easier to convert the algorithm into actual code that would comprise of multiple nested conditional clauses. Threshold algorithms can get rather complex, but avoiding complexity makes the algorithms easier to understand also for all the people who have not created them and in addition easier to modify for their developers.

An example of a flowchart when using a threshold algorithm for fall detection is presented in Figure 2.4. The flowchart starts from the left. In the example, sensor data is fed to the fall detection system continuously. The system then follows the acceleration values in the input data. If the acceleration does not exceed the set acceleration threshold, the system does not detect a fall. If the acceleration peaks over the threshold value, the event is then investigated further. If the acceleration value is very low after the peak, meaning for a given time and under another threshold value, the user is considered to be inactive and hence possibly fallen. If not, the user is considered to be active and not fallen. If the acceleration value after the peak is very low, the next step is to study whether the position of the wristband in the height position is below or above a given threshold level. If it is not, then no fall is detected. If the position is below the threshold height, as estimated by UWB technology or some other method, the system will deduct that the user is on the floor and that they have fallen. The system will detect a fall and send out an alarm about it. This example consists of three steps of decision making, but the number of steps can be greater or lower than in the example.

Setting the threshold levels just right is a time-consuming task that may involve some trial and error. For example in the case of determining if the user is lying on the floor, the range of values where the height threshold could be set is rather small. There could be overlap with daily activities, such as sitting on a sofa or chair with the arm hanging towards the floor, and if the patient is lying on the floor but trying to wave for help by extending their arm towards the ceiling. If the user is not moving while they are on the floor it is more likely that their wrist with the device is closer to the floor as well, which makes it easier to detect the situation.

2.5.2 Machine Learning-based Fall Detection Algorithms

Machine learning algorithms utilize labelled training data, and the resulting algorithms can then be tested with test data. This basically means that for fall detection, a set of data needs to be gathered with the information whether the data contains a fall or not. Both the input, the measurement data, and the output, fall detected or not, are known. The machine learning system then searches for features in the input data that affect the output. Then, the system builds an algorithm which can combine the inputs with the correct outputs. The process is called training a machine learning model. When the classes of the outputs are known and told to the classifier in the training phase, it is called supervised learning. The opposite of this is unsupervised learning, in which the output classes are unknown beforehand. This type of approach is useful when the developer does not know beforehand what to look for in the data. Then the developer can tell the machine learning training system to divide the data in a given number of classes, after which the training algorithm finds separating features in the data and produces outputs in the given number of classes and an algorithm that would then predict the class for future test data. This is not very useful in qualitative fall detection.

Neural networks are a part of machine learning that are inspired by the way neurons in the human brain work. They are also called artificial neural networks to differentiate the systems from the biological ones. Artificial neural networks consist of nodes that then form discrete layers. Figure 2.5 represents the structure of a deep neural network. The depth of the neural network refers to the number of layers that the network has. If a neural network has at least three hidden layers, it is called a deep neural network[31]. The leftmost layer of an artificial neural network is the input layer, while the rightmost layer is the output layer, as stated in Figure 2.5. The input is the base data that the neural network uses to make its predictions through a varying number of layers. In fall detection, the type of input data depends on the system, but can be for example time series data from wearable sensors or image sequences. The output, whether a fall took place or not, is then computed in the network.

The output of each node is the result of a calculation or a series of calculations based on the input node or nodes. The output is calculated as

$$\text{Output} = \sum_{i=1}^{m} w_i x_i + \text{bias} = w_1 x_1 + w_2 x_2 + \ldots + w_m x_m + \text{bias}, \tag{2.4}$$

where x_i is one input data value, while w_i, on the other hand, is a weight factor, a numerical value to multiply the input data value with. For example when using a triaxial accelerometer, x_i could be the acceleration value of a single axis at a certain time point. The bias is a scalar value that shifts the output the same amount regardless of the values of the input

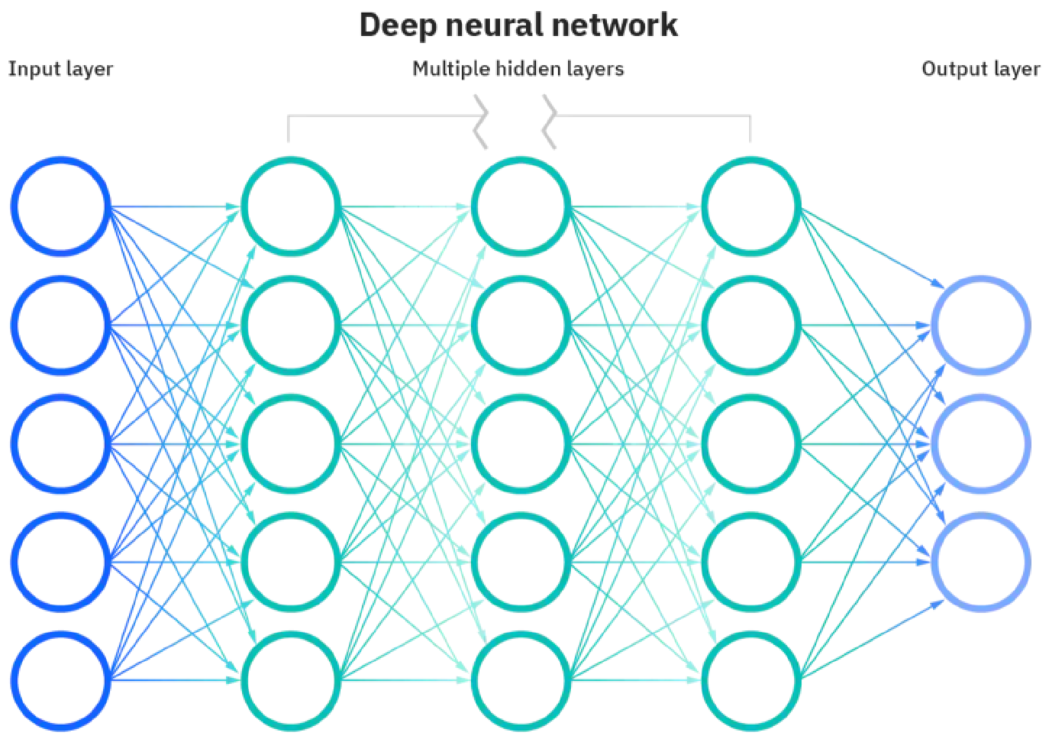


Figure 2.5. *The structure of a deep neural network[32]. The hidden layers apply weights on the input nodes and produce output nodes. The output nodes can then be used as the input for the next hidden layer. The output nodes of the final hidden layer are the final output, the output layer, of the neural network.*

data. The m in the sum function is the number of values in the input of the node. The data of the input layer is not always numerical, and in these cases the data needs to be converted to numerical values somehow. This can be done in multiple ways, for example characters can be converted to numbers according to rules that can then be backtracked at the end of the process.[32]

In deep learning, the AI learns the patterns that differentiate samples in different classes without human experts. This feature takes some of the burden away from humans and enables finding differentiating features that humans could not detect since the features can be very diverse and difficult to quantify. Proceeding from left to right in Figure 2.5, a given layer of the network uses the output of the previous layer, the layer on its left side, as its input. The layer then produces its output for the next layer to use as input, unless there is no next layer, in which case the output is the final result of the classification. The result is numeric, and it may need to be converted to another format for it to be easily interpretable for the user. For instance, all output values exceeding zero could be classified as falls, and all output values below zero as non-falls.

Both in supervised and unsupervised learning, the algorithm needs to be tested with data that was not used in the training stage. In the testing phase, input data is given to the algorithm, and the output class predicted by the algorithm is compared with the actual class that is known. If they match, the result for that sample is correct and if they do not, the result is considered incorrect. It is important that the algorithm is trained and tested with a big amount of data. Having more training data reduces the formation of biases such as overfitting, and therefore more data usually means reaching a more general classifier that works better in different types of falls. Overfitting means that the algorithm finds and exploits some details in the training data to reach a very high classification accuracy for the training data, but fails to reach a high classification accuracy with test data that is not part of the training data. This happens because the model puts too much weight on details that may only be present in the training data samples but not necessarily in the test data, thus leading to poor performance.

Machine learning algorithms are often referred to either as black box AI or white box AI. The key difference between these models is that in a white box AI model, the algorithm of the model is visible for its user and the logic of the algorithm can be understood[33]. On the contrary, black box AI models create their conclusions without the end-user knowing the logic behind them. Black box AI models can be more complex and thus potentially more accurate compared to white box AI models because the logic does not have to be easily understandable. The downside is that the users of black box AI may not fully trust the system since they do not know how it actually works. Using white box AI offers transparency and increases the users' trust. The added transparency makes white box AI a better choice for highly regulated industries.

According to Deutsch and Burgsteiner[34] the direction of movement is not needed for detecting falls qualitatively (i.e., determining if a fall has taken place or not). In their study, the absolute value of acceleration as defined in Equation (2.1) was enough to detect falls based on triaxial accelerometry data gathered from a smartwatch at a rate of 50 Hz. The absolute value of acceleration effectively measures the magnitude of acceleration. Deutsch and Burgsteiner used windowed data of the 7 previous seconds from each time point for developing their algorithms. In the preprocessing phase, the system measured the maximum acceleration deviation from the resting potential (i.e., 1 g because of gravitational acceleration) in the data frame. After this, more features were extracted from a data frame of 3 seconds, which covered 1.5 seconds before and after the peak acceleration time point. The researchers then proceeded with three different approaches to compare the classification results. The best classification result was achieved by extracting multiple features from the 3-second data frame to be used as the input nodes for training an artificial neural network. The neural network used 8 input neurons, which are listed below.

1. Maximum acceleration deviation from the resting potential
2. Minimum acceleration value before the maximum acceleration deviation

3. Fall intensity measured by the acceleration difference between the impact and the end of the fall
4. Impact duration measured by the time between the acceleration peak and the end of the fall
5. Fall duration
6. Mean value of the 3-second data frame
7. Average value of the 3-second data frame
8. Acceleration difference between the starting point and end point of the fall

One of the two alternative options used only 5 input neurons, which were items 1–5 of the list above. The other alternative method used all the samples of the 3-second data frame as its input neurons. This meant 150 input neurons because of the 50 Hz sampling frequency. The networks had only 1 hidden layer, and the number of neurons in it was experimented with different values, but 10 neurons was the number that was chosen in the end because of the performance. Each algorithm had 10 output neurons, each of which represented the predicted probability that the input was caused by a certain event, such as a fall to the left. The class with the highest output value was determined as the classification result. For a fall to be reported, the highest output value had to be at least 89% and it had to be for one of the fall types: forward, backward, left or right.

Deutsch and Burgsteiner[34] chose to use a threshold value of 1.7 g for the total acceleration. Only if the total acceleration exceeded the threshold, the smartwatch started to send data to a smartphone that was connected with the watch via a bluetooth connection. The sent data included data from the previous 6 seconds of the time point where the acceleration exceeded the threshold and 30 seconds after the peak. The effect was that the runtime of the smartwatch battery increased by at least 180%, which is a substantial increase.

The solution by Deutsch and Burgsteiner[34] had a safeguarding method in case the fall detection algorithm left a fall undetected, because their system could detect also suspicious inactivity. For example, if the user could not trigger an alert manually after a fall, a report of unusual inactivity would be sent after at least 30 minutes of suspicious inactivity. The authors stated that customising the fall detection algorithm for each user could improve its results. In the setup phase, data from daily life activities could be gathered and then used in conjunction with data from pre-recorded falls for training the fall detection neural network of the device. The result would be a personalised fall detection algorithm. The sensitivity and specificity of the algorithm could most likely be further improved by gathering data of falls of the actual user. It is understandably not realistic to do it, because actual falls are very dangerous for certain patient groups. Thus, pre-recorded fall data from test subjects is the main source of fall data for training a classifier. An additional beneficial feature that

Deutsch and Burgsteiner pondered is that the user could provide input for training the neural network further. In the event of a false alarm, the user could press a button to dismiss the alarm. This would make it possible to use the false alarm data as additional training data labelled as activities of daily life.

3. MATERIALS AND METHODS

In this work, the main materials that were used included both devices and software. The fall detection system will be integrated as part of an activity tracking system that will have other monitoring functionalities as well, such as indoor positioning. This provided a framework for the system. The system has four main components: a tag, an anchor, a gateway and a cloud platform. The scope of the monitoring system is limited to indoor areas where the system has been setup. After getting the devices and software setup correctly, fall tests were carried out and their results analysed.

3.1 Devices

The main electronics components, which are a tag, an anchor and a gateway are presented below in Figure 3.1. A tag refers to an integrated circuit (IC) board which will be located inside the wristband that users will wear. Its purpose is to gather acceleration data and send it to a set of anchors wirelessly. The anchors are IC board units that will be mounted on the walls of the rooms where the system is needed. The purpose of the anchors is to communicate both with all the tags in their range as well as with a gateway. The anchors can also be used for location estimation together with the tags. The anchors will send data to a gateway wirelessly. The gateway consists of two main parts: an IC board that is similar with the tags and anchors, and a Raspberry Pi microcomputer (Raspberry Pi Foundation, Cambridge, United Kingdom). The microcomputer can read input data coming from the IC board through a universal asynchronous receiver transmitter (UART) connection, run Python for running the fall detection algorithm and transfer information to a cloud platform via an internet connection. The cloud platform serves many purposes. The platform can store data for further analysis as well and, importantly for the scope of the thesis, send out alarms for the caretakers when the monitoring system detects a fall.

The accelerometer model that was used in the project inside the tags was a digital MEMSIC MC3419 3-axis accelerometer (MEMSIC Semiconductor Co., Ltd., Tianjin, China). It is a small device designed to be used in cell phones and other consumer products for sensing acceleration and motion. This type of triaxial accelerometer was chosen because the goal of the bigger project is to integrate the accelerometer in a wristband that will be used by patients. The acceleration range can be chosen between ± 2, ± 4, ± 8, ± 12 and

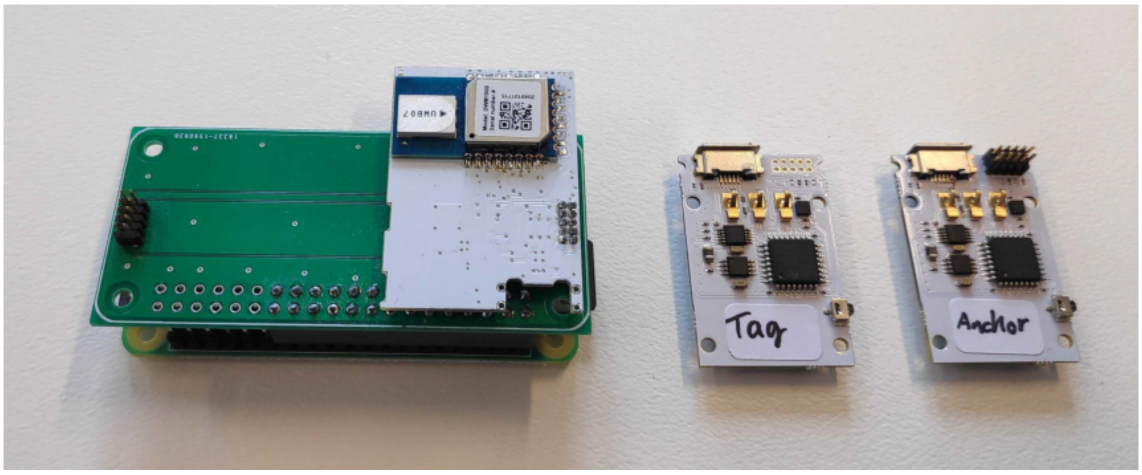


Figure 3.1. *The main electronics components of the tracking system: a gateway (left), a tag (middle) and an anchor (right). The tag will be mounted inside the wristband that the users will wear. It will send data to anchors mounted on walls and the gateway will gather data coming from multiple anchors in the final system.*

$\pm$16 g. The accelerometer produces triaxial data with a sample rate of 0.5 to 1000 Hz. The single sample resolution is 16 bits. This means that if the range is set bigger, the smallest detectable difference in acceleration is higher than when using a smaller range. The detailed specifications of the accelerometer are found in the device datasheet [35]. The chosen accelerometer can detect tilt and flip in addition to pure acceleration. The standby current is approximately only 4 µA, which makes the device suitable for ultra-low power applications.

The MC3419 is based on microelectromechanical systems (MEMS), and is therefore called a MEMS accelerometer. The working principle of the MC3419 accelerometer is that the accelerometer has a free mass that will move inside a housing along a single axis in presence of acceleration in the direction of the given axis. Figure 3.2 shows the basic structure of a MEMS accelerometer with variable capacitance for a single axis. The yellow electrodes, labelled Fixed Electrode 1 and 2 in the figure will not move in relation to the Fixed Support, the pink outer housing in the figure. The force required to move the proof mass a certain type depends on the attachment of the proof mass and the fixed support. The attachment can be executed with a set of springs, the features of which are known. The amount of movement is measured indirectly by measuring the change in capacitance. In equation

$$C = \epsilon_0 \cdot \frac{A}{d}, \tag{3.1}$$

C is the capacitance in F (Farads), ϵ_0 is the electric constant ($\epsilon_0 \approx 8.854 \cdot 10^{-12}\,\mathrm{F} \cdot \mathrm{m}^{-1}$), A is the overlapping parallel area of two capacitor plates in m^2 and d the distance between the plates in metres. As can be seen in Equation (3.1), the capacitance of a capacitor is

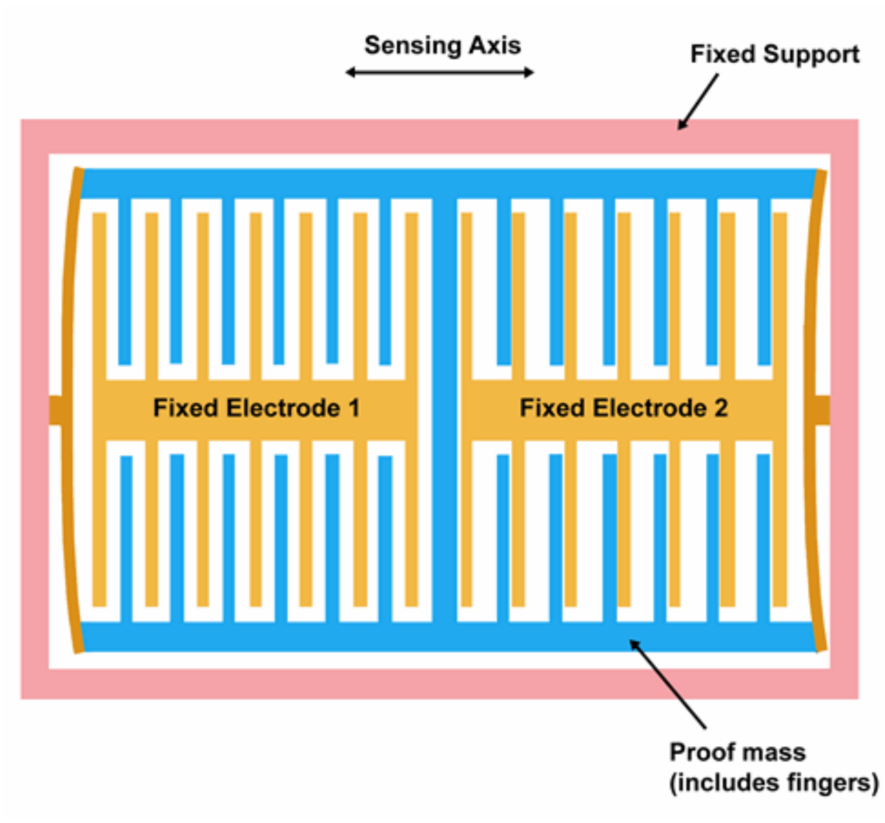


Figure 3.2. *The basic structure of a MEMS accelerometer with variable capacitance[36]*

inversely proportional to the distance between the capacitor plates. Using Equation (3.1), the change in capacitance can be used to calculate the change in the distance between the capacitor plates. The displacement is then converted to force based on the known features of the spring system. Based on the second law of Newton, $\mathbf{F} = m \cdot \mathbf{a}$, which can be converted to

$$\mathbf{a} = \frac{\mathbf{F}}{m}, \tag{3.2}$$

where $\mathbf{a}$ is acceleration in m$/$s^2, $\mathbf{F}$ is force in N and m is mass in kg. Now, both the mass of the proof mass and the force are known, and thus the acceleration can easily be derived. It is common to convert the acceleration to the gravitational acceleration of the Earth, g. $1\ g = 9.81$ m$/$s^2. With the MC3419, there are three orthogonal independent axes each of which calculate an acceleration value for their respective axes separately from each other but at the same frequency and phase.

The prototype IC board that was used in beginning of the study was connected to a computer via a USB cable. The accelerometer was soldered on an IC chip. Segger J-Link debug probe devices (Segger Microcontroller, Monheim am Rhein, Germany) were used to program the tags, anchors and gateway boards and to debug the software. The J-Links were connected to the tags, anchors and gateway via a dedicated connector and powered separately via a USB cable.

Decawave DW1000 transceivers (Decawave Ltd, Dublin, Ireland) were used to transmit and receive data via UWB radio frequency technology [37] between the tags, anchors and gateway. The term transceiver means that radio frequency signals can be both transmitted and received with the same device. The DW1000 requires a certain data format for its messages. The message consists of a preamble, a start frame delimiter, physical layer header and then finally the actual data that is wanted to be transmitted in the first place[37]. The default data frame length of the Decawave device allows the transmitted data to be up to 127 bytes (octets) long, while the extended length data frame allows for 1023 bytes. In the MC3419 accelerometer, the acceleration values of each axis are stored in 16-bit resolution[35]. In order to enable the tag to transmit messages with the acceleration values less frequently, the acceleration values of a certain number of previous time points can be stored in a buffer and then transmitted at once. The number of samples to be stored in the buffer can vary depending on the sampling frequency of the accelerometer and how often the values are chosen to be sent, which means that the maximum data frame length should be chosen accordingly.

An STM32L071KZT6 microcontroller (STMicroelectronics, Geneva, Switzerland) was integrated as a part of the IC. The STM32L071KZT6 is a microcontroller designed for ultra-low-power applications. Since a wearable device needs to be rather small for it to be convenient, it cannot have a physically big battery, which means that the capacity of the battery is also limited. Therefore the device needs to have a low power consumption. The microcontroller plays a part in decreasing the power consumption of the whole IC board. One important aspect in reducing the power consumption is that the microcontroller can be put to a sleep state when it is not needed, and then waken up quickly when needed. The purpose of a microcontroller is to control processes according to programmed instructions. A microcontroller switches between tasks quickly, but is only executing one task at a time. A microcontroller needs to have a central processing unit (CPU) to process instructions coming from computer programs. The clock frequency of the CPU found in the STM32L071KZT6 can be set from 32 kHz to a maximum of 32 MHz. The STM32L071KZT6 requires a power supply of 1.65 V to 3.6 V, and works in a temperature range of -40 to 125 °C. The wake up time of the microcontroller is 5 μs. [38]

In the final product the tag will be integrated to a wristband. The first prototype of the casing is presented in Figure 3.3. As can be seen, the wristband is similar in size with a smartwatch or smart bracelet. In this version the electronics were yet to be integrated in the wristband, since the IC boards and batteries used in the development phase had a slightly different shape and size than in the design of the final product, and thus they did not fit inside the casing. The device that was used for the first fall tests was just the tag with a battery attached to it. This combination was then gripped in the hand when falling. This rather inconvenient approach was chosen because the prototype casing had not yet

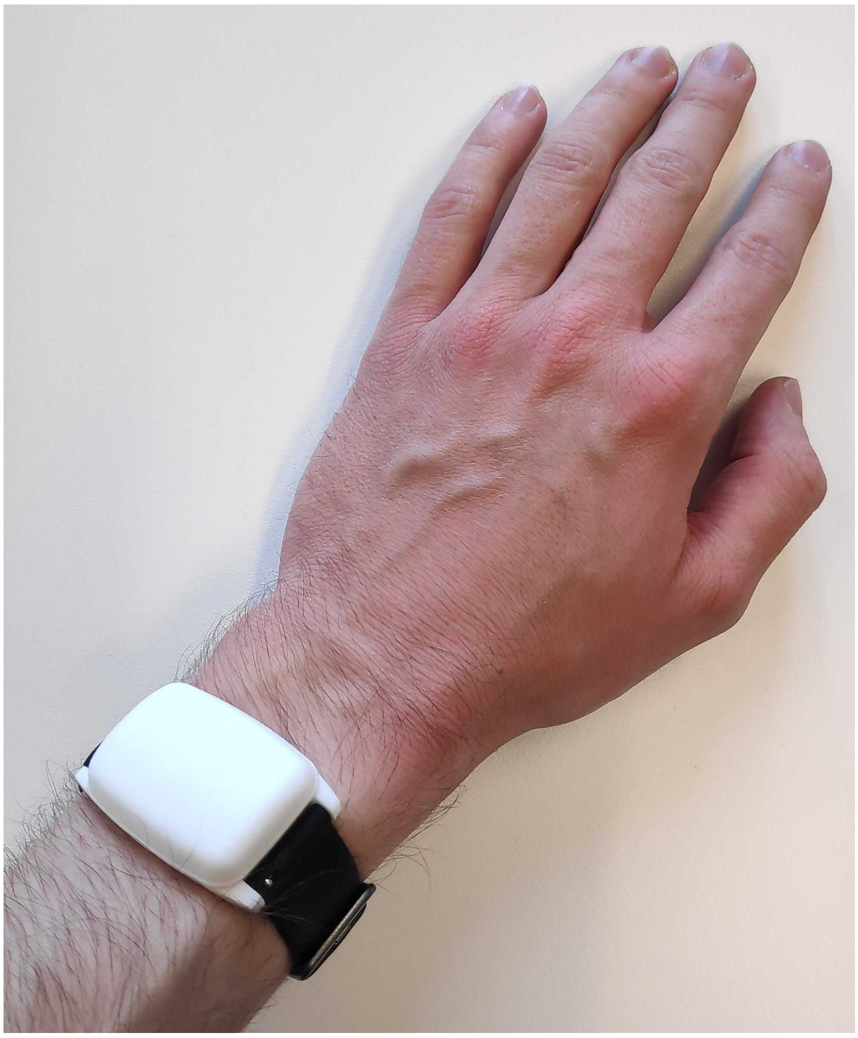


Figure 3.3. *The first prototype of the wristband where the tag will be integrated. The final design can change, but the basic concept and form factor is visible in the prototype.*

been received at the time of the first test. After the prototype wristband had arrived, a tag and a battery could be attached to the wristband. This prototype is shown in Figure 3.4.

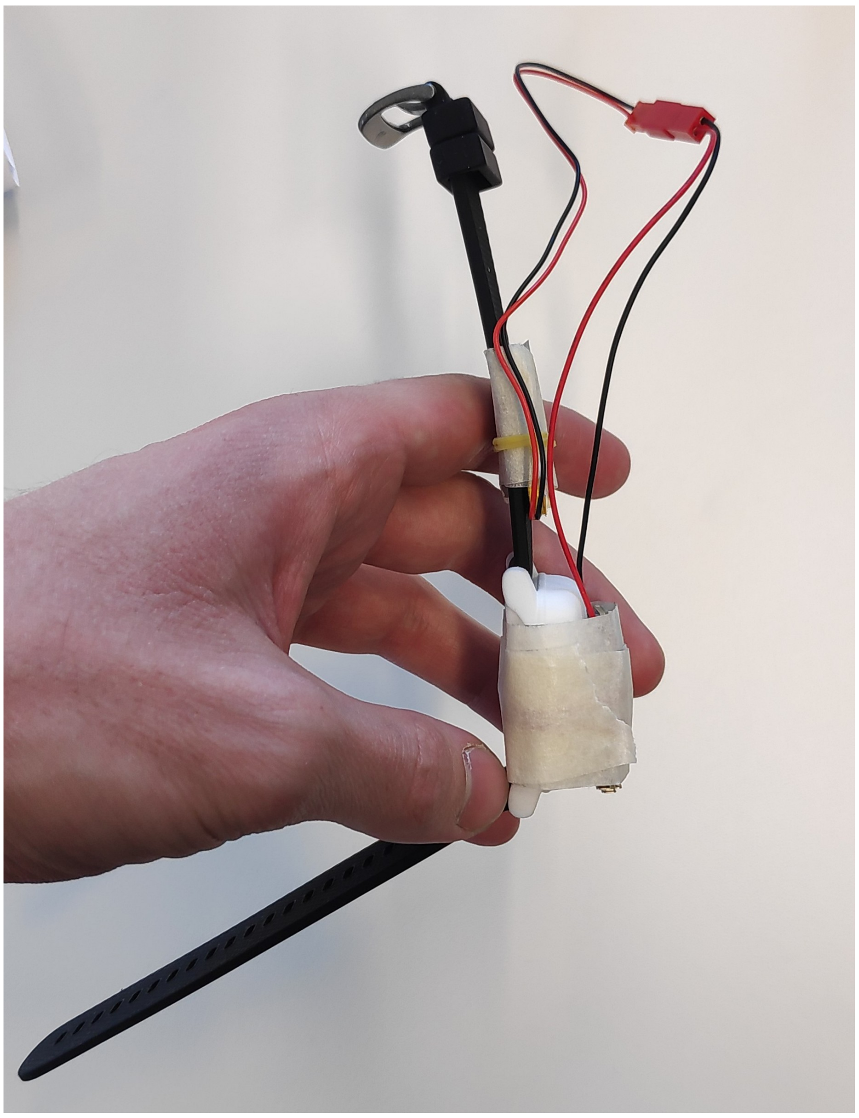


Figure 3.4. *Prototype wristband with a tag and battery attached using tape and a rubber band. This solution allowed collecting data from the wrist instead of from the hand as in the very first approach.*

All of the devices used were prototypes, which made them prone to faults and unexpected behaviour. Situations where unexpected behaviour occurred were investigated to decrease the likelihood of them happening again.

3.2 Software and Algorithms

Alongside the microcontroller, STM32CubeIDE from the same company was used as the integrated development environment (IDE). The IDE can be used on multiple operating systems, which enables cross-platform development. The STM32CubeIDE is designed for programming in C and C++ languages. The C language[39] was used for programming the microcontroller.

SEGGER J-Link V7.60e was used to control the Segger J-Link debug probe that was introduced in Section 3.1. The Segger J-Link debug probe was connected as a part of the IC to aid in debugging the source code and promoting software development. The debug probe communicates with the STM32CubeIDE and enables the debugging functionalities in the IDE. The debugger allows the programmer to set breakpoints in the source code when debugging the code. Breakpoints stop the execution of the source code on the row where the breakpoint is set. The values of variables in the code at the breakpoint can be examined when the code execution is halted. The programmer can then continue the execution of the program when they want. This makes it easier to decipher what exactly is happening in certain parts of the program and, consequently, find flaws in the code.

Tera Term[40] open source software was used as the terminal to print the measurement data to the screen for the user to examine. Tera Term allows the user to log data that is printed to the terminal for later analysis. The software was used to see and save the data that was printed on the screen in the debug mode.

For visualising the data and running the fall detection algorithm, Python 3[41] programs were used. The programs utilised external Matplotlib[42], NumPy[43], Pandas[44] and PySerial[45] packages. Jupyter Notebook[46] was also used in the visualisation process due to its user interface. Visual Studio Code version 1.63.2 (Microsoft Corporation, Redmond, Washington, United States) was used as the IDE due to its convenient functionalities.

3.3 Practical Tests

The first goal of the practical tests was to test that the device was working as intended, meaning that the measurement results were credible and that the results were transmitted successfully wirelessly. The first tests were done by having a tag with an accelerometer connected to a computer with a USB cable, and the data was printed by the tag to the Tera Term terminal. Because having a cable connected to the tag would cause troubles in conducting actual fall tests, and the final product needs to be wireless, there was a need to get the tag to transmit the information to an anchor. To do this, the measurement results needed to be inserted to the messages that the tag transmitted in a binary format. The first phase was to read the raw data coming from the accelerometer and include the whole data to the transmitted message. It is both time-consuming and increases power consumption to constantly send the data. Instead, the approach was to read 30 samples of the acceleration data to a buffer before then transmitting the data at given time intervals. Transmitting the 30 samples at once required extending the DW1000 data frames to a non-standard mode that could support a maximum data frame length of 1023 bytes.

In the beginning, a sampling frequency of 100 Hz was used. Having a high sampling frequency means more data, which could potentially make fall detection easier. However, the 100 Hz sampling frequency did not function as intended when the data was read from the accelerometer. The structure of the program made the rest of the program of the tag stop at the point where data was read to the buffer from the accelerometer. When using the 100 Hz sampling frequency the reading of the samples took more time than the buffer represented, which lead to a loss of a significant portion of the data. This was not acceptable in this kind of application where missing samples could cause significant danger because of undetected falls. Debugging the program and finding the point that caused the delay was challenging, because setting breakpoints on different lines affected the execution time that was printed on screen. After finding the root cause, the sampling frequency was reduced to tackle the issue. With a 15.625 Hz sampling frequency, the reading of 30 of samples from the accelerometer was quick enough to not miss any samples. 15.625 Hz was chosen because of an article by Huynh and Tran[47] demonstrated that most of the acceleration in both falls and daily activities is in the frequency range significantly below 15 Hz, and the accelerometer could be easily set to have a sampling frequency of 15.625 Hz. This gave confidence that the significantly reduced sampling frequency would still be high enough to detect falls and other activities.

The transmitted data was received wirelessly by an anchor that was run in debugging mode, and the data received data was printed to the Tera Term terminal. The prints in the terminal were saved in a comma-separated values (CSV) file for further examination. The end goal was to make a real-time fall detection algorithm, but for beginning algorithm development it was enough to save the data and then study it afterwards with Python. Visualising data is convenient with Python due to the availability of powerful mathematics libraries that allow plotting different types of plots with relatively little effort. Python was used to plot the acceleration vs. time graphs for each of the three axis of the accelerometer. In addition, the total acceleration absolute value as in Equation (2.1) was also calculated and plotted with Python at first.

After testing that it was possible to fit the absolute value in the message transmitted by the tag, the approach was changed to do the calculation of the absolute value already at the tag and then transmit the results alongside the raw accelerometry data. In general, algorithm development in this case does not depend on the programming language, since the actual algorithm logic and calculations are the same regardless of the programming language. However, Python has a host of features and libraries for data analysis, visualisation and computing, and the syntax of the programming language is rather simple. These features make Python a more convenient language than C for developing an algorithm. Due to this, a decision was made to do the initial algorithm development with Python and then at a further stage decide what operations could be done before sending any data all the way to the Raspberry Pi microcomputer.

The less data that needs to be transferred, the smaller the overall power consumption of the system will be. Optimising the power consumption of the system is crucial in the long term because the system is mostly battery-powered. Power consumption optimisation is not the focus of this study, and therefore it is not discussed in detail in the thesis. It still had

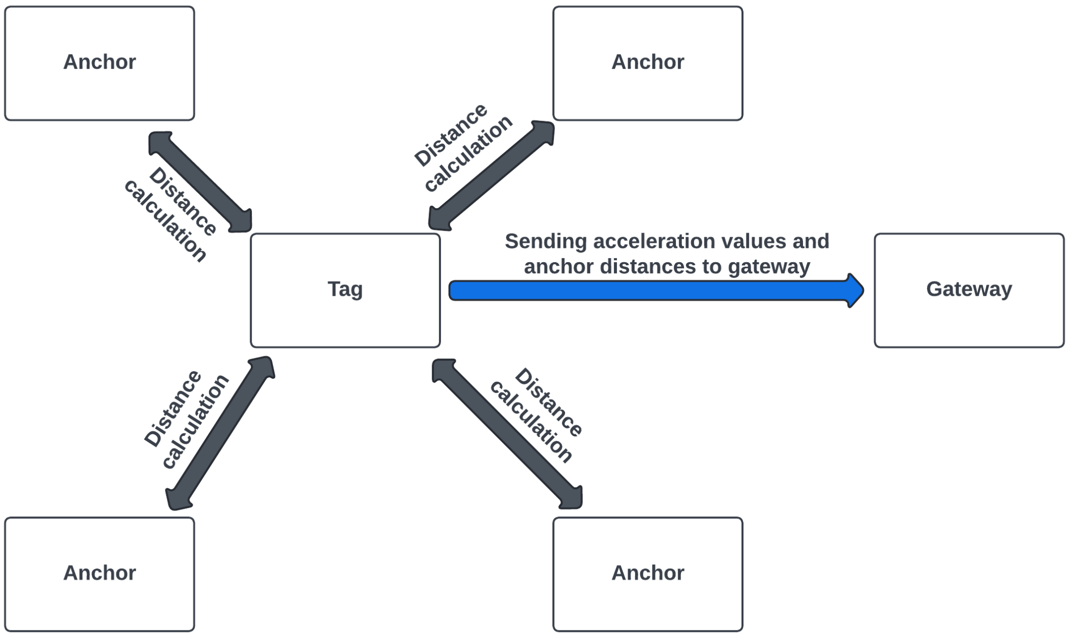


Figure 3.5. *The flow of data in the fall detection system office setup. The tag communicates with four anchors, all of which send the calculated distance betweeen the tag and anchor to the tag. The tag fetches a set of acceleration values from the accelerometer inside the tag. The tag then sends both the acceleration values and the calculated distances to the gateway.*

an impact on some of the software design choices of the thesis work because it is critical in the final application. Decreasing the number and length of messages sent between the devices reduces the power consumption. The length of the messages could be decreased by programming the tag to transmit only certain features of the acceleration signal such as described by Deutsch et al[34]. In these practical tests, the raw accelerometry data was still transmitted to the gateway because the data was used in the algorithm development process.

After successfully plotting the data, a few preliminary fall tests were carried out to get a rough idea about how the acceleration plots would look like in the event of a fall. The first tests were done with the tag and battery held in the hand of the test subject. The falls were lateral falls cushioned by a sofa. It was a challenge for the test subject to fall naturally because the tag was held in their hand and it had to be handled rather delicately. At this point the device did not have a wristband, but instead it only had the circuit board with the necessary electronics components and a battery.

After the first tests including only acceleration data, an office setup was planned. The office setup includes one tag, four anchors and one gateway. The flow of data in the system is presented in Figure 3.5. The functioning logic of the system is presented below.

1. The accelerometer inside the tag is measuring acceleration constantly at a frequency

of 15.625 Hz.

2. The accelerometry data is fetched from a buffer between every 30 samples.
3. The tag is told to communicate with multiple anchors and do a ranging process with them.
4. Once the distance between the tag and each anchor is calculated, each anchor sends the calculated distance to the tag.
5. Once the tag has received the distance values from all anchors, the tag sends the acceleration data from the previous 30 time points and the calculated distances to the gateway.
6. The gateway circuit reads the message from the tag and transfers it to the connected Raspberry Pi microcomputer via a UART connection.
7. A Python program running on the Raspberry Pi parses the data received via the UART connection.
8. The parsed data is saved to CSV file for further analysis and fed into a preliminary real-time version of the fall detection algorithm.
9. The preliminary algorithm prints an alarm on the screen if it detects a fall. Reaching the result takes multiple message cycles, because the preliminary algorithm analyses the data also after a potential fall before giving the final output.

The X- and Y-axes of the positioning system were horizontal axes along the floor level, and both of the axes were set to be in parallel with different walls of the office for the purpose of convenience. The Z-axis is vertical. All three axes are orthogonal. One corner of the office was chosen as the origin of the coordinate system, and the position of each anchor was measured relative to the origin. The anchor positions were then fed to the fall detection system initialisation program in metres to get the output of the tag position estimation in the same coordinates.

The goal of the office setup was to collect test data from simulated falls done by multiple test subjects. The collected test data could then be utilised to test and develop the fall detection algorithm. The data collected from fall tests was saved for further investigation and comparing fall detection algorithm performance when using different parameters. Investigating the similarities and differences in fall data between test subjects and different fall types made it possible to search for new, additional features that could separate falls and non-falls from each other. If new features that were relevant for fall detection were found, they could be incorporated into the fall detection algorithm to improve the algorithm performance, measured by sensitivity and specificity values.

To test the performance of the fall detection system, a dataset including both falls and non-falls was collected. There were 5 test subjects, all of whom were healthy and able to do the tests without impairment. Four different types of falls tests were done: falling

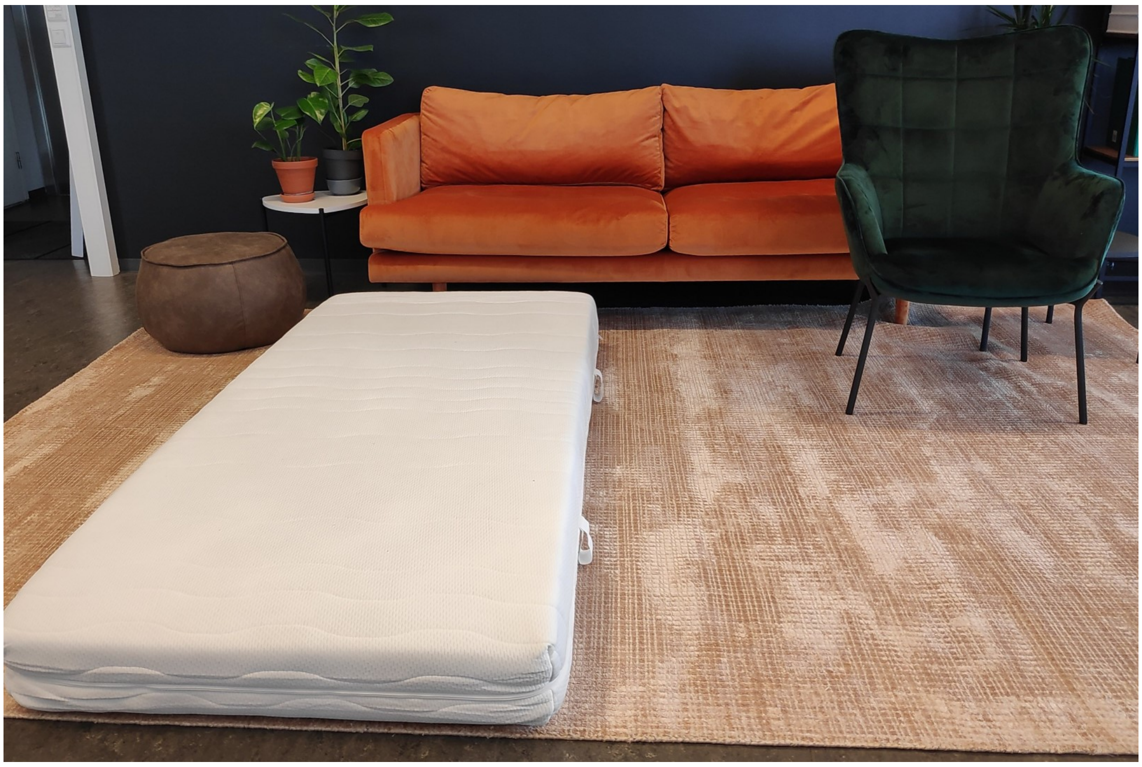


Figure 3.6. *A foam mattress, a sofa and a chair that were used in data acquisition. The mattress was used to cushion falls, while the sofa and chair were used to collect data of sitting down and getting up.*

forwards, backwards, to the left and to the right. All fall tests were done from a standing position and cushioned by a 22 cm thick foam mattress that was positioned on the floor. In addition to fall tests, certain activities of daily life were done and the data recorded. The daily activities included walking, sitting down to a sofa and chair, as well as standing up from the chair and sofa. The mattress, sofa and chair that were used are presented in Figure 3.6. The sitting height of both the sofa and chair were close to each other, but the chair was firmer than the sofa.

Both the fall tests and activities of daily life tests were recorded with a smartphone camera to make it easier to pinpoint which datapoint represents which event. Some preliminary tests including both falls and activities of daily life were done before collecting data that was used in evaluating the performance of the final algorithm. The data from these tests was not considered as a part of the dataset that was used in evaluating the algorithm performance. Each test subject was given the same basic set of instructions before the tests. The test subjects were free to choose which wrist they held the wristband on. 4 out the 5 test subjects chose to wear it on their left wrist. In the first part of data collection, the fall test were carried out as presented below.

1. Start data collection and video recording.
2. Shake wrist

3. Fall backwards, lay still for a short time and get back up. Repeat three times.
4. Fall forwards, lay still for a short time and get back up. Repeat three times.
5. Fall to the left side, lay still for a short time and get back up. Repeat three times.
6. Fall to the right side, lay still for a short time and get back up. Repeat three times.
7. Shake wrist
8. Stop data collection and video recording.

The second part of the data collection included measuring the activities of daily life. The process is presented below.

1. Start data collection and video recording.
2. Shake wrist
3. Walk around for 45 seconds.
4. Sit down on the sofa, and get back up after a few seconds. Repeat three times.
5. Sit down on the chair, and get back up after a few seconds. Repeat three times.
6. Shake wrist.
7. Stop data collection and video recording.

After both phases had been done, the collected data was transferred from the Raspberry Pi microcomputer to a laptop to utilise its higher computing power for the analysis phase. The final algorithm was developed based on the collected data and was done to analyse previously collected data instead of real-time data. The real-time version of the final algorithm was decided to be implemented in the next stages of the fall detection development process.

3.4 Statistical Methods

To develop and evaluate the fall detection system, a set of statistical methods are required. These methods are commonly applied in science for characterizing and evaluating different types of models. As the features are commonly used, different methods can be evaluated on the same scale. Sensitivity, also known as true positive rate (TPR), is defined as

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{3.3}$$

where TP stands for the number of true positive cases (falls correctly detected as falls) and FN for the number of false negative cases (falls that were not not detected). Often the rate is represented as a percentage instead of a decimal value. Specificity, also known as true negative rate (TNR), is defined as

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}, \tag{3.4}$$

where TN stands for true negatives (non-falls correctly identified) and FP for false positives (non-falls classified as falls). Although it is important to detect all the falls, the high sensitivity should not come with the expense of specificity. Having a high specificity value is important, because if there are a lot of false positives, they cause an equal number of false alarms. This can be measured with the term positive predictive value (PPV), which is defined as

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \tag{3.5}$$

where the variables are labelled similarly as in Equation (3.3) and Equation (3.4). When the number of false positives increases compared to the true positives, the positive predictive value decreases. PPV measures the ratio of true positives against all the positive results (true positives and false positives). For example, if the positive predictive value is 0.50, only half of the positives cases are true. This is obviously not ideal. Negative predictive value (NPV), on the other hand, is defined as

$$\mathrm{NPV} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FN}}, \tag{3.6}$$

where the variables are labelled as in the previous three equations. The purpose of the negative predictive value is to measure the fraction of true negative cases out of all the cases that were classified as negative. Positive and negative predictive values are useful meters, because they tell the person interpreting the results what is the probability that the classification or test result is correct. The combination of sensitivity, specificity, positive and negative predictive value that can be considered good enough or excellent depends heavily on the application. In some cases, such as screening for certain diseases, having a lower PPV can be acceptable as long as the NPV is very high to keep the number of missed disease cases minimal. In the case of fall detection, as stated earlier, all of the four metrics should provide as high a value as possible. Reaching values of 100% is not realistic, but the closer, the better. However, in case a need for compromises arises, higher priority should be given to sensitivity over specificity. This is because undetected falls have worse consequences than false alarms.

A common method of measuring classification performance of an algorithm is to create a confusion matrix. The basic layout of a confusion matrix is presented in Figure 3.7. In the final confusion matrix graph, the boxes labelled True positives, False positives, True negatives and False negatives are replaced with the number of samples in their respective categories. In an ideal confusion matrix, True positives would be equal with the true total number of samples in the positive category, and True negatives would be equal with the

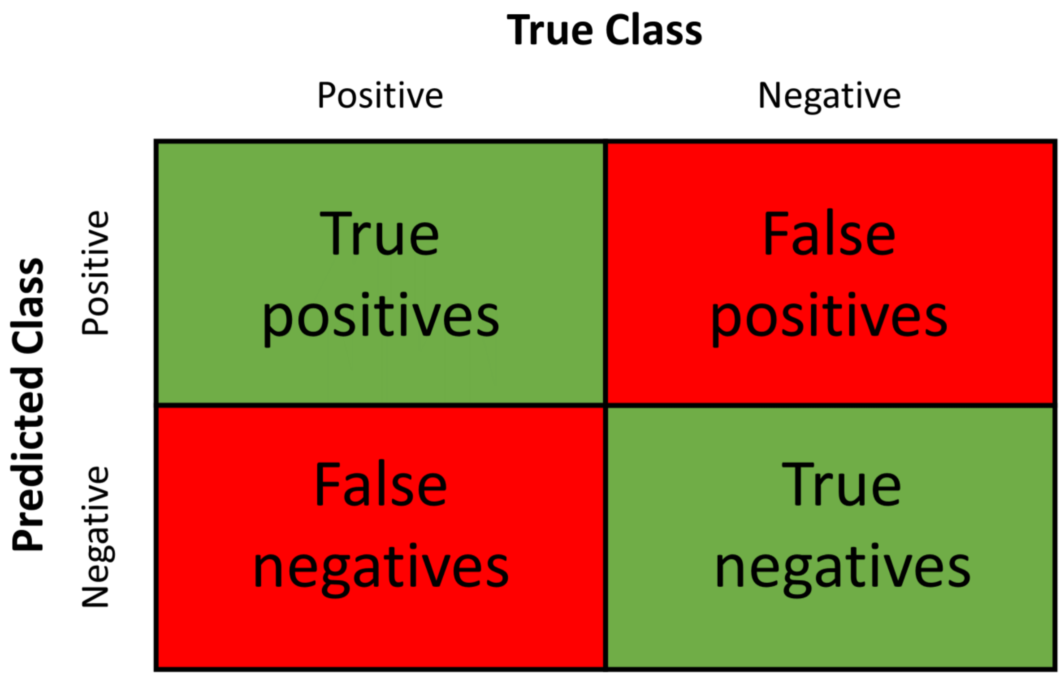


Figure 3.7. *The basic layout of a confusion matrix. The green areas are correct predictions and the red areas are false predictions. In an ideal predictive model the number of false predictions would be zero.*

true total number of samples in the negative category. Both False positives and False negatives would be zero. The matrix is a visual representation of the performance of the model and can be used in conjunction with sensitivity and specificity values as well as with positive and negative predictive values.

The confusion matrix can use colour coding, such as green and red with different saturation values, to visualize the performance better. It is important to note that the confusion matrix needs to be created using only test data and no training data as it would distort the results and most likely make the model seem better than it actually is.

Other common measures of classification performance include classification accuracy and error rate. Classification accuracy is defined as

$$\text{Classification accuracy} = \frac{\text{Correct predictions}}{\text{Total predictions}}, \tag{3.7}$$

where $\text{Correct predictions}$ stand for the sum of true positives and true negatives, and Total predictions include all predictions. Total predictions can be presented also in the form of
$\text{Total predictions} = \text{TP} + \text{TN} + \text{FP} + \text{FN}$. Error rate is related to classification accuracy simply as

$$\text{Error rate} = \frac{\text{False predictions}}{\text{Total predictions}} = 1 - \text{Classification accuracy}, \tag{3.8}$$

where False predictions is the sum of false positives and false negatives. If the results need to be displayed as a percentage, the results are multiplied by 100%.

4. RESULTS

In the visulisation phase the collected acceleration data and positioning data was plotted. The total acceleration calculated as in Equation (2.1) was plotted against the time axis. In addition to the total acceleration, the estimated height of the tag from the surface of the floor was plotted. An example plot including falls in four different directions as well as walking is presented in Figure 4.1. This data was collected as an example and was not a part of the dataset that was used in evaluating the performance of the fall detection algorithm. As can be seen, the falls caused distinct acceleration peaks. The Z-axis values, which represent the height position, had significant inconsistencies with the actual events.

The X position showed changes that were related to the fall patterns. Figure 4.2 shows how the estimated X position value increased after falls and decreased when the test subject got up from the mattress. The direction of the horizontal X-axis was approximately the same as the fall direction due to the orientation of the mattress. The change in X position from standing before the falls to lying on the mattress after the falls is approximately 1.5–2.5 meters depending on the fall.

In Figure 4.3, all the falls were detected and the wrist shaking was filtered out by the algorithm due to the repetitive nature of the acceleration peaks. The red dashed line is the chosen threshold value that needed to be exceeded, 2500 mg in this example, that the algorithm considered the point as a peak. Peaks that were closer than 15 sample indices from each other were considered as a single peak. After this filtering, the found peaks were plotted with the orange crosses. If two filtered peaks were closer than 30 sample indices from each other, the peaks were considered to be caused by repetitive movement, such as shaking, and not by a fall. The red triangles represent the falls that the algorithm detected. As can be seen, not all peaks were considered as falls.

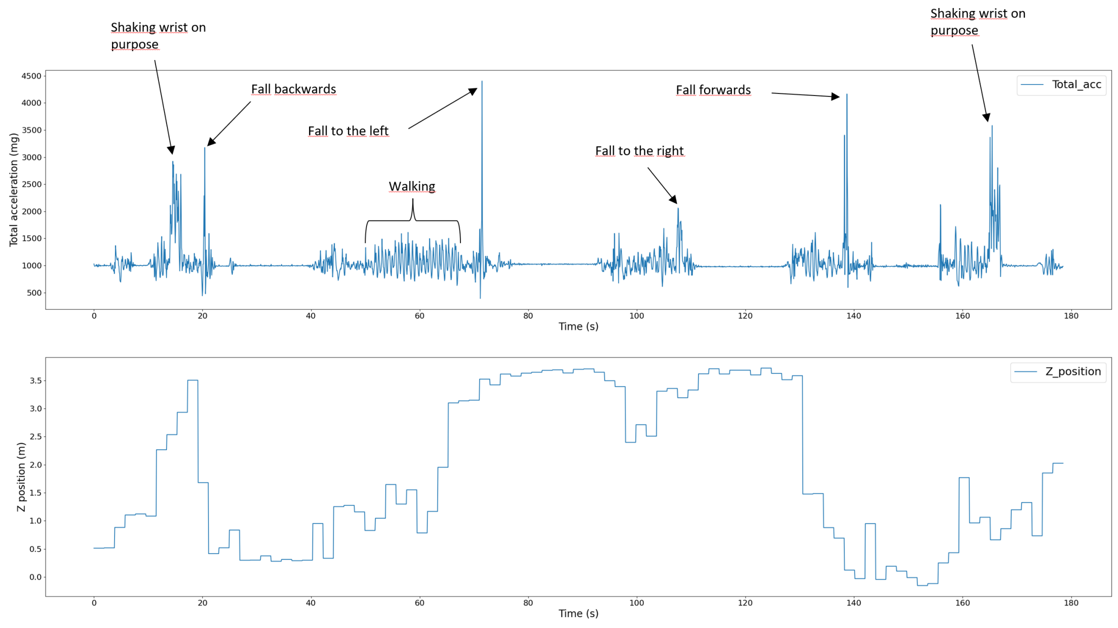


Figure 4.1. *Total acceleration plotted on the upper subplot and the estimated height, Z position, of the tag from the surface of the floor plotted on the lower subplot. The height estimation had significant errors.*

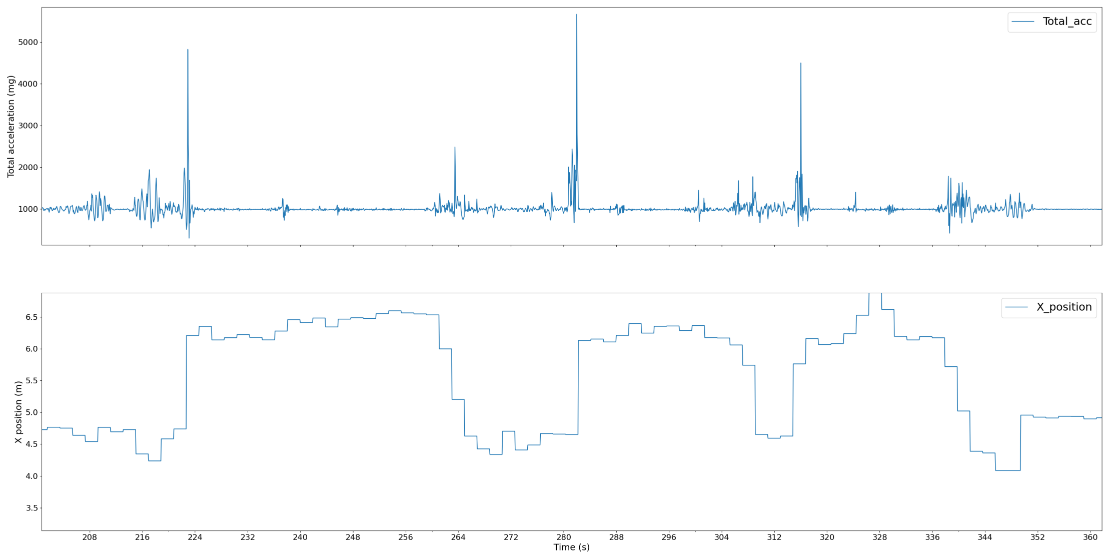


Figure 4.2. *Total acceleration plotted on the upper subplot and the X position on the lower subplot. The horizontal X position changed after the falls and after getting up from the mattress.*

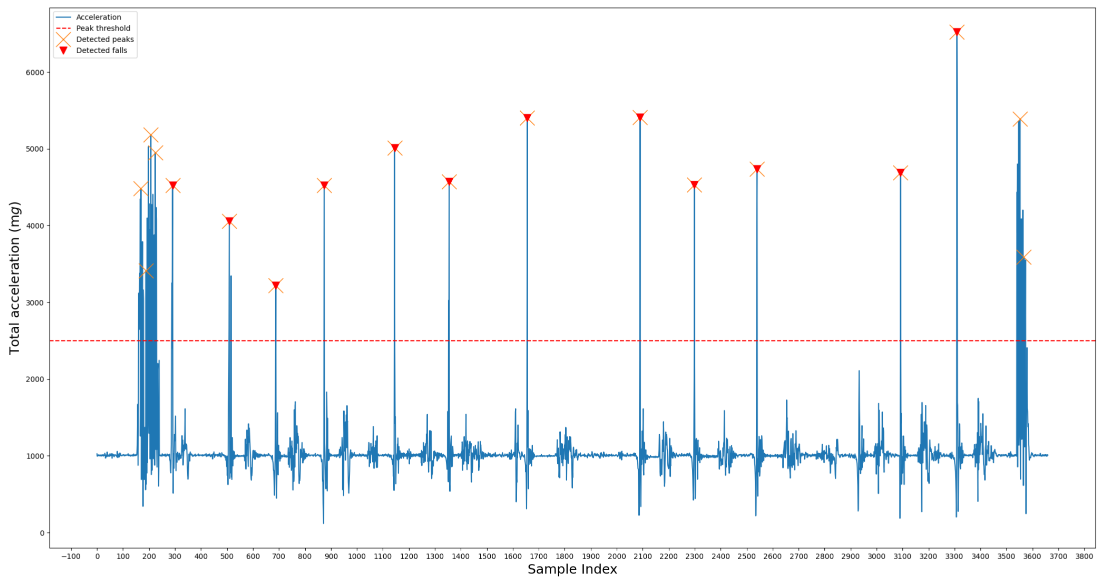


Figure 4.3. *The acceleration data of the fall tests of one test subject plotted against the sample index. The red triangles represent the detected falls from the data with the acceleration threshold value of 2500 mg. The fall detection algorithm detected every fall from this test subject.*

In Figure 4.4, the results of the fall dedection algorithm using the threshold value of 2500 mg are presented from the activities of daily life tests of one test subject. The fall detection system incorrectly labelled two non-falls as falls. The first false positive on the left side of the figure was caused by wrist shaking, while the second one on the right side of the figure was caused by sitting on the chair.

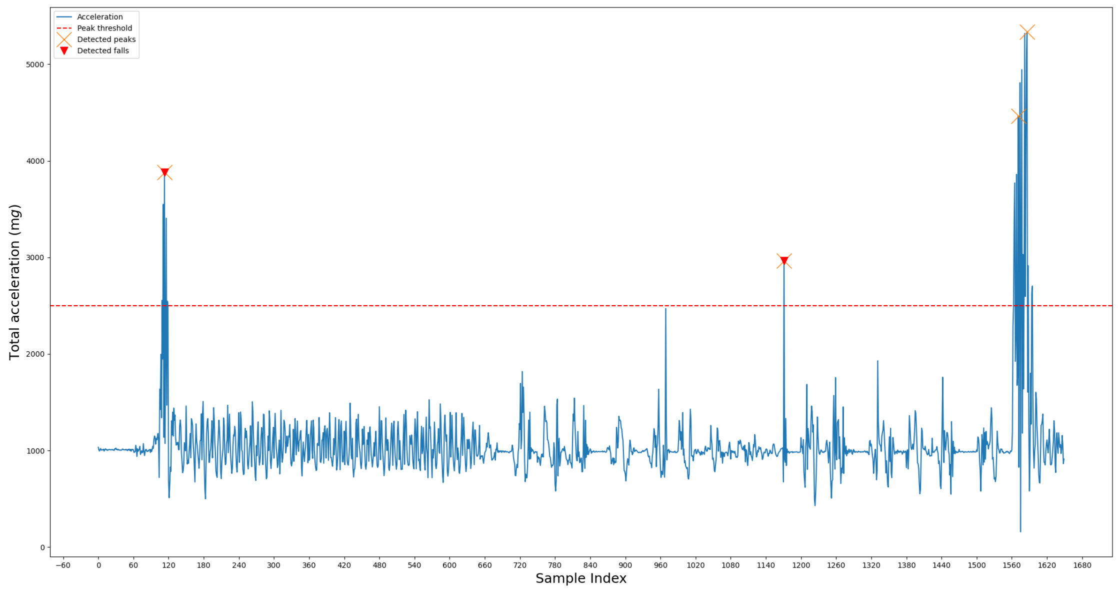
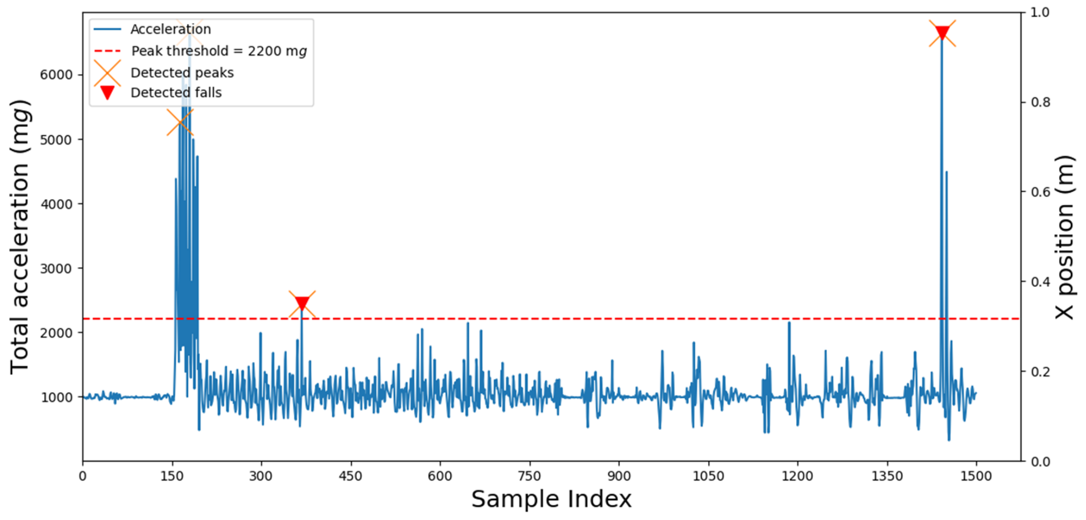


Figure 4.4. *The acceleration data of the activities of daily life tests of one test subject plotted against the sample index. The red triangles represent the detected falls from the data with the threshold value of 2500 mg. The fall detection algorithm detected two false positives.*

Figure 4.5 shows the effect of varying the acceleration threshold value on the specificity of the fall detection. Both graphs have the data of the ADL tests of the same test subject. With the 2200 mg threshold, there were two false positives as shown in the upper graph, but with the 2500 mg threshold value there was only one as shown in the lower graph. The false positive that was eliminated by increasing the threshold was caused by walking. The other false positive that is shown on the right side of both graphs was caused by the shaking of the wrist. None of the three tested threshold values could not eliminate this false positive.



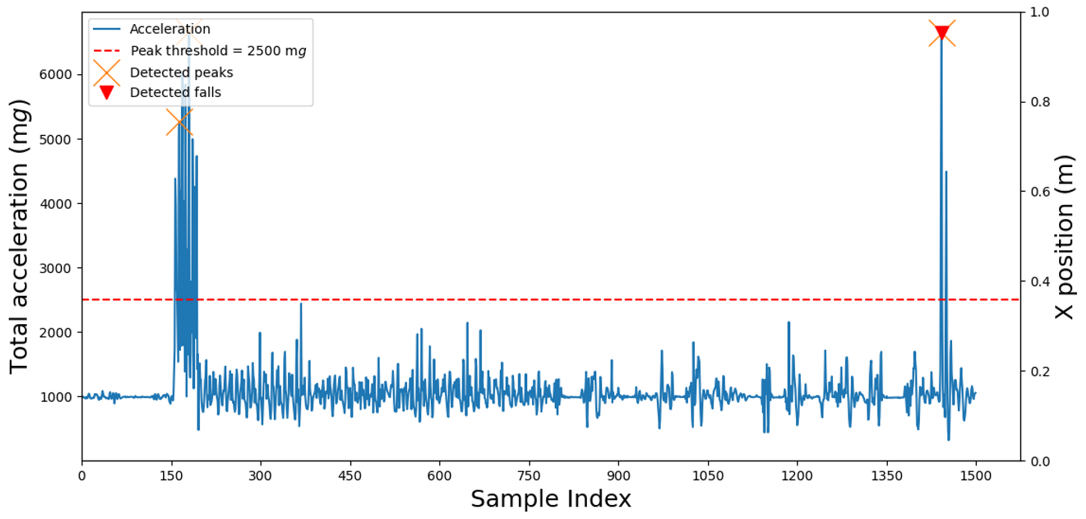


Figure 4.5. *The classification results of the ADL tests of a test subject using two different acceleration thresholds: 2200 and 2500 mg. The upper graph shows the results when the threshold was set at 2200 mg, and the lower graph shows the results with the threshold of 2500 mg. One of the two false positives was eliminated when the threshold was increased.*

Table 4.1 shows the performance of the fall detection system based on the collected data from 5 test subjects using 3 different acceleration threshold values: 2200, 2500 and 2800 mg. The algorithm that was used in calculating the results utilised only the acceleration data, namely the total acceleration as defined in Equation (2.1). The position data was collected simultaneously but was not used in the algorithm. The total number of falls was 61, which consisted of 13 falls backwards, 12 forwards, 12 to the left and 12 to the right. There were 13 falls backwards because one test subject unintentionally fell backwards 4 times instead of 3. In the analysis of activities of daily life, also known as non-falls, shaking the wrist on purpose was considered as a separate non-fall event that was taken into account in the specificity analysis. The total number of non-fall cases was 79, which consisted of 5 cases of walking, 15 cases of sitting on a sofa, 15 cases of getting up from

Table 4.1. *Performance of the fall the detection system in testing using 3 different threshold values: 2200 mg, 2500 mg and 2800 mg. The statistical values are calculated using the equations found in Section 3.4.*

Threshold value (mg)	2200	2500	2800
Total Number of Fall Events	61	61	61
Total Number of Non-Fall Events	79	79	79
True Positives	52	52	50
False Negatives	9	9	11
True Negatives	69	72	72
False Positives	10	7	7
Sensitivity	85.2%	85.2%	82.0%
Specificity	87.3%	91.1%	91.1%
Classification accuracy	86.4%	88.6%	87.1%
Error rate	13.6%	11.4%	12.9%
Positive predictive value	83.9%	88.1%	87.7%
Negative predictive value	88.5%	88.9%	86.7%

a sofa, 15 cases of sitting on a chair, 15 cases of getting up from a chair and 14 cases of the test subject shaking their wrist.

The thresholds of 2200 and 2500 mg produced a sensitivity value of 85.2%, while the threshold of 2800 mg produced a sensitivity of 82.0%. The lowest threshold value of 2200 mg produced the lowest specificity, 87.3%, while the two other thresholds both resulted in a specificity of 91.1%. The highest combination of sensitivity and specificity was reached with the threshold value of 2500 mg. With this threshold choice, 52 out of the 61 falls were detected, leaving 9 falls undetected. When the algorithm was tested with data from activities of daily life, the algorithm classified 72 out of 79 events correctly and produced 7 false positives. Therefore the classification accuracy with the threshold value of 2500 mg was 88.6%, leaving the error rate at 11.4%. The positive predictive value was 88.1%, while the negative predictive value was 88.9%.

5. DISCUSSION

Accelerometry has been used widely in fall detection. However, combining accelerometry with a positioning system is rarer. This multimodal method for fall detection would open up new possibilities compared to using only one type of data. Using accelerometry data alone for fall detection can make the system prone to a large number of false alarms for the reason that a lot of events other than falls can cause high acceleration peak values. Therefore using only accelerometry data is challenging. In a fall detection solution utilising a wristband, the free movement of the wrist produces high acceleration values in many situations, and for this reason the positioning data could provide valuable additional information for improved fall detection. However, during the thesis work the inaccuracies were too large to utilise the position data in fall detection. With additional studies and calibration the position data could become more accurate, which could allow it to be used in fall detection. Some of the positioning errors were most likely due to reflections of the UWB waves from surfaces like the walls, floor and furniture. Testing different positions for the anchors could decrease the errors. These tests could be carried out in the future systematically, but were left out of the thesis work.

Not utilising the position data made the fall detection algorithm more simple compared to doing the fall detection utilising both acceleration and position data. Even though the positioning data was not used in the final algorithm, it was still collected in the tests and can still be analysed more thoroughly. In this application the UWB technology needed to work even if the positioning data would not have been collected because the same technology was used to transmit data from the tag to the gateway.

Machine learning algorithms were not used in this study. This is because machine-learning algorithms are more complex and often computationally more heavy than threshold-based algorithms. In addition, machine-learning algorithms require vast amounts of data for training and testing. Having a threshold-based algorithm makes it easier to understand why an event is or is not considered as a fall. Thresholds are also quick to change, which makes it quick to change the properties of the algorithm. The possibility to do quick changes make the algorithm development process smoother because different parameters can be tested conveniently and the effect on the performance can be seen in a short time frame.

It is likely that the specificity values that were reached are partly misleading. The activities of daily life that were recorded were quite simple and they could not exhaustively mimic all the events that could happen in everyday life. If the fall detection system were to be held on for a longer period of time in a typical home or hospital environment, there could be a bigger proportion of false positives than what was found from the collected dataset presented in the thesis. Further testing would provide a more thorough picture of the performance of the system in settings that are closer to real life. Since falls are very dangerous to elderly people, it is crucial that any falls would not go undetected. The requirement of very high sensitivity remains a challenge when combined with the practical requirement of high specificity to avoid repetitive false positives.

An aspect that could be tested in the future is using a higher sampling frequency than 15.625 Hz in the accelerometer. The problem of missing samples with the sampling frequency of 100 Hz could most likely be overcome with different software design. The accelerometer hardware should not be a bottleneck with a sampling frequency of 100 Hz. Testing of higher sampling frequencies was left out because of time constraints.

The time interval between each position estimation was close to 2 seconds. It would have been possible to calculate the position more frequently from a software and hardware point of view, but it would not have been sensible because of the significantly higher power consumption. Because the position was determined by the tag and anchors transmitting and receiving multiple messages between each other, a higher positioning frequency would have meant more messages to and from the tag and anchors. The UWB transceiver played a major role in the overall power consumption of the tags and anchors. Thus, the transceivers were set to sleep mode when there were no messages to transmit or expected to be received. The sleep mode decreased the power consumption of the devices significantly. If the positioning was done at a higher rate, the benefit of sleep mode would be partly lost because the devices could not then be set to sleep mode for time periods as long as when using a smaller positioning rate. In addition, the software was easier to program by executing the ranging process always right before fetching the accelerometry samples from the accelerometer buffer.

From a software development point of view, unit tests and thorough testing in general were not done since the developed software was at a prototype stage. Unit tests and thorough testing are still required in the production stage of the software. Still, clear error sources (e.g., division with zero, negative distance values) were identified and handled accordingly to make the software more robust. A preliminary version of the fall detection algorithm was made to work with live data on a local Raspberry Pi microcomputer system. However, the final algorithm was implemented only for previously collected data instead of live data. This choice was made for the reason that in the future development versions the algorithm will be running on a cloud platform. Transferring the algorithm over to a real-time cloud-based implementation would have consumed a large portion of the time that was dedicated for doing the data analysis stage of the thesis. The fundamental fall detection logic of the algorithm will stay the same regardless of whether the data is live or read from a static file afterwards.

In general, the embedded software development involving multiple devices proved to be challenging and time-consuming. These challenges caused a delay in setting up the fall detection system in the office space. The challenges were mostly in the communication between the different devices of the system. Certain delay values had to be altered by using trial and error. Sometimes there were no actual errors in the software, but instead issues in powering the devices caused quite cryptic error messages that were not easily trackable to powering issues. After these issues were tackled, collecting data from falls and activities of daily living progressed quickly. However, there was less time for developing the fall detection algorithm than what was initially aimed for. This effectively meant that some aspects in the algorithm could not be developed further in the available time frame.

6. CONCLUSION

The thesis work was set out to implement a proof-of-concept level fall detection system based on data produced by a wristband. Both acceleration and position data were gathered for further analysis, but the position data had such large errors that it could not be utilised in the fall detection system. A fall detection system based on wristband accelerometry data was implemented. The devices of the fall detection system communicated successfully with each other. A threshold-based algorithm utilising accelerometry data with a frequency of 15.625 Hz was created and tested. The main result of the thesis is a functioning wireless fall detection system that utilises accelerometry data from a wristband. The performance of the algorithm was promising for further development, since the fall detection sensitivity of 85.2% and the specificity of 91.1% were achieved. Future research on accurate indoor positioning is needed to enable utilising the position data in fall detection systems to make them more accurate. Different values for the parameters of the acceleration analysis algorithm also need to be systematically tested to find an optimal balance between sensitivity and specificity.

REFERENCES

[1] World Health Organization, *Falls*, 2021. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/falls (visited on 01/24/2022).

[2] M. Mänty, S. Sihvonen, T. Hulkko, and A. Lounamaa, "Iäkkäiden henkilöiden kaatumistapaturmat: opas kaatumisten ja murtumien ehkäisyyn", *Kansanterveyslaitoksen julkaisuja*, no. B29, 2007. [Online]. Available: https://jyu.finna.fi/Record/jykdok.1066034.

[3] Statistics Finland, *Appendix table 1a. Deaths by underlying cause of death and by age in 2020, both sexes*, 2020. [Online]. Available: http://www.stat.fi/til/ksyyt/2020/ksyyt_2020_2021-12-10_tau_001_en.html (visited on 02/04/2022).

[4] A. B. A. Pinto, G. A. de Assis, L. C. B. Torres, T. Beltrame, and D. M. G. Domingues, "Wearables and Detection of Falls: A Comparison of Machine Learning Methods and Sensors Positioning", *Neural Processing Letters*, pp. 2165–2179, 2022. DOI: 10.1007/s11063-021-10724-2. [Online]. Available: https://doi.org/10.1007/s11063-021-10724-2.

[5] Y. S. Delahoz and M. A. Labrador, "Survey on fall detection and fall prevention using wearable and external sensors", *Sensors (Switzerland)*, vol. 14, no. 10, pp. 19 806–19 842, 2014. DOI: 10.3390/s141019806.

[6] M. Mubashir, L. Shao, and L. Seed, "A survey on fall detection: Principles and approaches", *Neurocomputing*, vol. 100, pp. 144–152, 2013. DOI: 10.1016/j.neucom.2011.09.037. [Online]. Available: http://dx.doi.org/10.1016/j.neucom.2011.09.037.

[7] M. Kangas, A. Konttila, P. Lindgren, I. Winblad, and T. Jämsä, "Comparison of low-complexity fall detection algorithms for body attached accelerometers", *Gait and Posture*, vol. 28, no. 2, pp. 285–291, 2008. DOI: 10.1016/j.gaitpost.2008.01.003.

[8] U. Korpilahti, R. Koivula, P. Doupi, and V. Jakoaho, *Safely at All Ages : Programme for the Prevention of Home and Leisure Injuries 2021–2030*. Ministry of Social Affairs and Health of Finland, 2021, ISBN: 978-952-00-5398-7. [Online]. Available: https://julkaisut.valtioneuvosto.fi/handle/10024/163155.

[9] K. J. Ruskin and D. Hueske-Kraus, "Alarm fatigue: Impacts on patient safety", *Current Opinion in Anaesthesiology*, vol. 28, no. 6, pp. 685–690, 2015. DOI: 10.1097/ACO.0000000000000260.

[10] S. Sendelbach and M. Funk, "Alarm fatigue: A patient safety concern", *AACN Advanced Critical Care*, vol. 24, no. 4, pp. 378–386, 2013. DOI: 10.1097/NCI.0b013e3182a903f9.

[11] L. Kowalczyk, *State reports detail 11 patient deaths linked to alarm fatigue in Massachusetts*, 2011. [Online]. Available: https://www.massnurses.org/news-and-events/archive/2011/p/openItem/7153 (visited on 03/11/2022).

[12] M. Görges, B. A. Markewitz, and D. R. Westenskow, "Improving alarm performance in the medical intensive care unit using delays and clinical context", *Anesthesia and Analgesia*, vol. 108, no. 5, pp. 1546–1552, 2009. DOI: 10.1213/ane.0b013e31819bdfbb.

[13] M. C. Chambrin, D. Calvelo-Aros, A. Jaborska, C. Chopin, P. Ravaux, and B. Boniface, "Multicentric study at monitoring alarms in the adult intensive care unit (ICU): A descriptive analysis", *Intensive Care Medicine*, vol. 25, no. 12, pp. 1360–1366, 1999. DOI: 10.1007/s001340051082.

[14] S. Siebig, S. Kuhls, M. Imhoff, U. Gather, J. Schölmerich, and C. E. Wrede, "Intensive care unit alarms—How many do we need?", *Critical care medicine*, vol. 38, no. 2, pp. 451–456, 2010.

[15] United Nations Department of Economic and Social Affairs, *World Population Prospects 2019*, 2019. [Online]. Available: https://population.un.org/wpp/Graphs/DemographicProfiles/Line/900 (visited on 02/03/2022).

[16] L. Schwickert, C. Becker, U. Lindemann, *et al.*, "Fall detection with body-worn sensors: A systematic review", *Zeitschrift fur Gerontologie und Geriatrie*, vol. 46, no. 8, pp. 706–719, 2013. DOI: 10.1007/s00391-013-0559-8.

[17] Apple Inc., *Use fall detection with Apple Watch*. [Online]. Available: https://support.apple.com/en-us/HT208944 (visited on 06/08/2022).

[18] L. Abou, A. Fliflet, L. Hawari, *et al.*, "Sensitivity of Apple Watch fall detection feature among wheelchair users", *Assistive Technology*, vol. 00, no. 00, pp. 1–7, 2021. DOI: 10.1080/10400435.2021.1923087. [Online]. Available: https://doi.org/10.1080/10400435.2021.1923087.

[19] Suvanto Care Oy, *Suvanto Mukana - turvalliseen liikkumiseen*. [Online]. Available: https://www.suvantocare.fi/turvaranneke-turvapaikannin/ (visited on 06/08/2022).

[20] S. B. Khojasteh, J. R. Villar, C. Chira, V. M. González, and E. de la Cal, "Improving fall detection using an on-wrist wearable accelerometer", *Sensors (Switzerland)*, vol. 18, no. 5, pp. 1–28, 2018. DOI: 10.3390/s18051350.

[21] ITU-R, "Recommendation ITU-R SM.1755-0: Characteristics of ultra-wideband technology", no. 2006, 2006. [Online]. Available: https://www.itu.int/dms_pubrec/itu-r/rec/sm/R-REC-SM.1755-0-200605-I!!PDF-E.pdf (visited on 02/13/2022).

[22] S. Kochkin, "MarkeTrak V: "Why my hearing aids are in the drawer": The consumers' perspective", *The Hearing Journal*, vol. 53, no. 2, pp. 34–41, 2000.

[23] L. Klack, C. Möllering, M. Ziefle, and T. Schmitz-Rode, "Future care floor: A sensitive floor for movement monitoring and fall detection in home environments", in *Wireless Mobile Communication and Healthcare*, J. C. Lin and K. S. Nikita, Eds., Berlin,

Heidelberg: Springer Berlin Heidelberg, 2011, pp. 211–218, ISBN: 978-3-642-20865-2.

[24] P. Leusmann, C. Möllering, L. Klack, K. Kasugai, M. Ziefle, and B. Rumpe, “Your floor knows where you are: Sensing and acquisition of movement data”, *Proceedings - IEEE International Conference on Mobile Data Management*, vol. 2, pp. 61–66, 2011. DOI: 10.1109/MDM.2011.29.

[25] MariCare Oy, *Elsi Smart Floor*. [Online]. Available: https://maricare.com/en/how-it-works/elsi-smart-floor (visited on 06/09/2022).

[26] V. Cotechini, A. Belli, L. Palma, M. Morettini, L. Burattini, and P. Pierleoni, “A dataset for the development and optimization of fall detection algorithms based on wearable sensors”, *Data in Brief*, vol. 23, p. 103 839, 2019. DOI: 10.1016/j.dib.2019.103839. [Online]. Available: https://doi.org/10.1016/j.dib.2019.103839.

[27] S. B. Khojasteh, J. R. Villar, E. de la Cal, V. M. González, and J. Sedano, “Fall Detection Analysis Using a Real Fall Dataset”, *Advances in Intelligent Systems and Computing*, vol. 771, pp. 334–343, 2019. DOI: 10.1007/978-3-319-94120-2_32.

[28] X. Yu, J. Jang, and S. Xiong, “A Large-Scale Open Motion Dataset (KFall) and Benchmark Algorithms for Detecting Pre-impact Fall of the Elderly Using Wearable Inertial Sensors”, *Frontiers in Aging Neuroscience*, vol. 13, no. July, pp. 1–14, 2021. DOI: 10.3389/fnagi.2021.692865.

[29] Helite, *Hip’Guard*. [Online]. Available: https://en.helite.com/hipguard/ (visited on 06/08/2022).

[30] O. Aziz, M. Musngi, E. J. Park, G. Mori, and S. N. Robinovitch, “A comparison of accuracy of fall detection algorithms (threshold-based vs. machine learning) using waist-mounted tri-axial accelerometer signals from a comprehensive set of falls and non-fall trials”, *Medical and Biological Engineering and Computing*, vol. 55, no. 1, pp. 45–55, 2017. DOI: 10.1007/s11517-016-1504-y.

[31] International Business Machines Corporation (IBM), *Deep Learning*, 2020. [Online]. Available: https://www.ibm.com/cloud/learn/deep-learning (visited on 02/24/2022).

[32] International Business Machines Corporation (IBM), *Neural Networks*, 2020. [Online]. Available: https://www.ibm.com/cloud/learn/neural-networks (visited on 02/24/2022).

[33] Big Cloud, *The Difference Between White Box and Black Box AI*. [Online]. Available: https://bigcloud.global/the-difference-between-white-box-and-black-box-ai/ (visited on 05/11/2022).

[34] M. Deutsch and H. Burgsteiner, “A smartwatch-based assistance system for the elderly performing fall detection, unusual inactivity recognition and medication reminding”, *Studies in Health Technology and Informatics*, vol. 223, pp. 259–266, 2016.

[35] MEMSIC Semiconductor Co. Ltd, “MC3419 3-Axis Accelerometer GENERAL DESCRIPTION FEATURES”, no. 158, pp. 1–79, 2020.

[36] Silicon Sensing Systems Limited, *MEMS Accelerometers*. [Online]. Available: https://www.siliconsensing.com/technology/mems-accelerometers/ (visited on 02/01/2022).

[37] Decawave Ltd., “DW1000 USER MANUAL (Version 2.11)”, 2017. [Online]. Available: https://www.decawave.com/sites/default/files/resources/dw1000_user_manual_2.11.pdf (visited on 02/13/2022).

[38] STMicroelectronics, *STM32L071x8 STM32L071xB STM32L071xZ Datasheet*, 2019. [Online]. Available: https://www.st.com/resource/en/datasheet/stm32l071cb.pdf (visited on 02/11/2022).

[39] B. W. Kernighan and D. M. Ritchie, *The C programming language*. 2006.

[40] TeraTerm Project, *Tera Term Home Page*. [Online]. Available: http://ttssh2.osdn.jp/ (visited on 04/28/2022).

[41] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace, 2009, ISBN: 1441412697.

[42] J. D. Hunter, “Matplotlib: A 2d graphics environment”, *Computing in science & engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[43] C. R. Harris, K. J. Millman, S. J. van der Walt, *et al.*, “Array programming with NumPy”, *Nature*, vol. 585, pp. 357–362, 2020. DOI: 10.1038/s41586-020-2649-2.

[44] W. McKinney, “Data structures for statistical computing in python”, in *Proceedings of the 9th Python in Science Conference*, Austin, TX, vol. 445, 2010, pp. 51–56.

[45] C. Liechti, *pySerial*, 2001. [Online]. Available: https://pyserial.readthedocs.io/en/latest/pyserial.html (visited on 04/28/2022).

[46] T. Kluyver, B. Ragan-Kelley, F. Pérez, *et al.*, “Jupyter notebooks – a publishing format for reproducible computational workflows”, in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, F. Loizides and B. Schmidt, Eds., IOS Press, 2016, pp. 87–90.

[47] Q. T. Huynh and B. Q. Tran, “Time-Frequency Analysis of Daily Activities for Fall Detection”, *Signals*, vol. 2, no. 1, pp. 1–12, 2021. DOI: 10.3390/signals2010001.