Rohit George Andrews

# Mobile Sensor Data Measurements and Analysis for Fall Detection in Elderly Health Care

**School of Electrical Engineering**

Thesis submitted for examination for the degree of Master of Science in Technology.

Espoo May 22, 2017

**Thesis supervisor:**

Prof. Jorma Skyttä

**Thesis advisor:**

Prof. Katsuyuki Haneda

Aalto University
School of Electrical
Engineering

AALTO UNIVERSITY
SCHOOL OF ELECTRICAL ENGINEERING

ABSTRACT OF THE
MASTER'S THESIS

Author: Rohit George Andrews

Title: Mobile Sensor Data Measurements and Analysis for Fall Detection in Elderly Health Care

Date: May 22, 2017 Language: English Number of pages: 8+54

Department of Signal Processing and Acoustics

Professorship: Signal Processing Code: ELEC3031

Supervisor: Prof. Jorma Skyttä

Advisor: Prof. Katsuyuki Haneda

In recent years, increased life expectancy in Finland and other parts of the world have led to an aging population. Accidental falls can cause severe injuries to elderly people, thereby, negatively impacting their quality of life and in some cases resulting in death. Accidental falls is a major public health care challenge. Real time monitoring of human activity can provide insight into an individual's functional ability and gives an indication of their ability to live independently. Automatic detection of falls enables us to provide timely medical attention, thereby, reducing the negative consequences of falls. This paradigm of home based health promotes independent living and reduces the burden on caregivers.

The aim of the thesis is to log real world sensory data from multiple sensors on board mobile devices and develop suitable algorithms to extract information from the data to solve the problem of detecting when elderly people fall down. In order to log the data, an Android application is developed that collects data from the various onboard sensors and stores it in a text file. The developed application is used to take measurements of sensor data pertaining to various human activities. Then patterns in the data are then analysed and exploited to distinguish between normal day-to-day activities and people falling down. To detect falls, we develop two algorithms based on statistical detection theory and convex optimization, respectively and also analyze the efficacy of these methods.

Keywords: Mobile sensors, Fall detection, Android application development, Statistical detection theory, Elderly health care, Convex optimization, Machine learning

# Acknowledgements

I would like to thank Prof. Jorma Skyttä for accepting to supervise this thesis despite his busy schedule. This thesis work would not be completed with your support and encouragement.

I would also like to thank Prof. Katsuyuki Haneda who is the instructor for this thesis work. Prof. Haneda has been advising me all along in my effort to get this thesis work completed. I am greatly indebted to him for his patience, guidance and support right throughout this thesis work. My sincere thanks goes to him.

I would like to thank my family and my close friends who encouraged and motivated me in completing this thesis work. I would also like to thank Dr. Pramod Jacob Mathecken for general discussions related to this thesis topic. I am truly grateful for your support. My gratitude goes immensely to Aalto University and, in particular to, Jenni Tulensalo and Eeva Halonen from the Student Services for their support and guidance.

Otaniemi, May 22, 2017

Rohit George Andrews

# Contents

# List of Symbols

| | |
|---|---|
| $E_s$ | Signal energy |
| $m/s^2$ | Meter per second square |
| $1g$ | 9.81 $m/s^2$ |
| $N_0$ | Noise power |
| $P_f$ | Probability of false alarm |
| $P_d$ | Probability of detection |
| $r(n)$ | Received time-domain signal |
| $s(n)$ | Desired time-domain signal |
| $T$ | Test statistic |
| $w(n)$ | Time-domain noise signal |
| $\chi^2_{2M}$ | Central chi-square distribution with $2M$ degrees of freedom |
| $\chi^2_{2M}(2\gamma)$ | Non-central chi-square distribution with $2M$ degrees of freedom and non-centrality parameter $2\gamma$ |
| $\eta$ | Decision threshold |
| $\gamma$ | $\frac{E_s}{N_0}$ |
| $F^{-1}_{\chi^2_{2M}}$ | Inverse cumulative distribution function of a $\chi^2$ distribution |

# List of Figures

# 1 Introduction

## 1.1 Background

Humankind has experienced unprecedented technological progress in the last century and, more so, in the last fifty years. Human civilization has developed automobiles and air transport, discovered antibiotics, broken the genetic code, split the atom to harness its energy and invented computing. Communication technology was revolutionized with the telephone, radio, television, cellular networks and the Internet. Perhaps much of this would not have been possible without the rapid growth of electronics and computing.

However, the recent information and communication revolution has brought about a paradigm shift in the way we lead our lives and do business. This has been due to the exponential growth of the internet and cellular networks. Communication has become such an integral part of modern human civilization that access to the internet is today considered a basic right. In recent years there has been a tremendous growth in the market for wireless communications. At the same time mobile and ubiquitous computing have become all pervasive because of the easy availability of mobile devices coupled with cheap wireless connectivity. As wireless connectivity has become more accessible, along with the increased bandwidth, it has become possible to connect these mobile devices to each other and to servers. Today users own multiple mobile devices such as laptop, smartphone and tablet. Each of these mobile devices has multiple sensors on-board. It is possible to acquire sensory data from multiple sources, perform signal processing and extract the required information from the data. Considering all these factors, today it is feasible to create applications on mobile phones that were once in the realm of science fiction.

Sensor data fusion is a multi-disciplinary subject that combines technique from areas such as statistical signal processing, computer science, machine learning and data mining among others. It can be described as the process of using information from several different sensors to compute an estimate of the state of a dynamic system. Essentially the question is how to combine, in the best possible manner, diverse and uncertain measurements in a multiple sensor system in order to estimate the state of a system. We use multiple sensors to improve the accuracy and reliability of the estimation in order to reduce uncertainty and obtain more complete knowledge of the state. The resulting estimate is generally better than it would be if the sensors were used alone. Currently there are many smart phones in the market. They have

multiple in-built sensors such as accelerometer, gyroscope, barometric sensor, GPS receiver, camera and multiple radios. There is a huge untapped potential to create applications using the data collected from different sensors [1, 2].

## 1.2 Motivation

One such application that can immensely reap the benefits of multi-sensor data and processing is tracking and fall detection of elderly people [3]. Demographic trends indicate that the population of western Europe, Japan and other parts of the industrialised world is ageing rapidly. Finland has one of the highest rates of ageing which is primarily due to developments in medical science and technology. However, at the same time, there is an increasing number of people suffering from type-2 diabetes, cardiovascular diseases, cancer, hypertension, osteoarthritis, depression, Alzheimer's disease and other diseases that result in decline of cognitive abilities. In most countries people are encouraged to stay at home and take care of themselves as long as possible in order to reduce costs. It is only when they suffer from life threatening conditions or are incapable of caring for themselves that they should move to hospitals or old age homes. This approach to taking care of the elderly is practical only if there is a mechanism to remotely monitor their health..

The problem of an ageing society not only results in a smaller workforce but also causes a reduction in the number of potential caregivers. A solution that is capable of monitoring elderly people and notifying caregivers will definitely improve the situation. According to caregivers two major causes of concern are [4]

1. Falls - the problem of falls among elderly people has a big social and economic impact. It is the sixth most likely cause of death for people aged over sixty five years, second for people between sixty five years and seventy five years and first for people over seventy five years. For people suffering from Alzheimer's disease, the probability of fall increases by a factor of three.

2. Wandering around - People who suffer from Alzheimer's have the tendency of wandering around increasing the likelihood of them meeting with accidents

Remote monitoring can potentially be very useful in providing care for senior citizens living independently. Most people in the sixty to seventy year age group are active and travel outdoors independently whereas, people in the seventy to ninety year age group, are confined indoors in old age homes. However, they are likely to

carry a mobile phone. By processing data acquired by various sensors on-board a mobile phone, i.e., camera, GPS receiver, accelerometer, gyroscope, WLAN receiver etc., one track and detect falls [5, 6]. For tracking, outdoor GPS receivers function quite well but the estimates can be improved by using additional information. Historically, inertial navigation was performed using accelerometer and gyroscope. Combining this information with the received signal strength of signals from the various radios and triangulation methods, it is possible to locate a person indoors by dead reckoning. Augmenting the GPS data with inertial navigation data, WLAN, 2G , 3G and 4G radio positioning data can potentially enhance the performance of the indoor localisation system [7]. A preliminary review of the literature on this topic suggests that this approach has not been considered before. Furthermore, it is possible to detect when the person has fallen down by combining data from accelerometer, gyroscope and barometric sensor [8, 9, 10].

The approach that this thesis proposes is to use mobile phones for monitoring health of the elderly [9, 11]. This involves fall detection and tracking of elderly people. Mobile phones present a mature hardware and software platform which inherently has many of the components required to detect and communicate a fall or track a person [12]. Moreover, they are portable and function practically everywhere. In addition it would be comparatively easy to convince an elderly person to carry a mobile phone. Currently , there are many smartphones in the market with multiple sensors including accelerometers, gyroscopes, magnetometer, barometric sensor and multiple radios [13, 14, 15, 16].

To summarize, falls are a major health hazard for elderly people and they have a huge social and economic impact. The consequences of a fall can vary from scratches to fractures and in some cases death. Even if there are no immediate consequences, a long-wait lying on the floor without medical help increases the probability of death from the accident. This underscores the importance of real-time monitoring and detection of a fall to enable first-aid by relatives, paramedics or caregivers as soon as possible. The care of elderly people can be improved by using sensors to monitor the vital signs and activities of patients and communicate this information to their doctors and caregivers [17].

## 1.3 Research Problem and Scope

With unintentional falls being among the leading causes of severe injuries to elderly people, often resulting in death, the goal of this thesis is to demonstrate the

feasibility of designing a smartphone based system to detect fall events. This would require real world sensory data collection from multiple sensors, mathematically model the data and, finally, to develop suitable algorithms that extract meaningful information in order to solve the problem of fall detection.

## 1.4 Contribution of the Thesis

In this thesis, we develop a data logger application for Android phones that collects and stores data from various sensors onboard an Android smart phone. This is followed by a series of measurements of the sensors for a variety of human activities including walking, climbing and falling. The obtained data is then fed as input to train two algorithms to detect falls. In particular, the algorithms are based on statistical detection theory and convex optimization theory, respectively. We demonstrate that these algorithms are efficient at detecting falls.

## 1.5 Outline of the Thesis

The rest of this thesis is organised as follows. Chapter 2 provides a brief background of various sensors onboard a mobile phone. Chapter 3 is about Android application development. In particular, an application that logs sensory data is presented. Chapter 4 is about sensor measurements, where we present results of sensor data for common human activities. These measurements are used as training data for training the algorithms used to detect falls. The theory behind the usage of these algorithms is presented in Chapter 5. We end this thesis with a quick summary of the thesis along with some thoughts on future work.

# 2 Sensors

In this chapter, we provide brief overview of the sensors that are present on the two devices used in this thesis, namely, the Google Nexus 4 and Motorola Xoom. Modern smartphones are ubiquitous and have multiple onboard sensors like gyroscope, accelerometer, magnetometer and barometric sensor. Most of the sensors used in modern mobile devices are MEMS (micro electromechanical systems) sensors. Data collected from multiple onboard sensors can be used to detect falls. This approach is more cost effective and easier to implement because the only requirement for a functioning system is downloading an application onto a mobile phone. This section describes what sensors are available and what they measure. These measurements can be used for fall detection and perhaps for indoor localisation too.

## 2.1 Available Sensors

The Sensors available on board the phone are as follows:

- 3-axis accelerometer
- 3-axis gyroscope
- 3-axis magnetometer
- barometric sensor
- GPS receiver

## 2.2 Accelerometer

Accelerometers are sensing transducers that provide an output proportional to acceleration, vibration and shock, i.e., they measure acceleration [18, 19, 20, 21]. In simple terms when an object moves, it experiences acceleration. Measurement of this acceleration helps us in understanding the dynamic characteristics that govern the behaviour of that object. Acceleration is generally measured/quantified in the SI unit metres per second per second ($m/s^2$) [22]. In other words, what that means is that an accelerometer at rest will measure an acceleration of 1g (9.81 $m/s^2$) upwards whereas an accelerometer in free fall will measure an acceleration of zero.

Traditional accelerometers can be broadly classified int two categories namely mechanical and solid sate accelerometers. A mechanical accelerometer is made up of a mass suspended by springs. The displacement of the mass, which is proportional to the force acting on the mass is measured and Newton's law is then used to calculate the acceleration.

There are many types of solid state accelerometers like for instance surface acoustic wave, vibratory, quartz and silicon devices. Some of these devices use the piezoelectric effect to measure acceleration. In the case of surface acoustic wave (SAW) accelerometer a cantilever beam is resonated at a particular frequency. One end of the beam is fixed rigidly while the other end has a mass attached to it that is free to move. When a force is applied to the beam accelerating it the beam bends. This causes the frequency of the surface acoustic wave to change in proportion to the applied force. We can can measure the acceleration from the change is frequency.

Most modern electronic devices use micro electromechanical systems (MEMS) or micro-machined silicon accelerometers. MEMS acclerometers can also be of two types: The first one consists of mechanical accelerometers made using MEMS fabrication techniques and the second one consists of devices which measure change in frequency like the SAW accelerometer. The advantages of MEMS accelerometers are they have low power consumption, quick startup times and are very small and light compared to traditional systems. However they are not as accurate although performance is continuously improving as time goes by. The accelerometers onboard modern day smartphones are 3-axis accelerometers.

## 2.3 Gyroscope

A gyroscope is a device used to measure of angular velocity. Angular velocity can be measured in the SI unit of radians per second [23]. Gyroscopes are used for maintaining orientation, based on the principles of angular momentum . In modern smartphones 3-axis gyroscopes are often implemented with a 3-axis accelerometer to provide a full six degree-of-freedom (DoF) motion tracking system [24]. The classic mechanical gyroscope exploits the law of conservation of angular momentum according to which, the total angular momentum of a system is constant in both magnitude and direction if the resultant external torque acting upon the system is zero. These gyroscopes typically consist of a spinning disk or mass on an axle, which is mounted on a series of gimbals.

Gyroscopes based on many other operating principles have subsequently been developed. Some of them are the electronic microchip-packaged MEMS gyroscope devices found in consumer electronic devices, solid-state ring lasers and fibre optic gyroscopes. The most used MEMS gyroscopes are vibrating structure gyroscopes [6]. The operating principle of vibrating structure gyroscope is based on the Coriolis force. Vibrating structure gyroscopes are MEMS devices that are easily available commercially, affordable, and very small in size. Optical gyroscopes were developed using laser technology and their functionality depends on the constancy of the speed of light. The operating principle of optical gyroscopes is based Sagnac effect. The advantage of this type of gyroscope is that they contain no moving parts, and hence are not susceptible to mechanical wear or drifting.

## 2.4 Magnetometer

The magnetometer sensor onboard modern smartphones is a miniature Hall-effect sensor that detects the Earth's magnetic field along three perpendicular axes namely X, Y and Z [25]. The magnetometer measures magnetic field in micro Tesla units. The magnetometer is used for detecting the relative orientation of the device relative to the Earth's magnetic north.The operating principle is based on Hall-effect [26]. The Hall-effect produces a voltage which is proportional to the strength and polarity of the magnetic field along the axis of each sensor. Other operating principles used in magnetometers may include magneto resistive devices which change the measured resistance based on changes in the magnetic field.

## 2.5 Barometric Sensor

Barometer is a device used to determine atmospheric pressure. The measurements are recorded in mbar [27]. The barometric sensor can be used as an altimeter by computing the altitude at a particular place from the variation in pressure along the the vertical based on the atmospheric pressure at a reference like sea level. There are numerous methods used to measure pressure using transducers. However,most MEMS barometric sensors used smartphones work based on piezo-electric principle. The other working principle employed in some digital MEMS barometric sensors is the capacitive sensing principle [28].

## 2.6 GPS

The global positioning system (GPS) is a satellite-based positioning system suitable for outdoor applications that was developed by the United States government for military purposes but is currently used worldwide for civilian applications [29]. In its basic configuration, it consists of twenty four satellites in six circular orbital planes. Currently, however, there are thirty two satellites in orbit for improved accuracy. These satellites are continuously transmitting radio signals, which be decoded by a user to determine position and velocity.

There are four signals available for civilian use these are L1 C/A, L2C, L5 and L1C. The military signals are encrypted. GPS signals include ranging signals and navigation messages. The ranging signals are used to measure the distance to the satellite. The navigation messages include ephemeris data which is used to calculate the position of each satellite in orbit as well as information about the time and status of the entire satellite constellation called the almanac. The GPS system can be augmented using base stations broadcasting local correction terms. There are a variety of augmented sytems such as wide area augmentation system (WAAS), European geostationary navigation overlay service (EGNOS), differential GPS (DGPS), inertial navigation systems (INS) and Assisted GPS. The accuracy of these systems varies from fifteen meters to three meters . There are other navigation systems such as Russia's GLONASS, the European Union's Galileo positioning system, China's BeiDou Navigation Satellite System and India's NAVIC.

# 3 Application Development

The earlier attempts in fall detection were mostly based on techniques in which sensors were attached to the body of the person of interest to track motion. This approach was in our opinion neither practical nor economical. Modern mobile devices are ubiquitous and have multiple onboard sensors like gyroscope, accelerometer, magnetometer, barometric sensor etc. Data collected from multiple onboard sensors can be used to detect falls. This approach would be more cost effective and easier to implement because all that was required was downloading an application onto a mobile phone.

We were provided with two devices namely Google Nexus 4 and Motorola Xoom, which were running Android 4.2 Jelly Bean and Android 3.2 (Honeycomb) respectively [30, 31, 32]. Android Phones and tablets are mobile devices that have Android OS running on them and can have Android applications installed on them. These applications are written in Java programming language and they are called mobile device applications or apps [33]. Development of apps involves writing sets of Java code focused on implementing particular task that performs some function for a mobile device application. Three important tools that are used in an Android development environment are the java programming language, Eclipse which is an integrated development environment (IDE) and the Android operating system. In order to build an android app the following components must be installed: the Android java development kit, Android software development kit (SDK), Eclipse, Android Developer Tools plug-in and, finally, setting up testing environment.

We developed an android application to read and store data from various sensors. Using the application we performed various measurements for different activities like walking, climbing, falling etc. We analysed the collected data and developed algorithms to detect falls using simple heuristics.

## 3.1 Android Platform

Android is an open source platform for mobile devices that is based on the linux kernel for mobile devices and is currently developed and managed by Google. Android is designed for touchscreen mobile phones, tablets, etc., and is currently the most popular mobile OS. The user interface of android requires touch inputs like swiping, tapping, etc., to manipulate on screen objects. In addition inbuilt sensors such as a accelerometers, gyroscopes and proximity sensors respond to user actions

like tilting the phone by changing screen from portrait to landscape depending on the orientation of the device [34]. The Android OS allows users to customize their screens with short cuts and widgets that improve the ease of operation and help retrieve relevant information easily. The android play store has over a million apps and there are over a billion android devices in the world. The android developer community is one of the most mature communities.

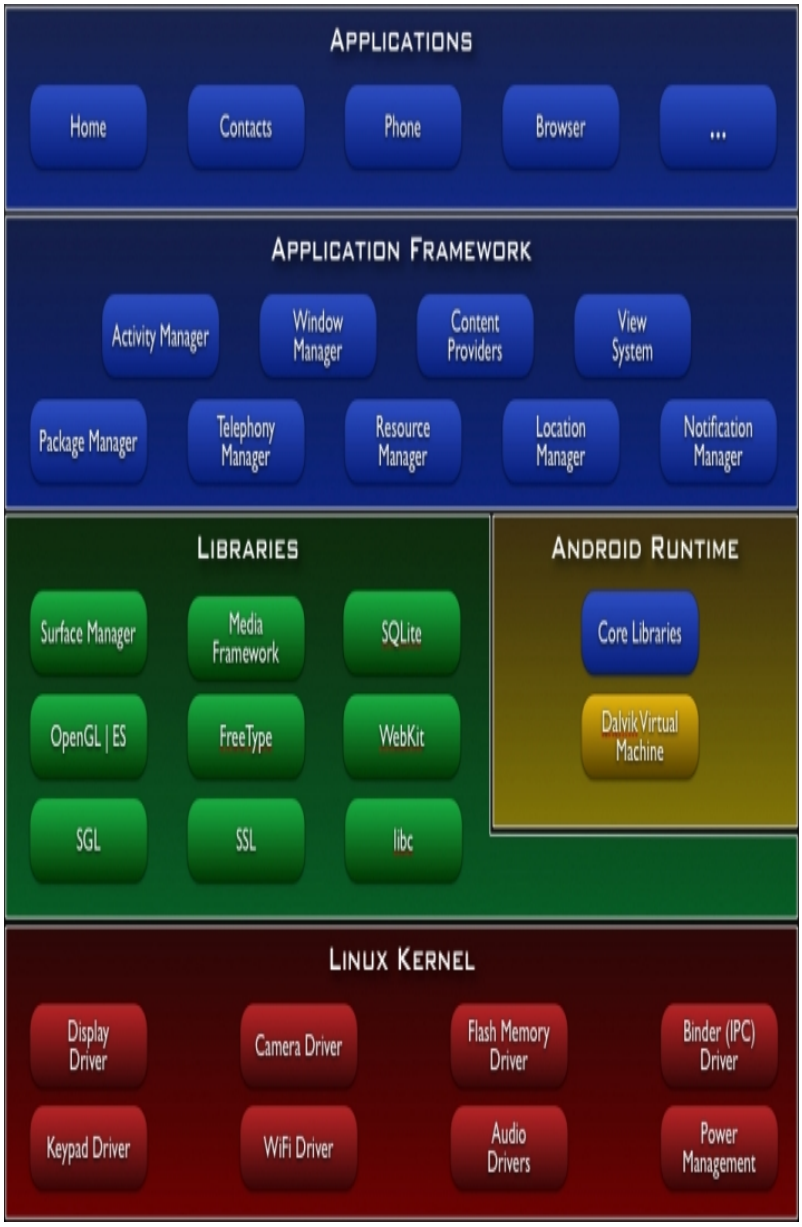



**Figure 1:** *Android system architecture.*

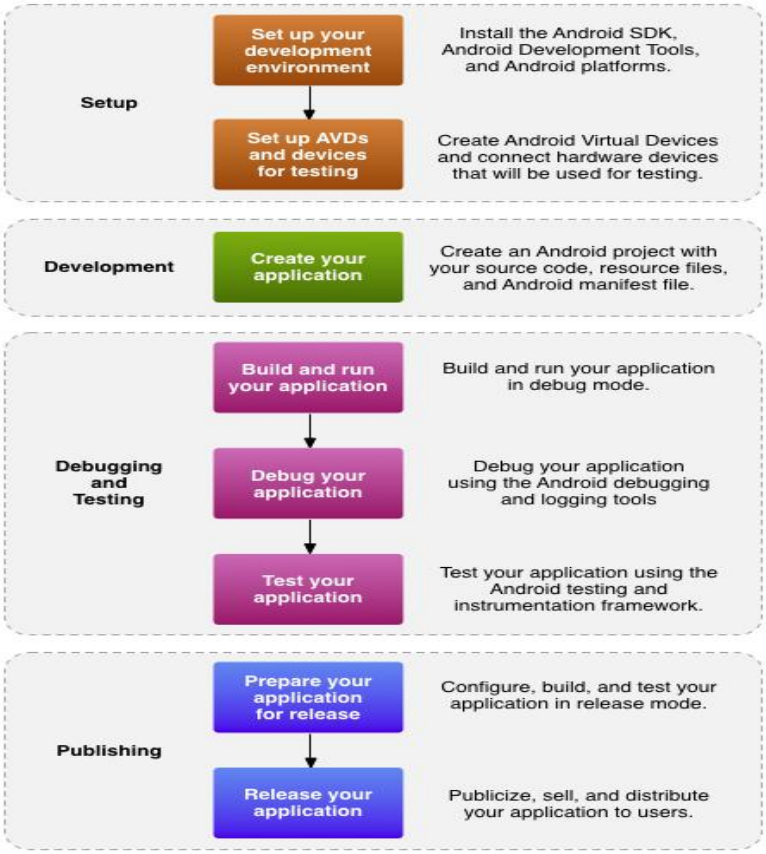



**Figure 2:** *Android development overview.*

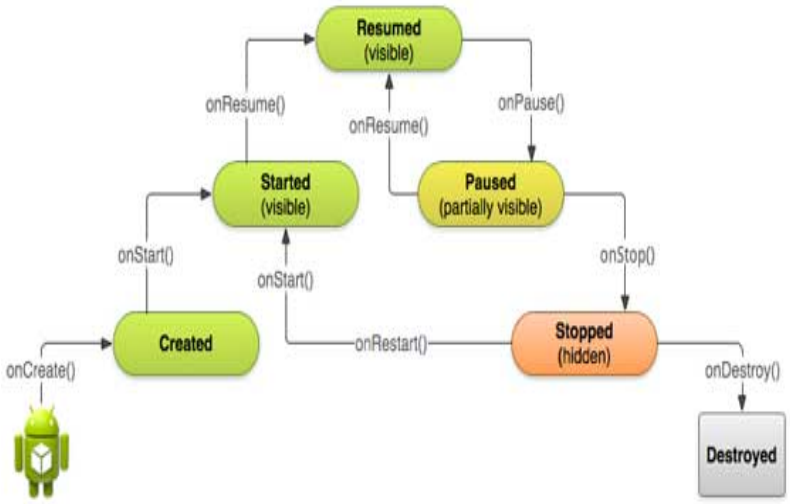



**Figure 3:** *Android activity lifecycle.*

## 3.2 Android Software Development

Android software development refers to the process used to create new applications for the Android operating system. Applications are generally developed in the Java programming language using the Android SDK. The Android SDK has a comprehensive set of development tools including a debugger, libraries, a handset emulator, documentation, sample code and various other tools.

Android software development involves the following steps:

- Setup
- Development
- Debugging and Testing
- Publishing

### 3.2.1 Setup

In this phase of the development cycle we installed and set up Eclipse IDE along with Android SDK and ADT to develop an application that collects and saves data from the various onboard sensors. We also created Android Virtual Devices (AVDs) in order to install and test our application. An AVD is an emulator that allows us to model an actual device by defining hardware and software options to be emulated by the Android emulator.

An AVD consists of a hardware profile that defines the hardware features of the virtual device. It is possible for instance to define whether the device has a camera, whether it uses a physical QWERTY keyboard or a dialling pad, how much memory it has, etc. It is also possible to define what version of the Android platform will run on the virtual device. One can choose a version of the standard Android platform and can also specify the emulator skin to be used with the AVD, which allows to control the screen dimensions, appearance, etc. The emulated SD card to be used with the AVD can also be specified. In addition the device's user data (installed applications, settings, and so on) and emulated SD card can be stored in this area. We can can create as many AVDs as we need, based on the types of device we want to model. To thoroughly test our application, we should create an AVD for each device configuration with which our application is compatible and test our application on each one.

While selecting a system image target for our AVD we should keep in mind that the API Level of the target is important, because our application will not be able to run on a system image whose API Level is less than that required by our application. We should create at least one AVD that uses a target whose API Level is greater than that required by our application, because it allows us to test the forward-compatibility of our application. When building a mobile application, it's important that we always test our application on a real device before releasing it to users. We can use any Android-powered device as an environment for running, debugging, and testing your applications. The tools included in the SDK make it possible to install and run our application on the device each time you compile. We can install your application on the device directly from Eclipse or from the command line with ADB.

### 3.2.2 Development

In this phase we set up and develop our Android project, which contains all of the source code and resource files for our application. Projects act as containers for storing things such as code and resource files. The SDK tools expect your projects to follow a specific structure so it can compile and package your application correctly, so it is highly recommended that you create them with Eclipse and ADT or with the android tool on the command line. There are three types of projects, and they all share the same general structure but differ in function:

- Android Project: An Android project is the container for our application's source code, resource files, and files such as the Ant build and Android Manifest file. An application project is the main type of project and the contents are eventually built into an .apk file that you install on a device.
- Test Projects: These projects contain code to test our application projects and are built into applications that run on a device.
- Library Projects: These projects contain shareable Android source code and resources that we can reference in Android projects. This is useful when we have common code that we want to reuse. Library projects cannot be installed onto a device, however, they are pulled into the .apk file at build time. When we use the Android development tools to create a new project, the essential files and folders will be created for us by default. There are only a handful of files and folders generated, and some of them depend on whether we use the

Eclipse plugin or the android tool to generate our project. As our application grows in complexity, we might require new kinds of resources, directories and files.

### 3.2.3 Debugging and Testing

In this phase we build our project into a debuggable .apk package that we can install and run on the emulator or an Android-powered device. When we use Eclipse, builds are generated each time our project is saved. when we use other IDE's , we build our project using Ant and install it on a device using adb. We debug our application using a JDWP-compliant debugger along with the debugging and logging tools that are provided with the Android SDK. Eclipse already comes packaged with a compatible debugger. We can test our application using various Android SDK testing tools.

The Android framework includes an integrated testing framework that helps us to test our application and the SDK tools contain tools for setting up and running test applications. We can choose to use either Eclipse with ADT or command line, in order to set up and run our tests within an emulator or the device we are targeting.

### 3.2.4 Publishing

In this phase we configure and build our application for release and distribute our application to users. Publishing is the general process that makes our Android applications available to users. When we publish an Android application we perform two main tasks:

We prepare the application for release. During the preparation step we build a release version of your application, which users can download and install on their Android-powered devices. We release the application to users. During the release step we publicize, sell, and distribute the release version of our application to users. Usually, we release our application through an application marketplace, such as Google Play. However, we can also release applications by sending them directly to users or by letting users download them from our own website.

The figure 2 shows how the publishing process fits into the overall Android application development process. The publishing process is typically performed after you finish testing your application in a debug environment. Also, as a best practice, our application should meet all of our release criteria for functionality,

performance, and stability before we begin the publishing process

## 3.3 Activity

An activity can be described as an entry point for an application to interact with a user through the UI. Switching between screens means moving from one activity to another. Activities are classes within packages that interact with the user. They extend an Activity type and hence inherit methods and other related information. The figure 3 presents the Android Activity life-cycle, in other words the states that an activity can be in, depending on the user's interaction with it and also describes the state paths an activity can take. According to the Android Developers guide, the programmer can implement callback methods to make the application perform the desired operations when the Activity moves between states. If an activity in the foreground of the screen it can be said to be active or running. If an activity has lost focus but is still visible, it is paused. A paused activity is completely alive. It maintains all state and member information. But, it can be killed by the operating system when memory is extremely low. If an activity is in the background and completely obscured by another activity, it is stopped. It still retains all state information. But, it is no longer visible to the user. When memory is needed it will be killed by the operating system. If an activity is paused or stopped, the operating system can drop the activity from memory by either asking it to finish, or simply killing it's process. When the activity comes back to the foreground and is displayed again to the user, it must be restarted and restored to its previous state.

## 3.4 Application

The objective of the Android application development process was to create an application that collects data from the various on board sensors and stores it in a text file in .txt format. The design of the software was relatively straightforward since the android software development kit provides inbuilt classes and methods to access the onboard sensors. The use case can be intuitively guessed from the screenshot of the application's user interface and is as follows. The user of the application presses the start button in order to issue the start commmand. Next the application starts recording the data from the various onboard sensors in a text file. In addition the data is displayed on the screen for the user to read. This goes on until the stop button is pressed.

When it comes to Android Applications there are four primary components : Activity, Service, BroadcastReceiver, and ContentProvider . A single Android application might not contain all of these elements, but will have at least one of these elements. In our App there is an Activity and a BroadcastReceiver. The Activity listens to the sensors and displays the results on the UI while the Broadcast receiver gets notified whenever an event happens to the Wifi network. In addition to being displayed on the UI the data from both these modules is also written into a text file. In general terms an activity displays a UI (user interface) and responds to system as well as user initiated events. Activities are declared in AndroidManifest.xml file, which could be considered the bedrock on which an Android application is built. Activities present their views through XML layouts and communicate with each other through Intents. A broadcast receiver is an Android component which allows you to register for system or application events. All registered receivers for an event are notified by the Android runtime once this event happens.

The figure 6 shows the sequence diagram of the application. When the start button is pressed the OnClickListener is triggered. The OnClickListener routine activates SensorManager which gets a list of sensors and registers listeners for all of them. Objects of View class like button contain a collection of nested interfaces with callbacks that can be defined by the programmer. These interfaces, called event listeners, are used to capture the user's interaction with the UI. When there is any movement or change in reading a sensor event is triggered. As a consequence of the event, the sensor values are output to the screen and at the same time written to a text file. When the stop button is pressed the Listeners for the various sensors are unregistered.

The UI was made using Automatic XML layout setup. One way of creating XML layouts is by dragging and dropping objects inside a work area. Eclipse will generate the XML code and create a few attributes for them. Objects on different type in the Palette on the left side in the Android JDK. The FinalappActivity and BroadcastReceiver were declared in their respective class files and registered in AndroidManifest.xml file. The code was compiled and an App was created. Later the App was installed on both the devices namely Google Nexus 4 and Motorola Xoom.

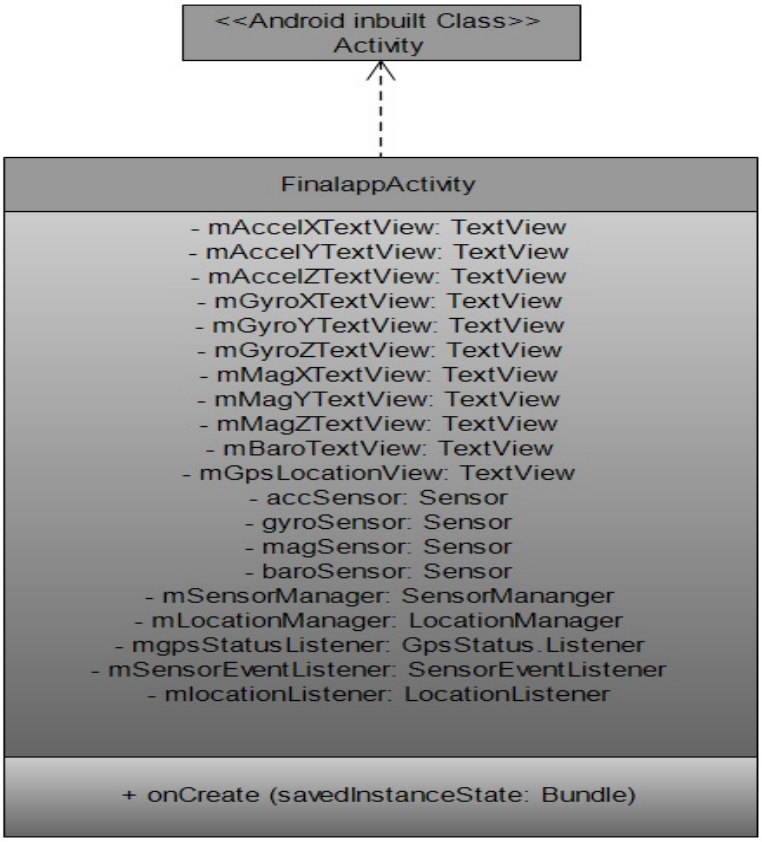



**Figure 4:** *Class diagram of Finalappactivity.*

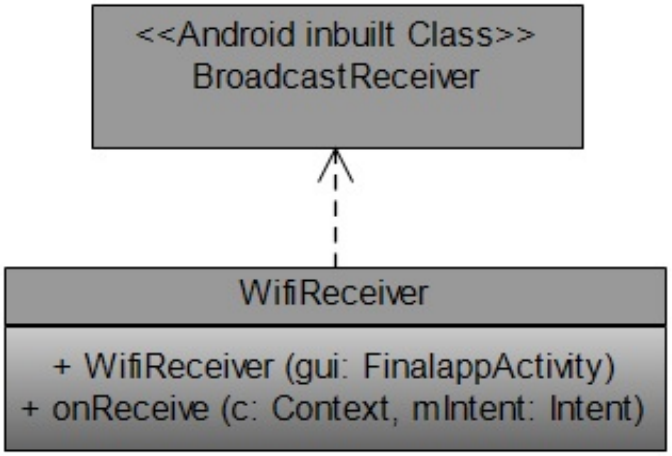



**Figure 5:** *Class diagram of BroadcastReceiver.*

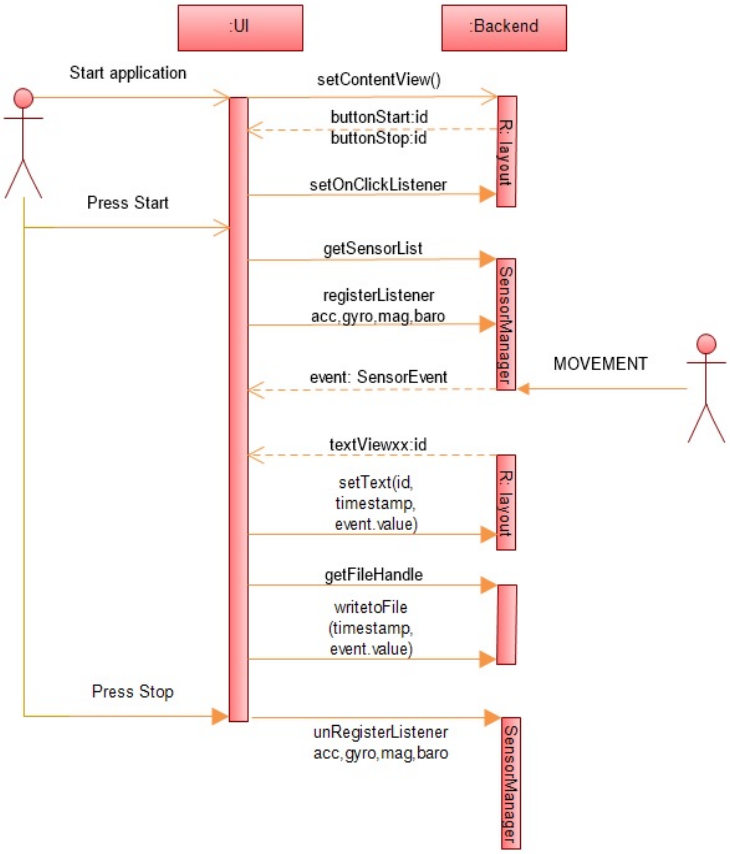



**Figure 6:** *Sequence diagram.*

# 4 Sensor Measurements

In this chapter we describe the measurements using various sensors in multiple scenarios that were performed to collect data for analysis in order to develop fall detection algorithms. The goal of performing the measurements was to collect data and the exploit patterns in the data to distinguish between normal day to day activities and people falling down. Since each fall is unique we had multiple falls in different directions namely forward, backward and sideways. To keep the data as realistic as possible the readings were taken by multiple users using both the devices. To perform calibration and to serve as a reference the devices were placed on a desk in the office for the stationary case. For the walking and climbing stairs case the measurements were performed in the electrical engineering department. Most of the measurements were taken during the course of a month in the summer of 2012.

To increase the diversity four different people were engaged to perform the falls. A thick mattress was placed on the floor and readings were taken as the test subjects repeatedly simulated falls in the forward, backward and sideways directions respectively.

The Measurements were performed using the following two devices.

- Google Nexus 4
- Motorola Xoom

The test set-up involved the volunteer user holding the device at waist or hip level when the application was started and after a certain duration the application was stopped. Some reading were taken with the device in with user's pant pocket with the application running. Once the data was obtained, it was plotted for each sensor and studied individually. This was done in order to visually inspect the data and find any defining characteristics or patterns in the data. We observed that each of the different activities have unique profiles for various sensor data. Also, amplitudes and frequencies of movement depend on the size and weight of the volunteer.

The tests involved the following activities:

1 The device in a stationary position.

2 The user walking.

3 The user climbing up the stairs.

4 The device being thrown vertically upwards.

5 The user falling forward.

6 The user falling backwards.

7 The user falling sideways.

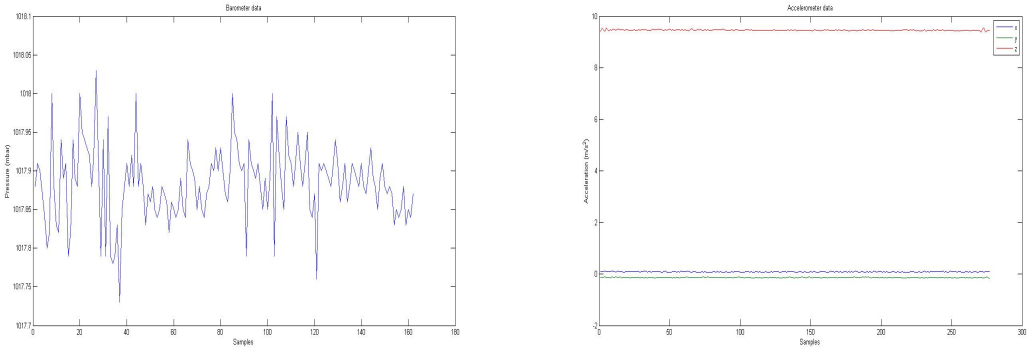
One of the major problems, while performing the measurements, is the inability to set a desired sampling frequency as well as the inherent noisiness of the measurements. It would be ideal to choose a high sampling frequency. However, in an android device, it is not possible to set a user specific sampling frequency and read directly from the inertial sensors. Instead a sensor event is generated each time the sensor value changes. The disadvantage of this setup is that crucial information can be lost. To solve the indoor localization problem, ideally we should have control over the sampling frequency. The data collected from the sensors, in general, can be expected to be noisy. In addition to the noise, a large bias is also noticeable as show in Table 2.

| Parameter | Average bias along z | Average bias along y | Average bias along x |
|---|---|---|---|
| accx | 0.104778141 | 0.131273742 | 0.106861459 |
| accy | -0.010368234 | -0.061804072 | 0.049341285 |
| accy | -0.360383032 | 0.382245928 | 0.250262686 |
| omegax | -0.009591585 | -0.007659805 | -0.009172497 |
| omegay | 0.016612772 | 0.016956013 | 0.01738512 |
| omegaz | -0.01511746 | -0.015525771 | -0.015539849 |

**Table 2:** *Bias in accelerometer and gyroscope.*

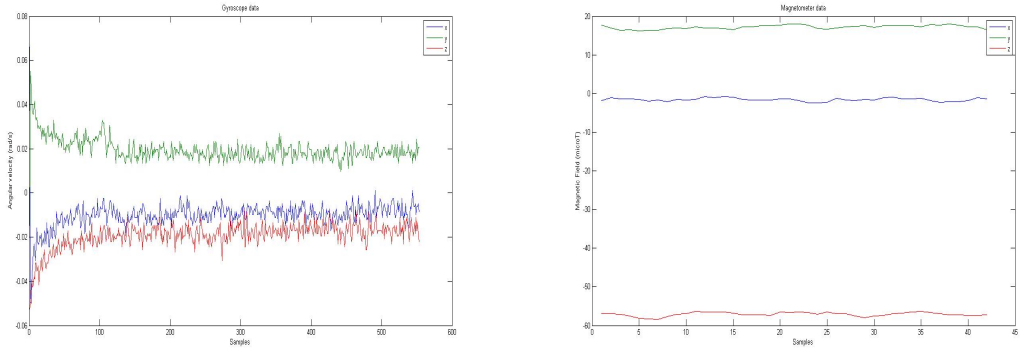
## 4.1 Measurement of Sensory Data for Phone in Stationary Motion

As a baseline or a point of reference and for calibration data was collected from the device when it was stationary. We can observe from the figure 7b of raw data for acceleration that the effect of acceleration due to gravity can be observed along the z axis axis. In the case of the gyroscope we observe that there is a slight tremor initially which is a remnant of the user pressing the start button. The data from the magnetometer and pressure sensor are constant.



**(a)** *Barometer Signal.* **(b)** *Accelerometer Signal.*

**Figure 7:** *Measured time-domain signals from barometer and accelerometer for the stationary case.*



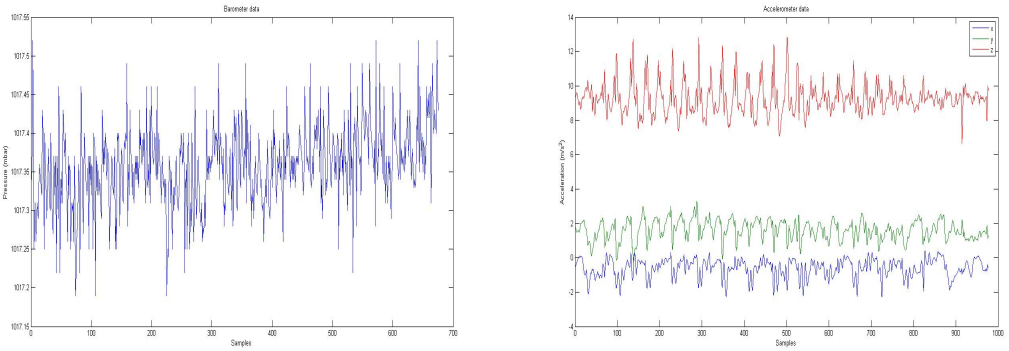
**(a)** *Gyroscope signal.* **(b)** *Magnetometer signal.*

**Figure 8:** *Measured time-domain signals from gyroscope and magnetometer for the stationary case.*

## 4.2 Measurement of Sensory Data for Activity Walking

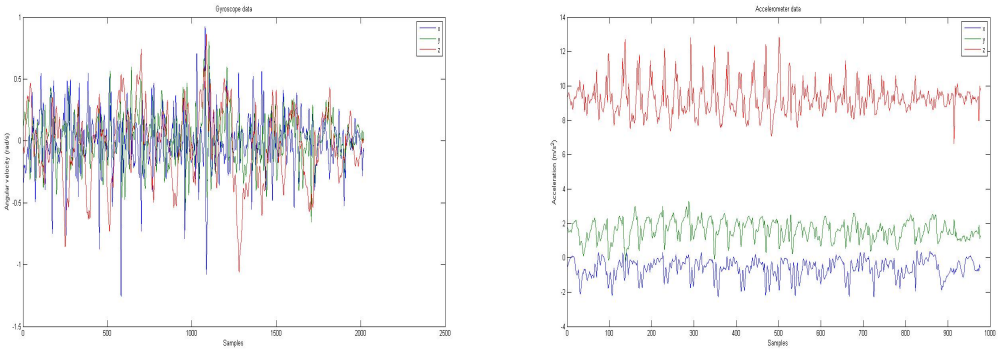
We can observe from the figure of raw data for acceleration that the effect of acceleration due to gravity can be observed along the $z$-axis. In addition, we observe a jerky motion that is characteristic of a person moving. along the $y$-axis we can observe a small acceleration which is an indication of a person moving forward. The acceleration along the $x$-axis is relatively close to zero indicating relatively small lateral/horizontal movement. The gyroscope readings show a rapid change of orientation. The magnetometer is comparatively stable in comparison except for when the person turns back. The pressure readings are constant because the test

subject is walking along a relatively flat surface.



**(a)** *Barometer signal.* **(b)** *Accelerometer signal.*

**Figure 9:** *Measured time-domain signals from barometer and accelerometer for the walking case.*



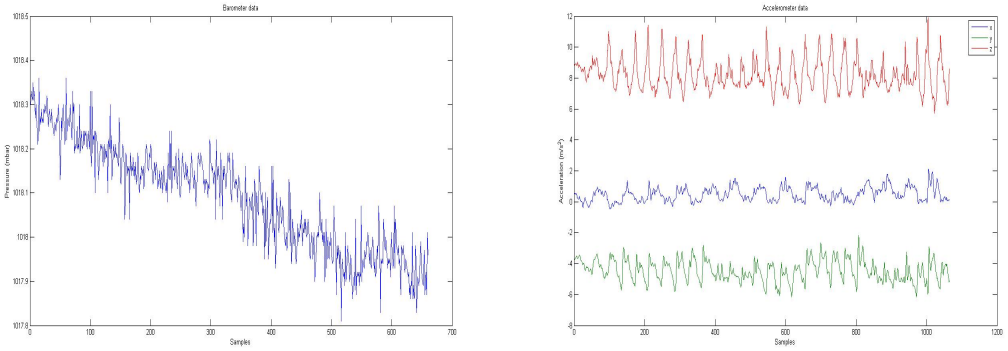
**(a)** *Gyroscope signal.* **(b)** *Magnetometer signal.*

**Figure 10:** *Measured time-domain signals from gyroscope and magnetometer for the walking case.*

## 4.3 Measurement of Sensory Data for Activity Climbing Stairs

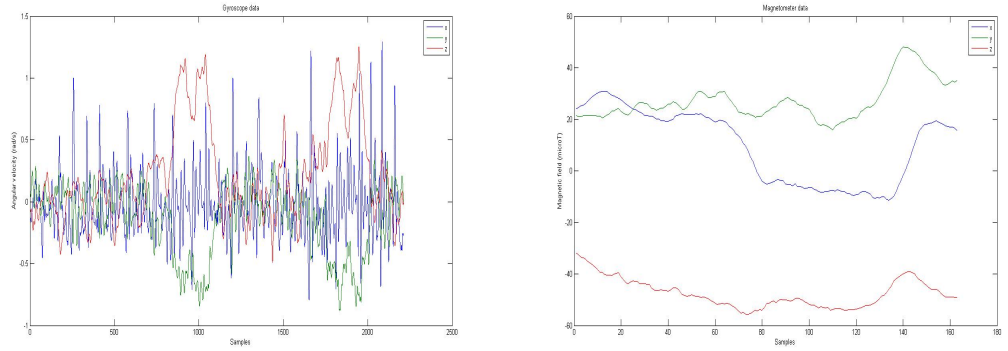
From the figure 11b on the raw data for acceleration, in addition to the effect of acceleration due to gravity, we can observe a jerky motion that is characteristic of a person moving and also a person turning while climbing the stairs. The plot gyroscope data makes the fact that the person was turning obvious. The

magnetometer reading also helps us make this inference. The pressure reduces by 1 mbar indicating a clear change in height.



**(a)** *Barometer signal.* **(b)** *Accelerometer signal.*

**Figure 11:** *Measured time-domain signals from barometer and accelerometer for the climbing up stairs case.*

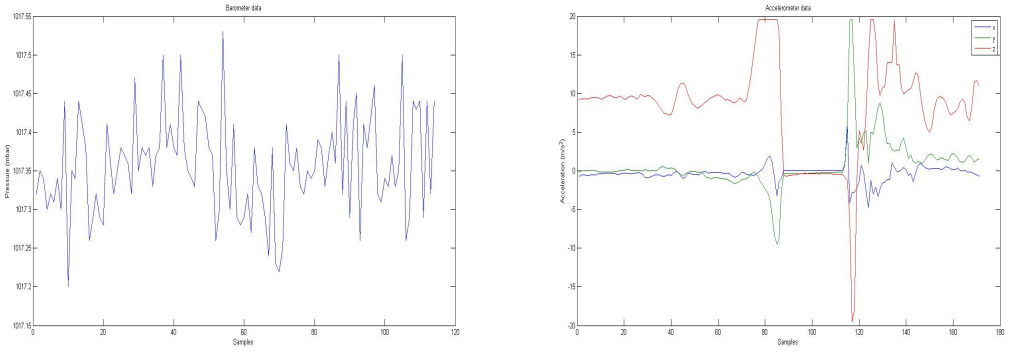


**(a)** *Gyroscope signal.* **(b)** *Magnetometer signal.*

**Figure 12:** *Measured time-domain signals from gyroscope and magnetometer for the climbing up stairs case.*

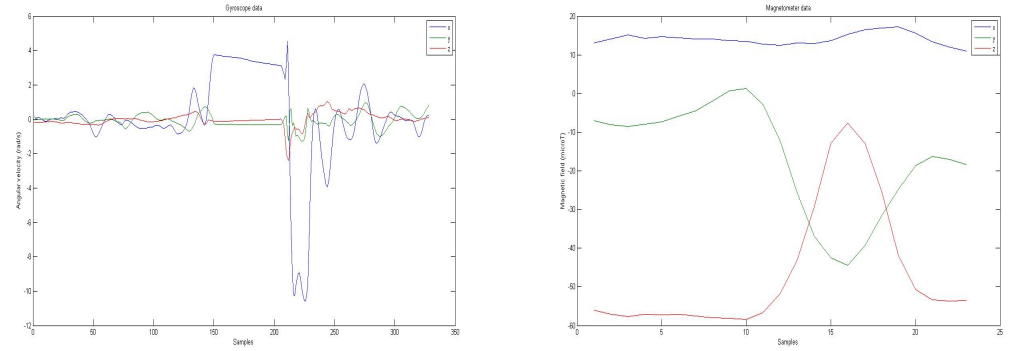
## 4.4 Measurement of Sensory Data for Phone in Flight Motion

As the phone is thrown up there is a massive increase in acceleration, then the phone decelerates, comes to a stop and then accelerates downwards continuously until it comes to rest in the hands of the test subject. There are dramatic changes in the gyroscope readings caused by the device turning on its axis. The change is less dramatic in the case of magnetometer. The barometric sensor is relatively stable.




**(a)** *Barometer signal.* **(b)** *Accelerometer signal.*

**Figure 13:** *Measured time-domain signals from barometer and accelerometer when the phone is thrown vertically upwards.*

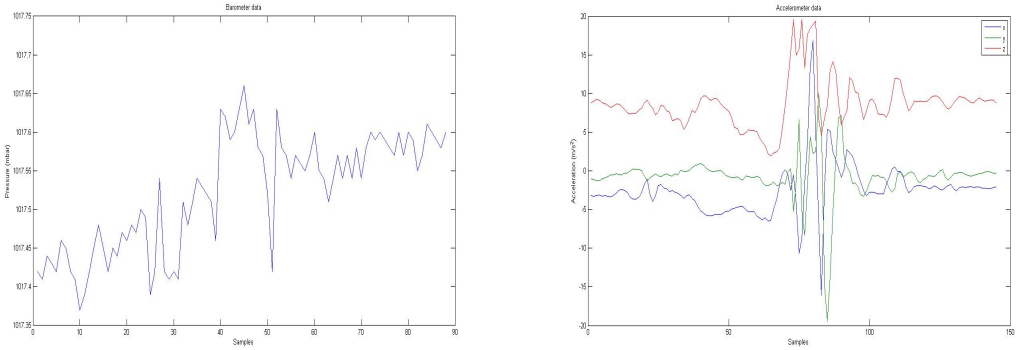



**(a)** *Gyroscope signal.* **(b)** *Magnetometer signal.*

**Figure 14:** *Measured time-domain signals from gyroscope and magnetometer when the phone is thrown vertically upwards.*

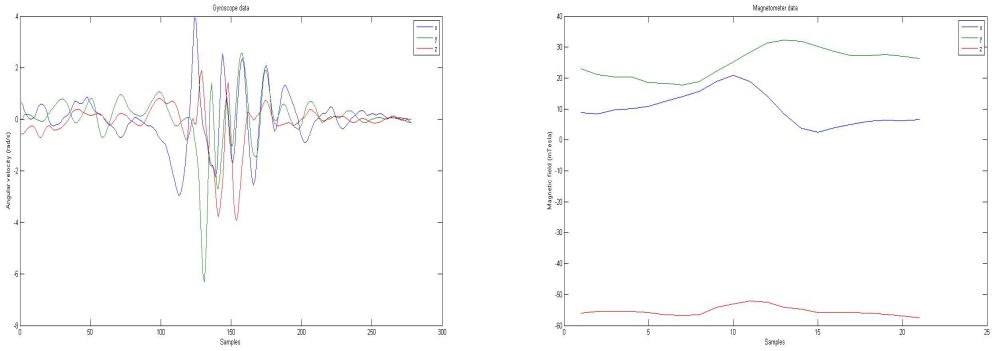
## 4.5 Measurement of Sensory Data for Activity Falling Down

In order to detect falls in elderly people, we performed multiple tests falling forward, falling backward and falling sideways. All falls are unique, however, we observe certain patterns. In both the forward and backward cases, there is a decrease in the acceleration at the start of the fall followed by a steep increase. The gyroscope shows a dramatic perturbation. There about 0.1 mbar change in the pressure reading.




**(a)** *Barometer signal.* **(b)** *Accelerometer signal.*

**Figure 15:** *Measured time-domain signals from barometer and accelerometer when the user falls down forward.*

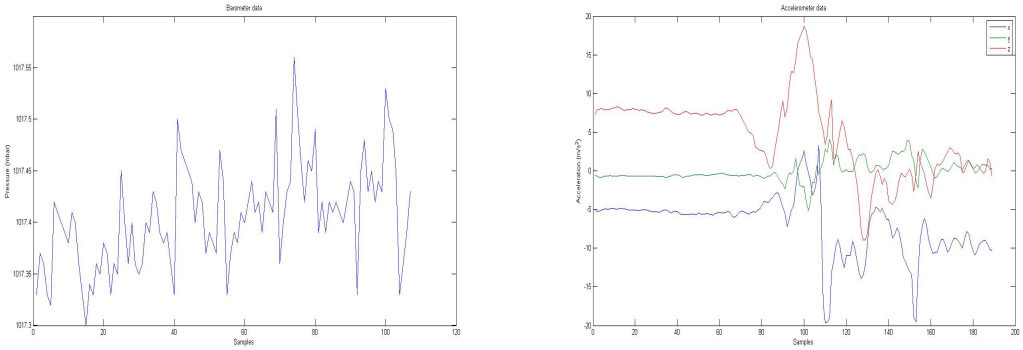



**(a)** *Gyroscope signal.* **(b)** *Magnetometer signal.*

**Figure 16:** *Measured time-domain signals from gyroscope and magnetometer when the user falls down forward.*

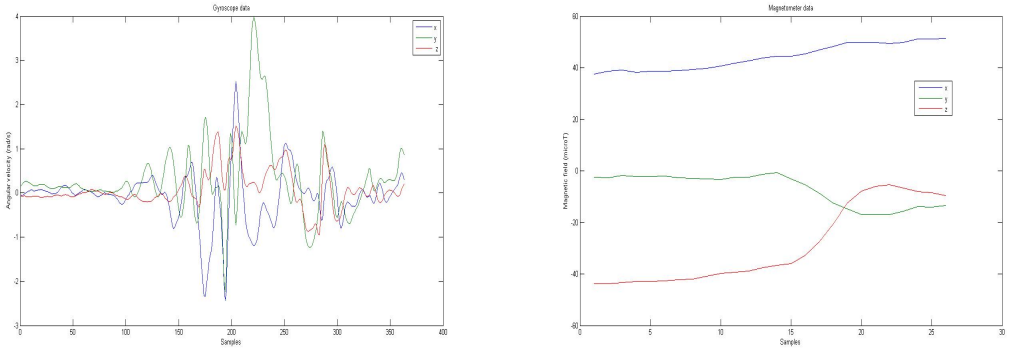
## 4.6 Measurement of Wifi Signal Strength for Phone in Motion

The Android Wi-Fi manager returns a list of Wifi networks after scanning. The output includes the following parameters, i.e., SSID (network name), BSSID (MAC Address of access point), RSSI (received signal strength in dB). The following plots show a variation in signal strength measured while walking and climbing stairs. The signal strength varies due to reflections from the walls, ceiling, floor and other obstructions in the building. The signal strength also depends on position of the person making measurements and the people present or moving around in the




**(a)** *Barometer signal.* **(b)** *Accelerometer signal.*

**Figure 17:** *Measured time-domain signals from barometer and accelerometer when the user falls down backwards.*




**(a)** *Gyroscope signal.* **(b)** *Magnetometer signal.*

**Figure 18:** *Measured time-domain signals from gyroscope and magnetometer when the user falls down backwards.*

vicinity. If we have a data base of the locations and coordinates of the access points then it is possible to perform indoor localisation either through location fingerprinting or triangulation.

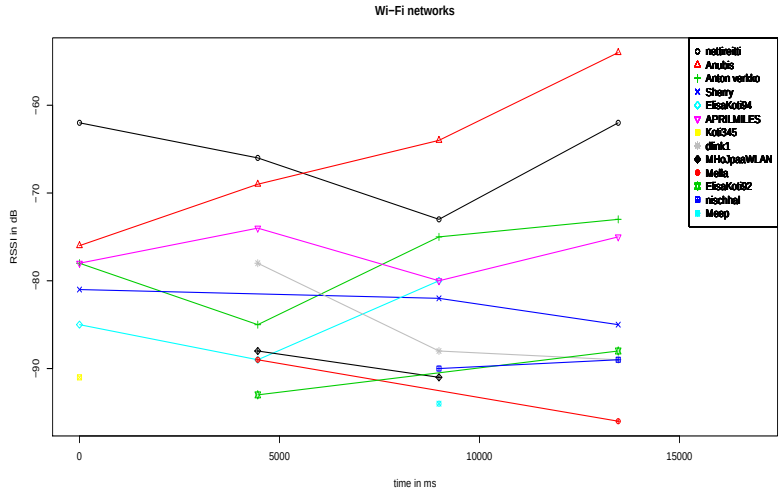



**Figure 19:** *WiFi Networks RSSI detected while climbing stairs.*

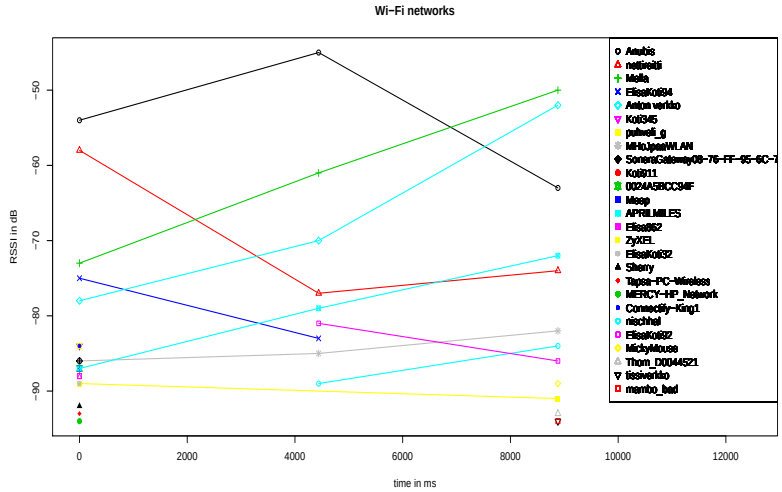



**Figure 20:** *WiFi Networks RSSI detected while walking along the corridor.*

# 5 Algorithms For Fall Detection

In the previous chapter, we present some measurement data of the sensors onboard a mobile smart phone. The measurement data illustrated various scenarios such as when a person is walking, running, in a stationary position and falling down. A cursory glance at the measurement data reveals a distinct pattern for the fall data when compared to other scenarios. For example, in Figs. 21 to 26, we plot the *gradient* of the measurement data corresponding to various scenarios. One can clearly see a distinct pattern corresponding to the fall data in comparison with other scenarios. Taking a gradient of the measurement data has the following advantages: First it eliminates the bias inherent in the measurement data and second its operation is akin to that of a high-pass filtering. This is useful as the action of falling down induces an abrupt change in the state of the sensor data which implies the existence of high-frequency components and, thus, the fall data can be interpreted as a high-pass process.

Our goal in this chapter is to utilize this distinctive pattern of the fall data in relation to other scenarios to develop effective algorithms for detection of falls. Specifically, the algorithms should be easily implementable on the mobile platform, for e.g., a mobile smart phone. The algorithms should keep track of its surroundings and when a fall occurs it must detect, within a certain acceptable time duration, if an actual fall has occurred. In this chapter, we use two well known mathematical methods for fall detection: The first method uses the prowess of statistical detection theory while in the second method we cast fall detection as a geometrical classification problem [35, 36, 37, 38].

## 5.1 Detection Theory

Detection theory is a mathematical technique used to distinguish between the desired signal of interest and undesired noise[39, 40, 41]. Specifically, the problem the signal detection problem is formulated as a *binary hypothesis testing problem* (BHT) [35]. In BHT problem, we define the so called null hypothesis, designated by $H_0$, which describes the case when there is only noise and the alternate hypothesis, designated by $H_1$, denotes the case which contains the signals of interest. This is shown below:

$$H_0 : r(n) = w(n), \tag{1}$$

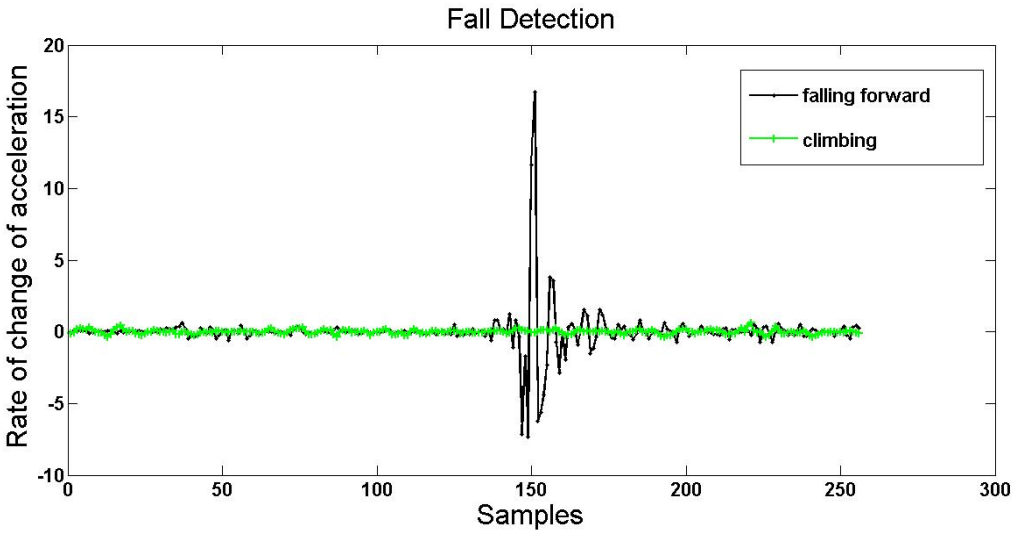



**Figure 21:** *Comparison of gradient of accelerometer data for fall and climbing stairs case.*

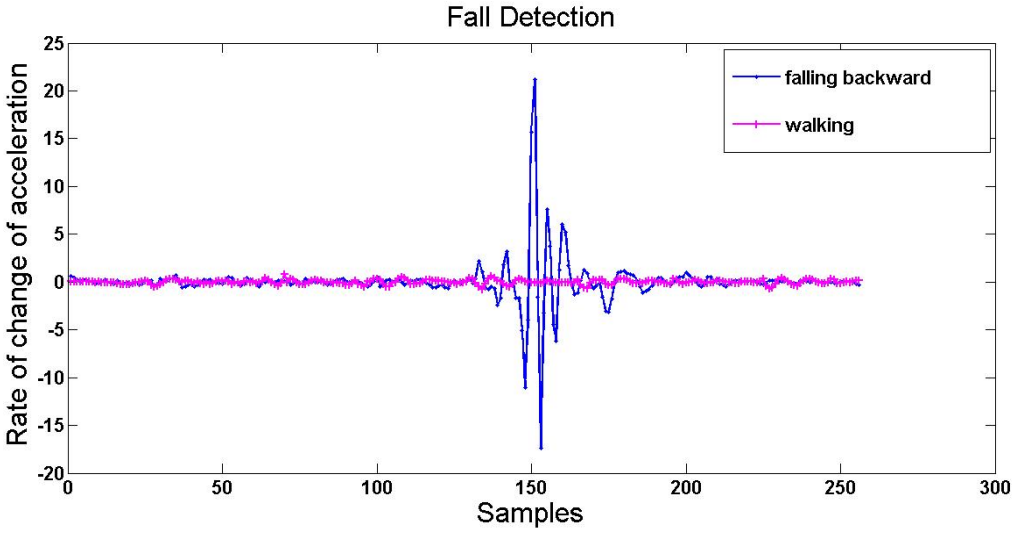



**Figure 22:** *Comparison of gradient of accelerometer data for fall and walking case.*

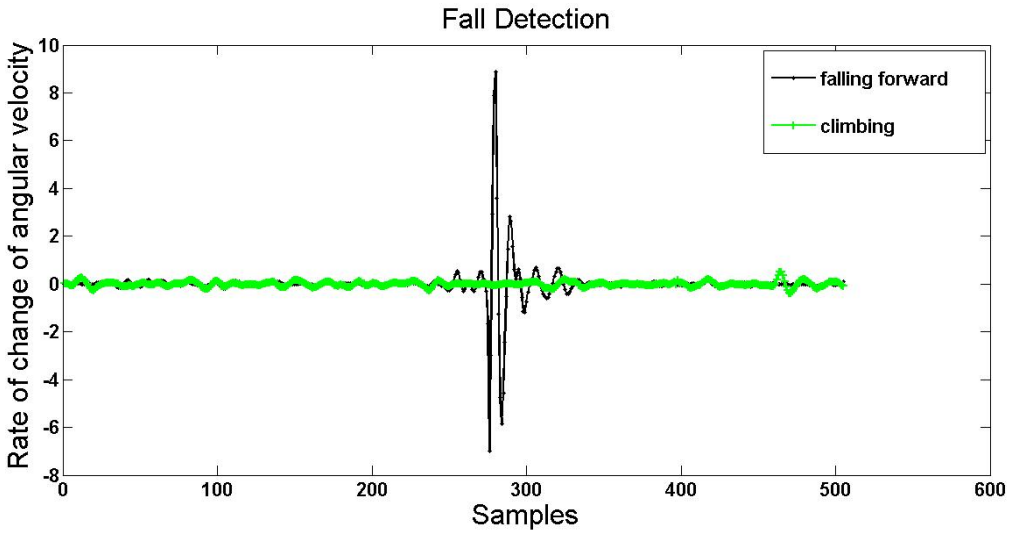



**Figure 23:** *Comparison of gradient of gyroscope data for fall and climbing stairs case.*

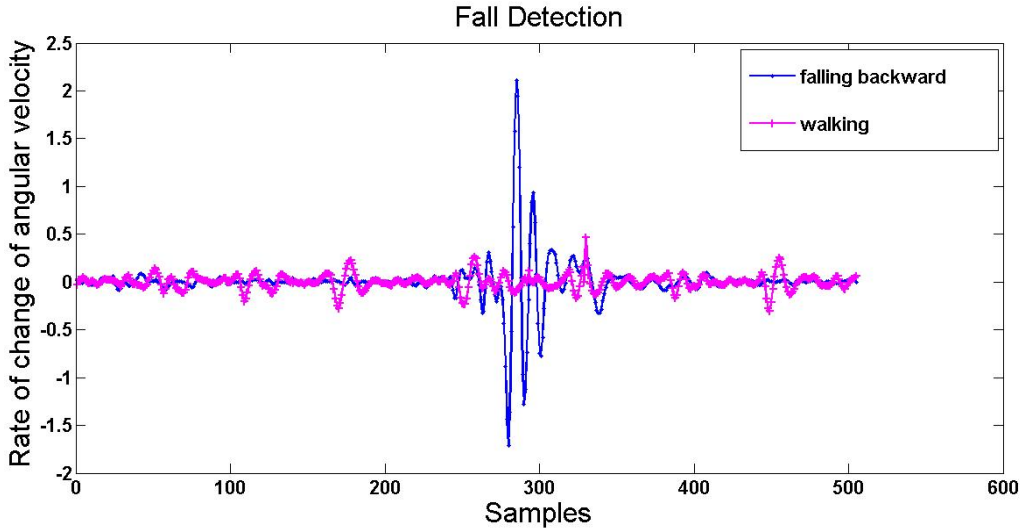



**Figure 24:** *Comparison of gradient of gyroscope data for fall and walking case.*

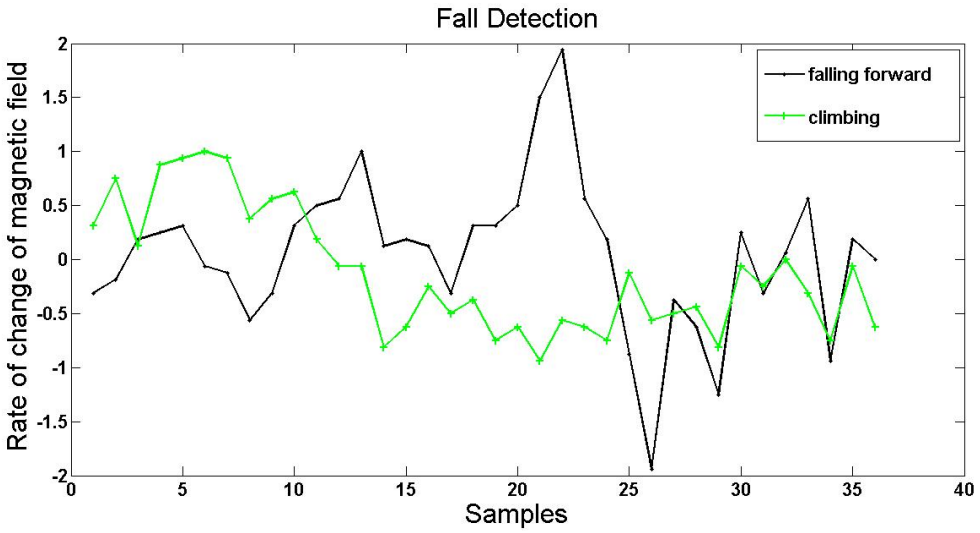



**Figure 25:** *Comparison of gradient of magnetometer data for fall and climbing stairs case.*

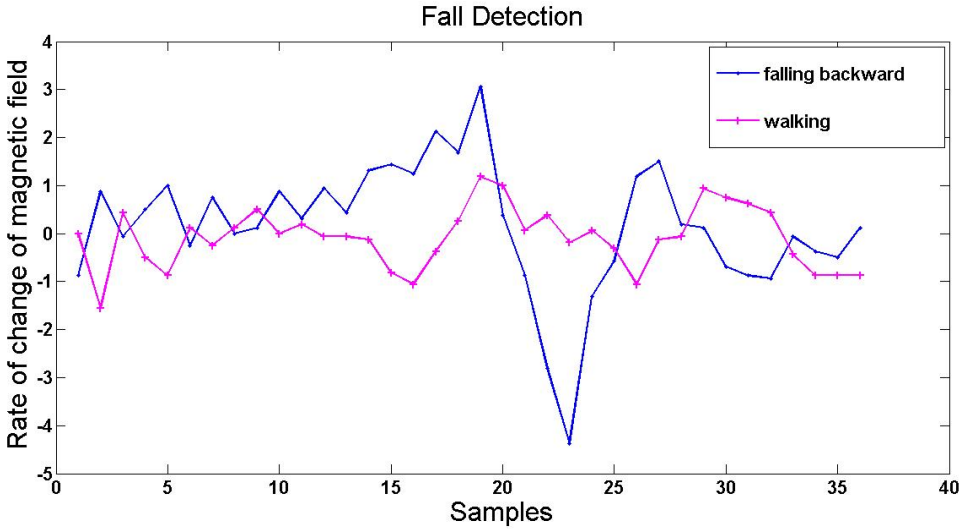



**Figure 26:** *Comparison of gradient of magnetometer data for fall and walking case.*

$$H_1 : r(n) = s(n) + w(n), \tag{2}$$

where $r(n)$ denotes the observed signal, $w(n)$ denotes inherent noise in the system and $s(n)$ denotes the signal to be detected. We assume a discrete-time system without any loss in generality. We can cast the fall detection problem as BHT problem wherein the $s(n)$ corresponds to the fall data while $w(n)$ can be classified as the unwanted non-fall data. Given that we observe $r(n)$, our task is to determine under which case does $r(n)$ belong to. A practical signal detector achieves this objective by evaluating something known as a *test statistic*. It is generically given by

$$T = f\left(r(n)\right) \tag{3}$$

where $f\left(\cdot\right)$ is some given function. The hypothesis is then selected by comparing $T$ to a set value of threshold as follows:

$$T \underset{H_0}{\overset{H_1}{\gtrless}} \eta \tag{4}$$

Each time a decision is made, four possible cases can occur:

1. Decide $H_0$ when $H_0$ is true - Correct rejection
2. Decide $H_0$ when $H_1$ is true - Missed detection also called type II error
3. Decide $H_1$ when $H_0$ is true - False alarm also called Type I error
4. Decide $H_1$ when $H_1$ is true - Correct detection

Thus, as we can see, the performance in terms of missed detection or correct detection depends on the kind of test static $T$ which in turn depends on various quantities, for example, on the choice of $f(\cdot)$, on the value set for $\eta$, the signal-to-noise-ratio, the underlying distributions of $r(n)$ under $H_0$ and $H_1$ respectively. A simple demonstration of this fact is shown in Fig. 27. In the figure, we plot the distributions of $T$ under $H_0$ and $H_1$. If these distributions overlap then we can expect, for a set value of threshold, a non-zero probability of false alarm and missed detection as seen in the figure. Thus, one measure of a good detector is its ability to separate as much as possible the distributions of $T$ under $H_0$ and $H_1$ respectively.

In the following subsection, we discuss the classical energy detector which we shall use for fall detection.

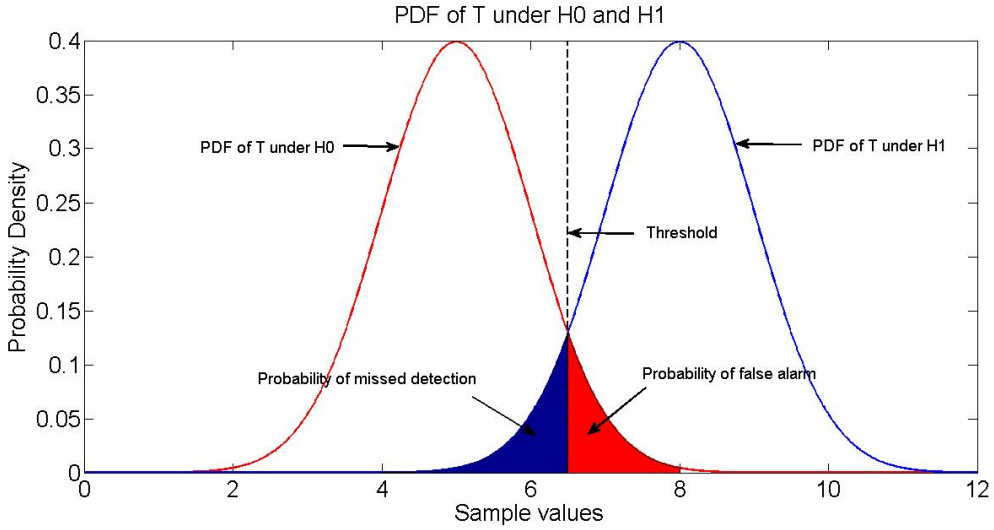



**Figure 27:** *PDF of $T$ under $H_0$ and $H_1$*

## 5.2 Classical Energy Detector

Energy detector measures the received signal energy and compares it with a threshold [42, 43, 44]. The following signal model is used

$$r(n) = s(n) + w(n). \tag{5}$$

The energy based detector computes the signal energy as follows

$$T = \frac{2}{N_0} \sum_{n=1}^{M} |r(n)|^2, \tag{6}$$

where, $r(n)$ is the received signal, $N_0$ is the noise power and $M$ is the number of observations. Consider the detection of a deterministic signal in the presence of zero mean i.i.d Gaussian noise; the energy detector test statistic obeys the following distribution [43]

$$H_0 : T \sim \chi^2_{2M}, \tag{7}$$

$$H_1 : T \sim \chi^2_{2M}(2\gamma), \tag{8}$$

where, $\gamma$ is the signal energy to noise spectral density defined as $\gamma = E_s/N_0$, where $E_s = \sum_{n=1}^{M} |s(n)|^2$ is the signal energy. The test statistic follows a central chi-square distribution with 2M degrees of freedom under $H_0$ and non-central chi-square

distribution with 2M degrees of freedom and non-centrality parameter $2\gamma$ under $H_1$. The thresholds for a given probability of false alarm are calculated theoretically using the equation

$$\eta = F^{-1}_{\chi^2_{2M}}(1 - P_f) \tag{9}$$

where, $F^{-1}(\cdot)$ denotes the inverse cumulative distribution function of a $\chi^2$ distributed random variable with 2M degrees of freedom, $\eta$ is the detection threshold and $P_f$ is the probability of false alarm. The theoretical probability of detection is given by the equation

$$P_d = 1 - F_{\chi^2_{2M}}(2\gamma) \tag{10}$$

where, $\gamma$ is the signal energy to noise spectral density defined as $\gamma = E_s/N_0$, where $E_s = \sum_{n=1}^{M} |s(n)|^2$ is the signal energy and $N_0$ is the noise power.

To summarize, the classical energy detector implements the test statistic using (6) and compares it to the the threshold $\eta$ which is set using (9). The threshold is set in accordance with a desired probability of false alarm.

## 5.3 Empirical Fall detector

We cast fall detection as BHT problem and specifically use the observed signal energy as a test statistic. Specifically, this is given by

$$T = \frac{1}{N} \sum_{n=1}^{N} r(n)^2 \tag{11}$$

where $N$ denotes the observation interval and $r(n)$ is the (processed) sensor signal. The main task to set the threshold $\eta$ which in turn must be related to the desired probability of false alarm. In the classical energy detector case, we know this relation which is given by (9). The equation is derived based on the assumption that $r(n)$ is Gaussian under both $H_0$ and $H_1$. We, thus, need to verify if the (processed) sensor data also follows the Gaussian distribution.

Consider first the accelerometer sensor data. In Fig. 28 we plot the empirical probability density function (PDF) of $r(n)$ under fall ($H_1$) scenario and Fig. 29 plot the non-fall ($H_0$) case. We compare the empirical PDF with its Gaussian approximation. This is obtained by calculating the mean and variance of the data and then plotting a Gaussian distribution with these parameters. Clearly from the figures, we can conclude that the accelerometer data is non-gaussian under both fall

and non-fall scenarios. Similar conclusion can be reached for the gryoscope data whose empirical PDFs are shown in Fig. 33 and 34. The pattern is similar for the other directions in the case of both sensors.

The non-Gaussianity of the sensor data renders non-applicability of the classical energy detector for the mobile sensory data. We overcome this problem by employing an empirical approach for fall detection. Specifically, we do the following: First, we empirically evaluate the distribution of the test statistic $T$ from the measurement (training) data under both null and alternate hypothesis. Secondly, depending on the desired false-alarm rate, we empirically calculate the desire threshold value of $\eta$. Such an approach is valid when enough measurements are taken for characterizing the true PDF of $T$ under both fall and non-fall cases.

Figures 30 to 32 show the empirical PDF of $T$ for an observation interval of $N = 10$ samples for accelerometer sensory data. The plots show the PDF under both the null (non-fall) and alternate (fall) hypothesis. In the figures, the threshold is shown by the dashed vertical black line. The threshold is calculated for a set probability of false alarm which is set to a value of 0.05. The PDFs corresponding to the gyroscope data are shown in Figs. 35 to 37. From the figures, we see good separation between the PDFs of $T$ under $H_0$ and $H_1$, thus, allowing freedom in choosing a threshold corresponding to small false alarm rate and high probability of detection. Thus, the test-statistic based on the energy detection principle provides good performance for detection of falls. We can also combine the sensory data from each individual direction and compare it against a set value of threshold as follows:

$$T = x^2 + y^2 + z^2 \tag{12}$$

where $x$, $y$ and $z$ denote, respectively, the x, y and z directions of the accelerometer and gyroscope. Figures 38 and 39 show the empirical PDF of the test statistic of (12) under null and alternate hypothesis. Consistent with the time-domain approach, the spatial energy detector of (12) also separates well the PDFs of the null and alternate hypothesis allowing for the design of small false alarm rate.

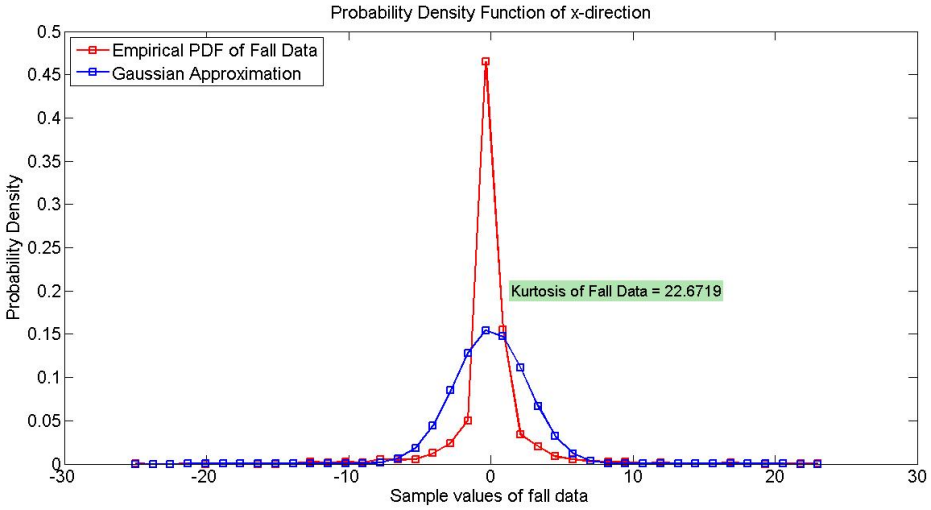



**Figure 28:** *Comparison of empirical PDF of fall data with Gaussian approximation along x-direction*

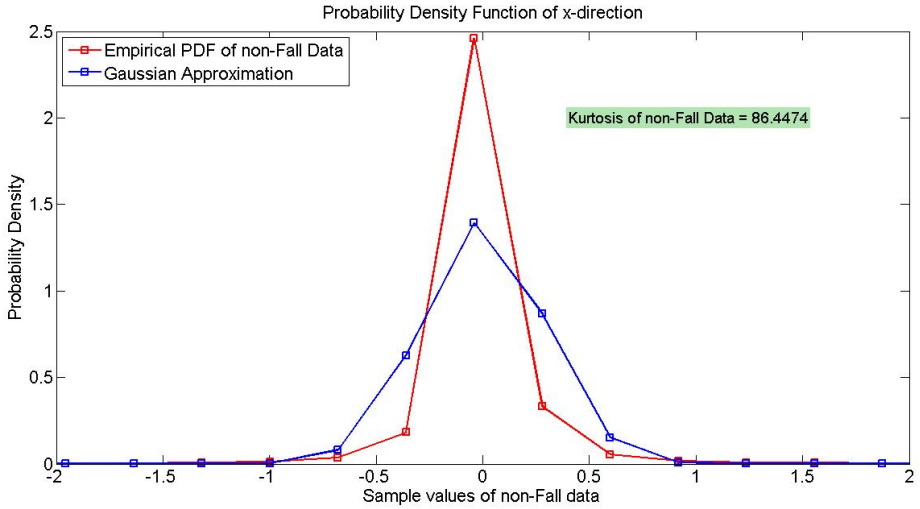



**Figure 29:** *Comparison of empirical PDF of non fall data with Gaussian Approximation along x direction*

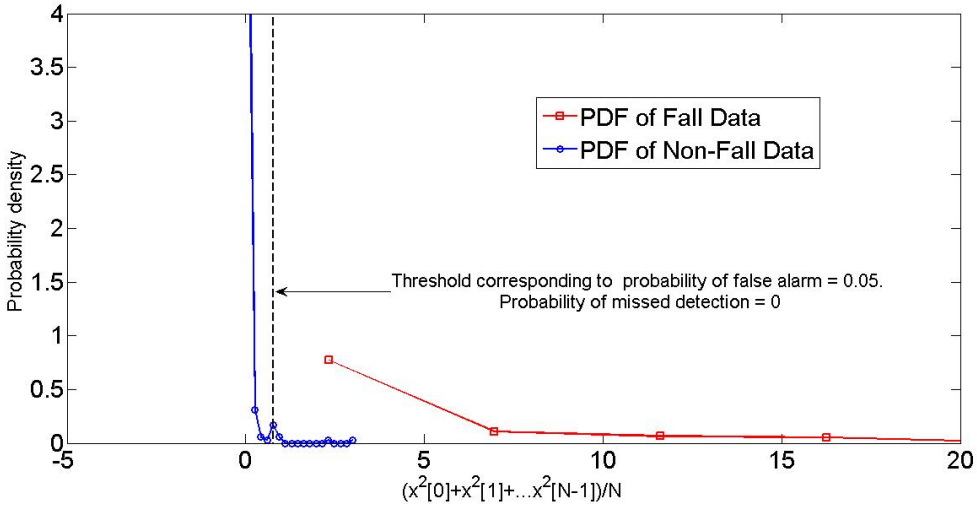



**Figure 30:** *Empirical PDF of test statistic along x-direction*

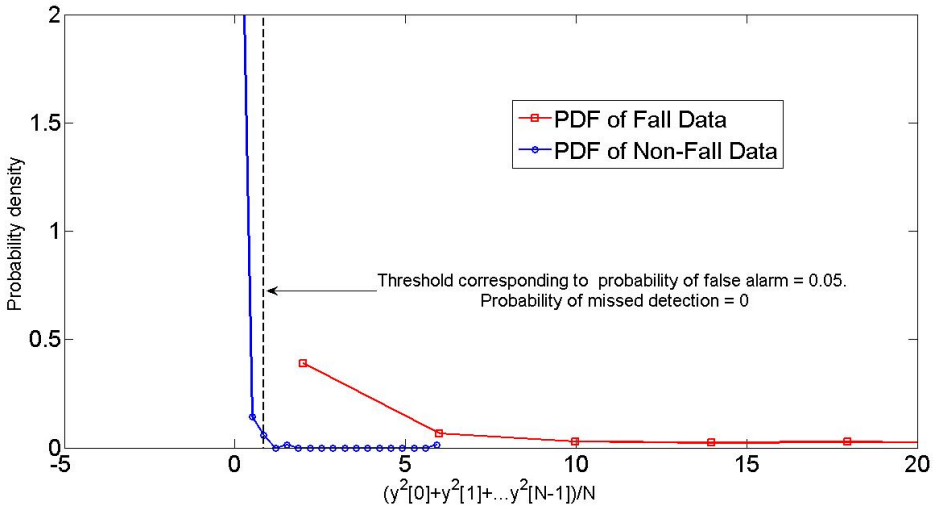



**Figure 31:** *Empirical PDF of test statistic along y-direction*

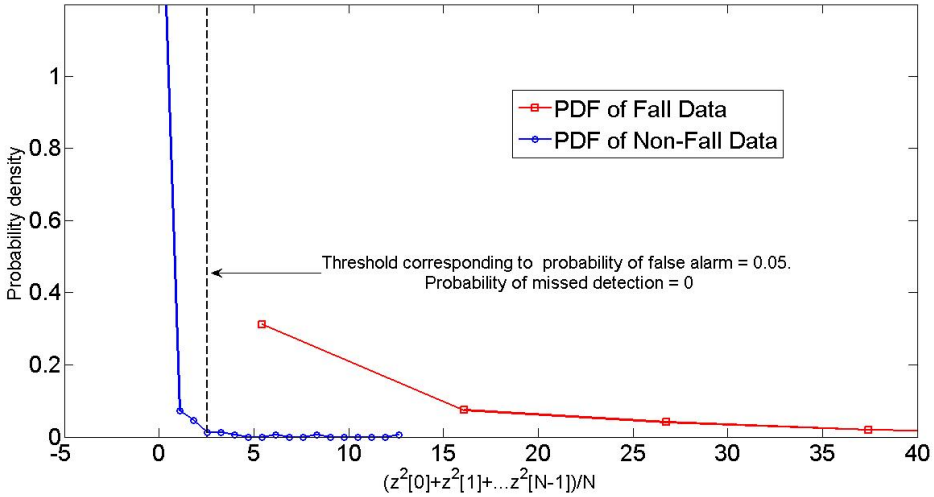



**Figure 32:** *Empirical PDF of test statistic along z-direction*

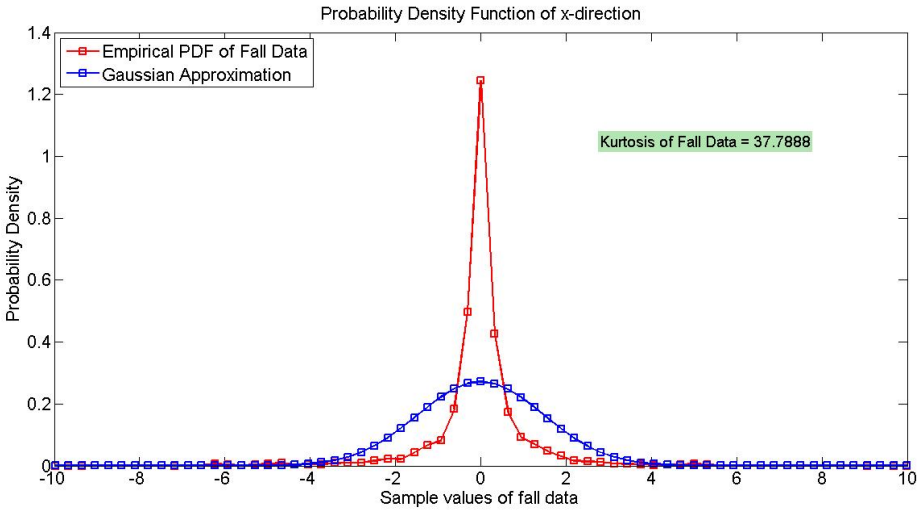



**Figure 33:** *Comparison of empirical PDF of fall data with Gaussian approximation along x-direction*

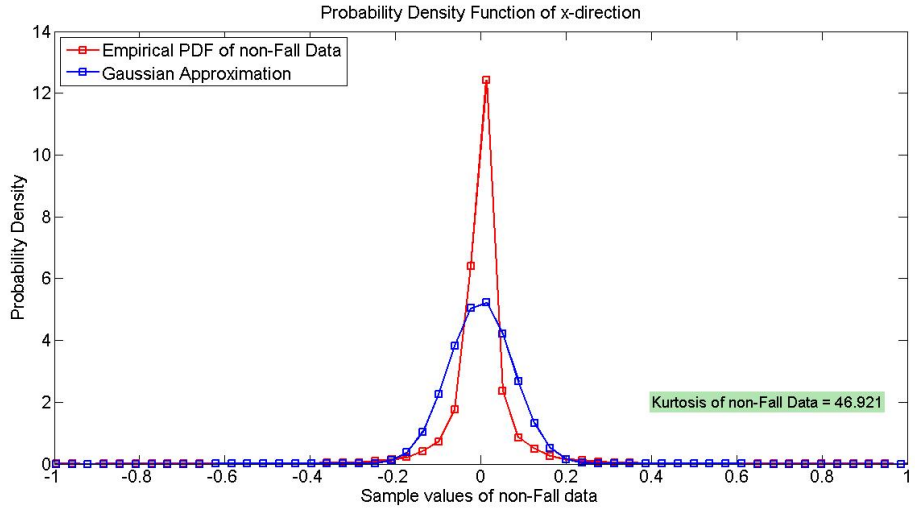



**Figure 34:** *Comparison of empirical PDF of non fall data with Gaussian approximation along x-direction*

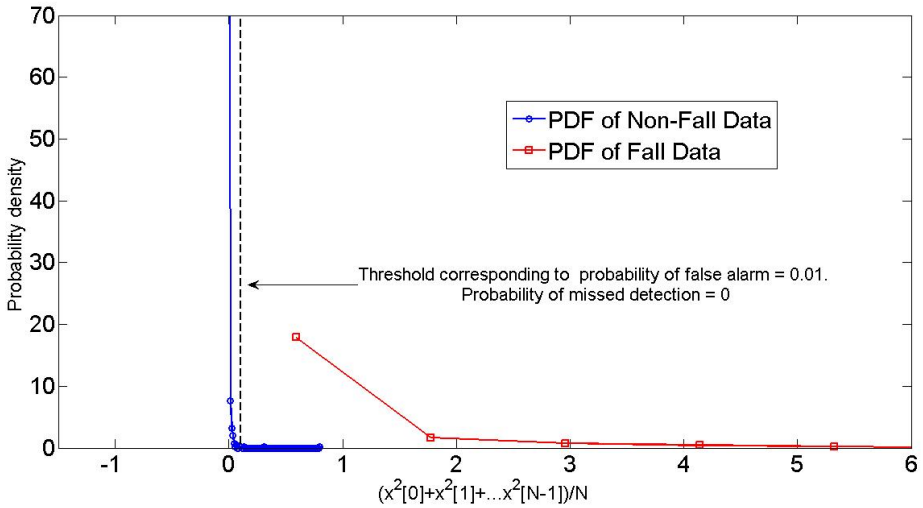



**Figure 35:** *Empirical pdf of test statistic along x-direction*

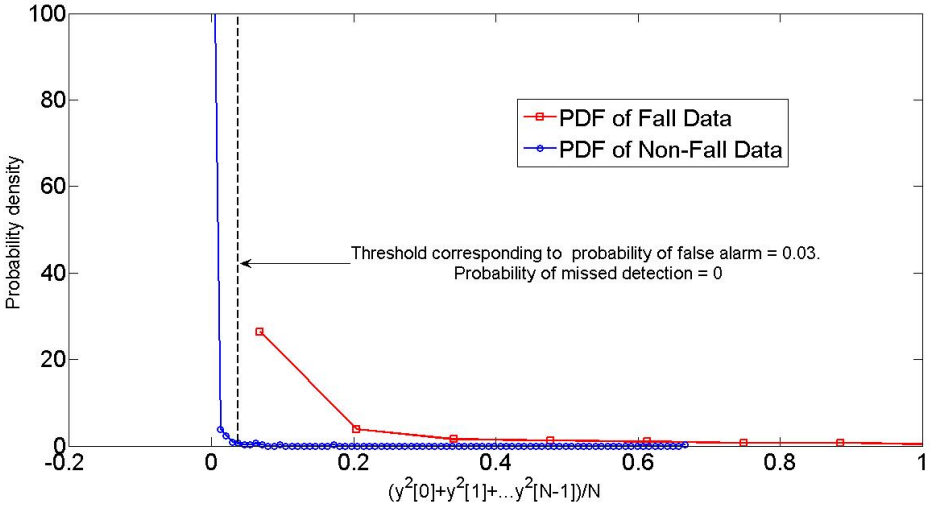



**Figure 36:** *Empirical pdf of test statistic along y-direction*

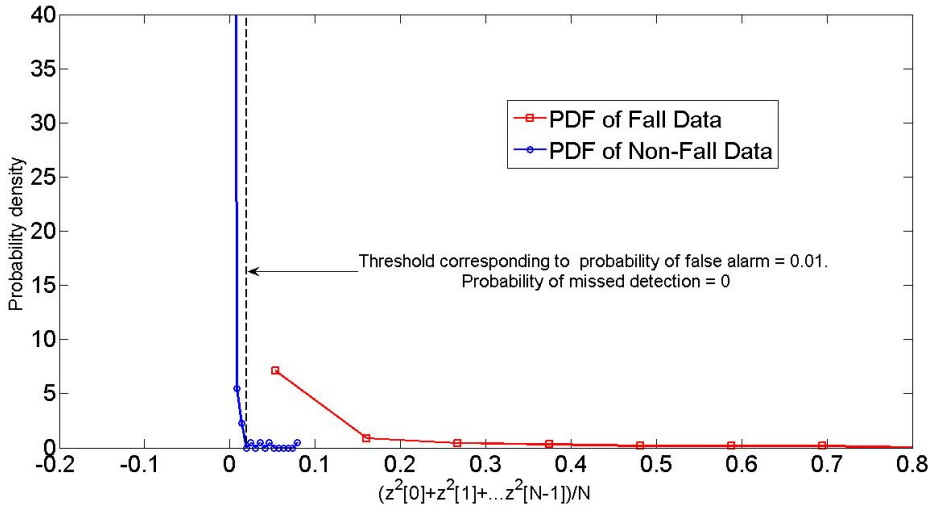



**Figure 37:** *Empirical pdf of test statistic along z-direction*

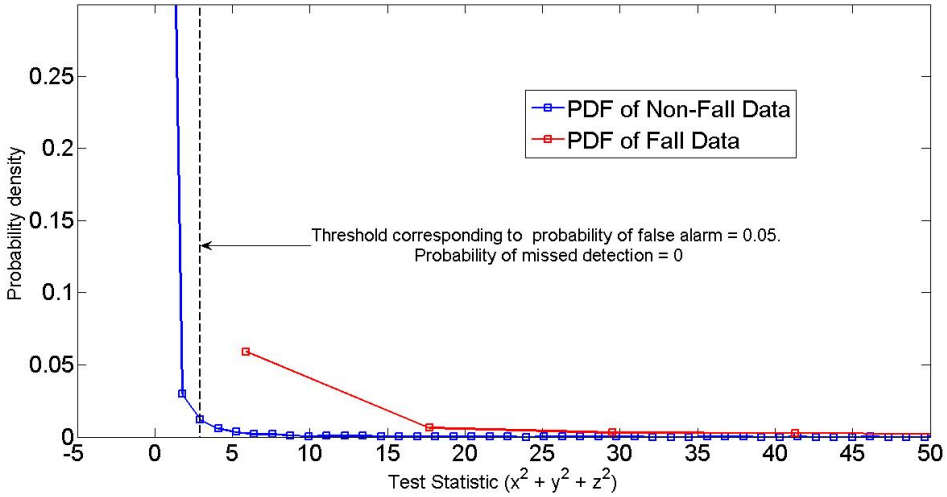



**Figure 38:** *Empirical PDF of test statistic for accelerometer*

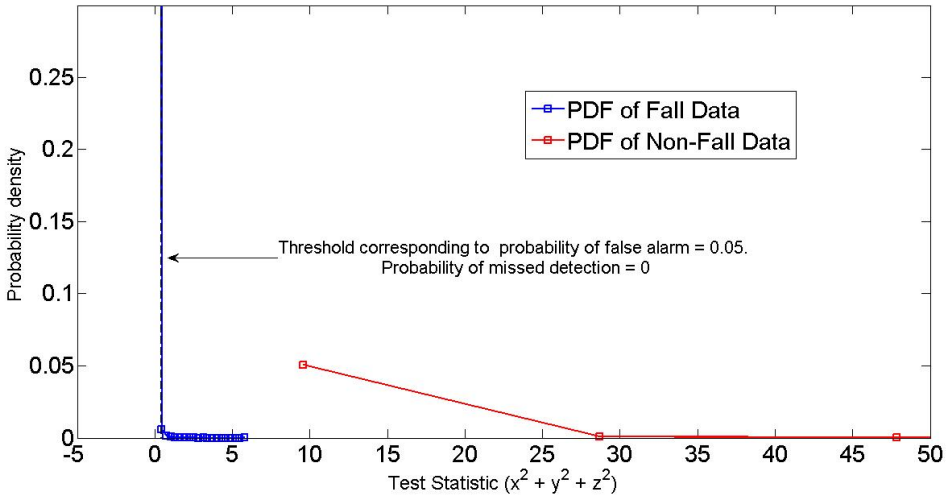



**Figure 39:** *Empirical pdf of test statistic for gyroscope*

## 5.4 Fall Detection as a Geometrical Classification Problem

In this section, we use the spatial nature of the measurement data to detect falls. We can interpret the phenomena of a fall as a sudden change in the state of the sensory data. Thus, for example, when we plot the accelerometer data on the Euclidean coordinate space, we can expect the points corresponding to a fall be scattered far away from its normal state which is the non-fall case. Figure 40 show cases such a

scenario. In the figure, we plot the accelerometer x-direction versus its y-direction. From the figure, we can clearly see a spatial separation between the fall points and the non-fall points. Based on this data set, we can design a geometric object that automatically classifies fall and non-fall points. For example, we can put a circle around the non-fall points and state that all points that fall inside the circle correspond to non-fall data while fall data lies outside of this circle [45]. Given that we choose a type of geometric object for classification, for example a second-order classifier, the following question that arises is which is the best second-order classifier such that the percentage of false classification is minimized. We address this problem in the rest of this section. First, we briefly summarize the subject of classification after which we design a quadratic classifier for fall detection.

## 5.5 Pattern Recognition or Classification

Pattern recognition, machine learning, data mining etc [46, 47, 48, 49] are terms that are largely synonymous and generally refer to techniques used to solve problems in a variety of engineering and scientific disciplines as diverse as biology, psychology, medicine, marketing, computer vision, artificial intelligence, remote sensing and of course signal processing. But that raises the question how do you define a pattern? The reference [50] defines a pattern as "opposite of a chaos; it is an entity, vaguely defined, that could be given a name." Some examples of patterns are a human face, a speech signal or handwritten letter/digit.

Once we are given a pattern or data set , the job of recognition or classification may consist of one of the following two tasks [47, 51]

- supervised classification - in which the input pattern is identified as a member of a predefined class where the class is defined by the system designer.
- unsupervised classification - in which the pattern is assigned to an unknown class where the class is learned based on the similarity of patterns.

Classification is an approach to solving a class of problems which involves defining a large set of N digits $x_1, ..., x_N$ called a training set which is then used to tune the parameters of an adaptive model. We assign categories to the digits by using a target vector $t$, which represents the identity of the corresponding digit. When the training data comprises of input vectors along with their corresponding target vectors, the problem is known as a supervised learning problem. On the other hand when the

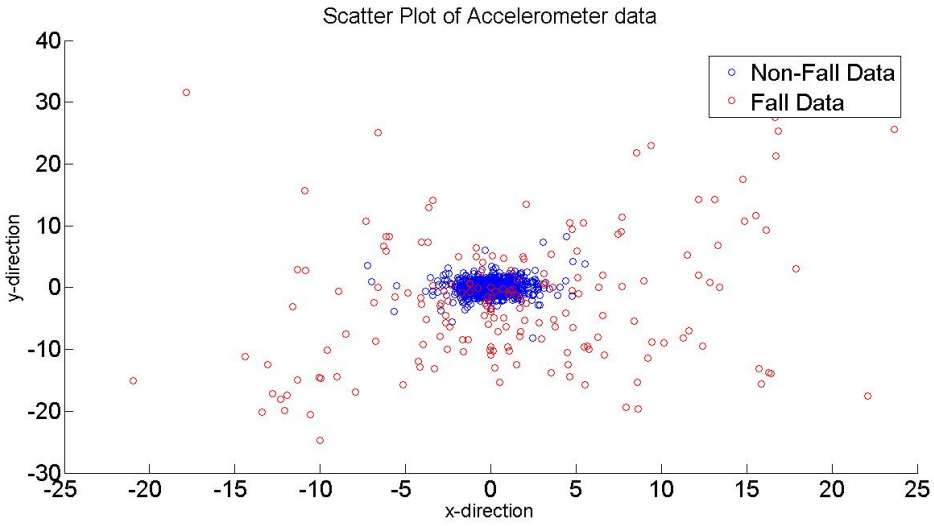



**Figure 40:** *Scatter plot of accelerometer data along x y directions*

training data consists of a set of input vectors x without any corresponding target values the problem is known as an unsupervised learning problem and the goal in such unsupervised learning problems is to discover groups similar data, otherwise called clusters.

The design of a pattern recognition system essentially involves the following three aspects:

- data acquisition and preprocessing
- data representation
- decision making

The type of problem dictates the choice of sensors, preprocessing technique, representation scheme, and the decision making model. A well-defined and sufficiently constrained recognition problem can be described as one with small intraclass variations and large interclass variations. We can assume that if these conditions are met a simple data representation system and decision making strategy can be developed. Learning from a set of examples called training set is the first step of most pattern recognition systems.

## 5.6 Quadratic Binary Classification

We first summarize a generic quadratic classifier given in [36]. In a classification problem, we are given two sets of points, $x_1, x_2, \ldots, x_N$ and $y_1, y_2 \ldots, y_N$ and we wish to find a function f : $\mathbb{R}^n \to \mathbb{R}$ that is positive on the first set and negative on the second i.e.,

$$f(x_i) > 0, i = 1, ..., N \tag{13}$$

$$f(y_i) < 0, i = 1, ..., M \tag{14}$$

where the function $f(\cdot)$ is a quadratic polynomial given by

$$f(x) = x^T P x + q^T x + r \tag{15}$$

The parameters $P \in \mathbb{S}^n$, $q \in \mathbb{R}^n$ $r \in \mathbb{R}$ must satisfy the inequalities

$$x_i^T P x_i + q^T x_i + r > 0, i = 1, ...., N \tag{16}$$

$$y_i^T P y_i + q^T y_i + r < 0, i = 1, ...., M \tag{17}$$

which is a set of strict linear inequalities in the variables $P$, $q$, $r$. Since $f$ is homogeneous in $P$, $q$, and $r$, so we can find a solution to the strict inequalities by solving the nonstrict feasibility problem given below,

$$x_i^T P x_i + q^T x_i + r \geq 1, i = 1, ...., N \tag{18}$$

$$y_i^T P y_i + q^T y_i + r \leq -1, i = 1, ...., M \tag{19}$$

The separating surface $z|z^T P z + q^T z + r = 0$ is a quadratic surface, and the two classification regions are defined by quadratic inequalities given below:

$$z|z^T P z + q^T z + r > 0 \tag{20}$$

$$z|z^T P z + q^T z + r > 0 \tag{21}$$

Solving this quadratic discrimination problem is the equivalent of determining

if the two sets of points can be separated by a quadratic surface. It is possible to impose conditions on the shape of the separating surface or classification regions by putting constraints on $P$, $q$, and $r$ e.g., we can impose the condition $P \prec 0$ to make the separating surface ellipsoidal. In effect, it implies that we are looking for an ellipsoid that contains all the points $x_1, ..., x_N$ but none of the points $y_1, ..., y_M$.The homogeneity in $P$,$q$,$r$ is used to rewrite the constraint$P \prec 0$ as $P \prec -I$. Thus, the quadratic discrimination problem can be reduced to a Semidefinite Programming feasibility problem that is stated below [52]:

$$\begin{aligned} \text{find} \quad & P, q, r \\ \text{subject to} \quad & x_i^T P x_i + q^T x_i + r \geq 1, i = 1, 2, \ldots, N \\ & y_i^T P y_i + q^T y_i + r \leq -1, i = 1, 2, \ldots, M \\ & P \preceq -I \end{aligned} \tag{22}$$

### 5.6.1 Quadratic Fall Classifier

In this section, we build a quadratic classfier specifically for fall detection. The classifier is based on the framework developed in [53]. In [53], the authors design a linear binary classifier while for our purposes, as can be seen from the scatter plot in Fig. 40, a linear classifier will perform poorly. Our goal is to predict the target class designated by $t$ i.e., $t \in \{1, -1\}$ for a given observed $x$. We assume the vector $x$ to be of dimension equal to 2 for simplicity i.e., $x \in \mathbb{R}^2$. The quadratic classifier we choose is the following:

$$f(x) = x^T A x - 1 \tag{23}$$

where the above classifier is an ellipse defined completely by the matrix $A$. To make the problem simpler, we assume $A$ to be diagonal i.e.,

$$A = \begin{bmatrix} d_1 & 0 \\ 0 & d_2 \end{bmatrix}$$

This has the advantage that we need to determine only two parameters thereby making the analysis simple. We are given $N$ training data points $x_i$ where $i = 1, ...N$. To each data point, we assign its corresponding class $t_i$

$$t_i = \begin{cases} -1 & \text{if } f(x) <= 0, \\ 1 & \text{if } f(x) > 0. \end{cases}$$

To measure the confidence of our classification, we define the so called margin which is given by

$$m_i = t_i f(x_i) \tag{24}$$

A margin $m_i < 0$ indicates $x_i$ is misclassified while $m_i > 0$ implies correct classification. Thus, the goal is to determine the best $A$ matrix such that percentage of misclassification is minimized. In order to do so, we assign a cost for classification errors defined which is given by

$$C(A) = \frac{1}{2}\sum_{i=1}^{N}(m_i - p)^2 \tag{25}$$

Substituting $f(x)$ in the cost function, the cost function can be expressed in terms of the diagonal values $d_1$ and $d_2$as follows:

$$C(A) = d^T W d - c^T d + b \tag{26}$$

where $d = \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}$, $W = \sum_{i=1}^{N} A_i$, $c = \sum_{i=1}^{N} c_i$, $c_i = \begin{bmatrix} ptx_1^2 & ptx_2^2 \end{bmatrix}$ and $A_i$ is of the form

$$A_i = \begin{bmatrix} t^2x_1^4 & x_1^2x_2^2t^2 \\ x_1^2x_2^2t^2 & t^2x_2^4 \end{bmatrix} \tag{27}$$

The cost function in (26) is quadratic in the parameters $d_i$ and subject to constraints of $W$ being positive semidefinite and $d > 0$, we have unique minimizer to the (26). The minimizer is obtained by differentiating (26) with respect to the vector $d$, setting the derivate to zero and solving for $d$ to obtain

$$d = W^{-1}c. \tag{28}$$

Excellent numerical methods exist that can evaluate matrix inverses efficiently [54].

We now present results on the obtained quadratic classifier using the training measurement data from the accelerometer. Figures 41 and 42 show the two classes which are the fall and non-fall-data. The fall data is classified under the target class of $t_i = 1$ while the non-fall data class is $t_i = -1$. The quadratic classifier is obtained using the approach discussed above and is shown by the black elliptical curve in the figures. A cursory glance indicates a good separation between these two classes. For this training data set, the percentage of miss classification is 12 percent.

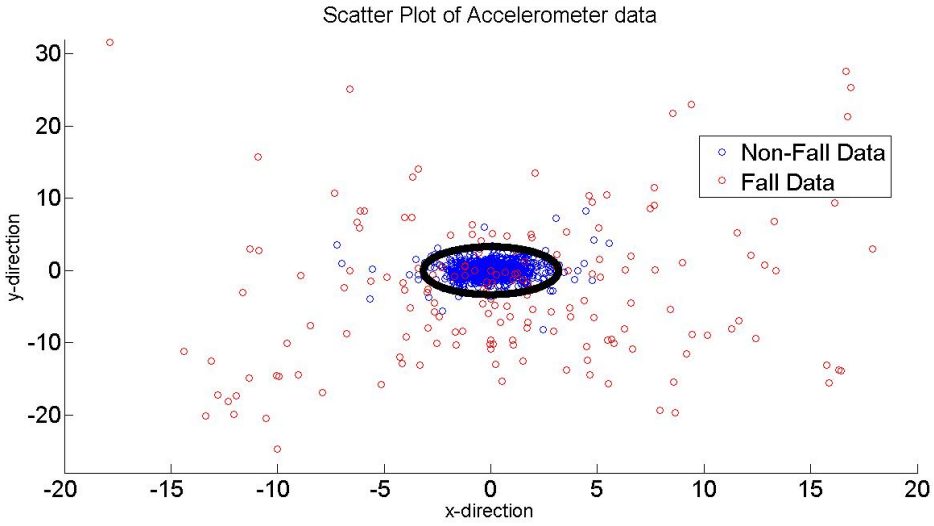



**Figure 41:** *Scatterplot of quadratic classifier along x y directions*

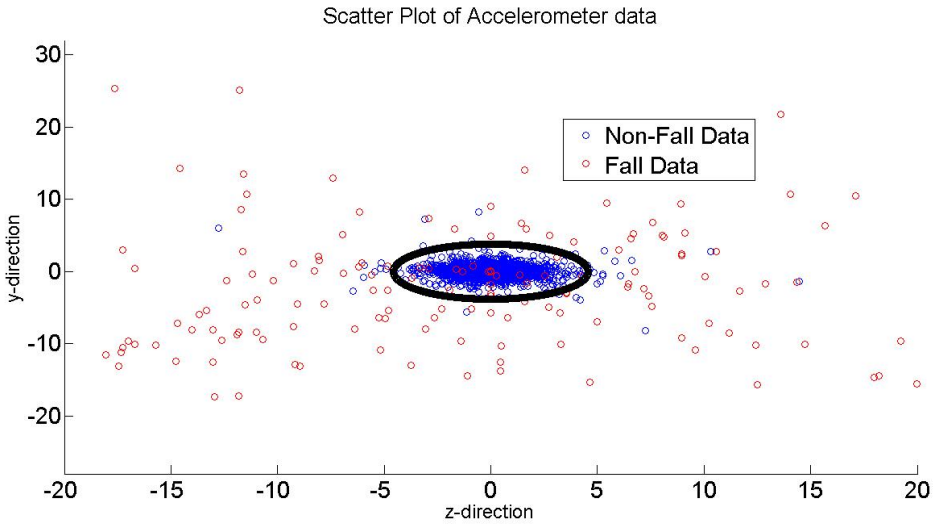



**Figure 42:** *Scatterplot of quadratic classifier along y z directions*

# 6 Summary

This thesis deals with mobile sensor data measurements for fall detection in order to provide an economical, mass deployable solution for home based care of elderly people. During the the last century life expectancy has increased dramatically due to advances in health care. In all cultures, it is universally acknowledged that taking care of elderly people is an important social responsibility. However, in countries such as Finland and Japan demographic shifts has reduced the number of potential caregivers. The most economical solution is to provide the elderly with health care at home.

Earlier paradigms in fall detection are based on sensors being attached to the body of a person of interest to track motion. Modern mobile devices are ubiquitous and have multiple onboard sensors like gyroscope, accelerometer, magnetometer and barometric sensor. A mobile phone can be used instead compared to solutions that require sensors being attached to a body. Data collected from multiple onboard sensors can be used to detect falls. This approach is more cost effective and easier to implement because all that is required is downloading an application onto a mobile phone.

## 6.1 Contribution

In this thesis, we develop an android application to read and store data from various sensors onboard an Android mobile phone. Using the application, various measurements are taken for different movements like walking, climbing and falling. The collected data is then used to train algorithms that detect falls. We demonstrate the efficacy of these methods for fall detection. This research is a step towards providing an economical and mass deployable solution for home based care of elderly people.

## 6.2 Future Work

The Android application can be enhanced to provide real time detection of falls. Improvements can be made to the fall detection algorithm by using other machine learning techniques including deep learning algorithms. Remote monitoring of patients can be improved through indoor localization to identify where exactly the person is in real time so that emergency services can respond more effectively.

Indoor tracking of the elderly is a future research work. Sensor Data Fusion might improve the overall performance of a remote health care system. Accessories such as heartbeat monitor, blood pressure monitor and sensors on wrist watches can be added to improve the overall performance. The data collected can also be used for analysis of human behavioural patterns.

# References

[1] H. Liu, H. Darabi, P. Banerjee, and J. Liu, "Survey of wireless indoor positioning techniques and systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews,*, vol. 37, no. 6, pp. 1067–1080, 2007.

[2] S. Abbate, M. Avvenuti, P. Corsini, J. Light, and A. Vecchi, "Monitoring of human movements for fall detection and activities recognition in elderly care using wireless sensor network: A survey," *Wireless Sensor Networks: Application-Centric Design Book*, 2010.

[3] M. Mubashir, L. Shao, and L. Seed, "A survey on fall detection: Principles and approaches," *Neurocomputing*, vol. 100, pp. 144 – 152, 2013.

[4] B. Celler, W. Earnshaw, E. Ilsar, L. Betbeder-Matibet, M. Harris, R. Clark, T. Hesketh, and N. Lovell, "Remote monitoring of health status of the elderly at home. a multidisciplinary project on aging at the university of new south wales," *International Journal of Bio-Medical Computing*, vol. 40, no. 2, pp. 147 – 155, 1995.

[5] Y. Lee, J. Kim, M. Son, and M. Lee, "Implementation of accelerometer sensor module and fall detection monitoring system based on wireless sensor network," in *29th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 2007, pp. 2315–2318.

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

[7] R. Gutiérrez, J. J. García, J. C. García, L. Marnane, D. Gualda, S. Fernández, and E. Garcia, "Activity monitoring and emergency warning with location information of the user," in *7th IEEE International Symposium on Intelligent Signal Processing*, Sept 2011, pp. 1–6.

[8] M. Tolkiehn, L. Atallah, B. Lo, and G. Z. Yang, "Direction sensitive fall detection using a triaxial accelerometer and a barometric pressure sensor," in *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 2011, pp. 369–372.

[9] J. Dai, X. Bai, Z. Yang, Z. Shen, and D. Xuan, "Mobile phone-based pervasive fall detection," *Personal and Ubiquitous Computing*, vol. 14, no. 7, pp. 633–643, 2010.

[10] N. Eagle and A. Pentland, "Reality mining: Sensing complex social systems," *Personal and Ubiquitous Computing*, vol. 10, no. 4, pp. 255–268, 2006.

[11] S. H. Fang, Y. C. Liang, and K. M. Chiu, "Developing a mobile phone-based fall detection system on android platform," in *2012 Computing, Communications and Applications Conference*, Jan 2012, pp. 143–146.

[12] T. Hori, Y. Nishida, H. Aizawa, S. Murakami, and H. Mizoguchi, "Sensor network for supporting elderly care home," in *Proceedings of IEEE Sensors*, 2004, pp. 575–578.

[13] A. Godfrey, A. Bourke, G. Ólaighin, P. van de Ven, and J. Nelson, "Activity classification using a single chest mounted tri-axial accelerometer," *Medical Engineering and Physics*, vol. 33, no. 9, pp. 1127 – 1135, 2011.

[14] M. Kangas, I. Vikman, J. Wiklander, P. Lindgren, L. Nyberg, and T. Jämsä, "Sensitivity and specificity of fall detection in people aged 40 years and over," *Gait and Posture*, vol. 29, no. 4, pp. 571 – 574, 2009.

[15] M. Benocci, C. Tacconi, E. Farella, L. Benini, L. Chiari, and L. Vanzago, "Accelerometer-based fall detection using optimized ZigBee data streaming," *Microelectronics Journal*, vol. 41, no. 11, pp. 703 – 710, 2010.

[16] M. Nyan, F. E. Tay, and M. Z. Mah, "Application of motion analysis system in pre-impact fall detection," *Journal of Biomechanics*, vol. 41, no. 10, pp. 2297 – 2304, 2008.

[17] M. Valero, I. Pau, L. Vadillo, A. Penhalver, E. Gago, M. Martin, M. Gonzalez, and E. Portillo, "An implementation framework for smart home telecare services," in *Future Generation Communication and Networking*, vol. 2, Dec 2007, pp. 60–65.

[18] J. S. Wilson, *Sensor technology handbook*. Elsevier, 2004.

[19] O. J. Woodman, "An introduction to inertial navigation," University of Cambridge, Computer Laboratory, Tech. Rep., 2007.

[20] A beginner's guide to accelerometers. [Online]. Available: https://www.dimensionengineering.com/info/accelerometers

[21] Accelerometer. [Online]. Available: https://en.wikipedia.org/wiki/Accelerometer

[22] Accelerometer. [Online]. Available: http://www.sensorwiki.org/doku.php/sensors/accelerometer

[23] Gyroscope. [Online]. Available: http://sensorwiki.org/doku.php/sensors/gyroscope

[24] Gyroscope. [Online]. Available: https://en.wikipedia.org/wiki/Gyroscope

[25] Magnetometer in smart phones and tablets. [Online]. Available: http://www.rotoview.com/magnetometer.htm

[26] Magnetometer. [Online]. Available: https://en.wikipedia.org/wiki/Magnetometer

[27] Pressure sensor. [Online]. Available: https://en.wikipedia.org/wiki/Pressure_sensor

[28] Low-power barometric pressure sensor for mobile and wearable gadgets and IOT devices. [Online]. Available: https://phys.org/news/2015-02-low-power-barometric-pressure-sensor-mobile.html

[29] Global positioning system. [Online]. Available: https://en.wikipedia.org/wiki/Global_Positioning_System

[30] Google nexus 4. [Online]. Available: http://www.android.gs/device/google-nexus-4/

[31] Nexus 4. [Online]. Available: https://en.wikipedia.org/wiki/Nexus_4

[32] Motorola XOOM MZ604. [Online]. Available: http://www.gsmarena.com/motorola_xoom_mz604-3833.php

[33] Android 0 developer preview. [Online]. Available: https://developer.android.com/

[34] Motion sensors. [Online]. Available: https://developer.android.com/guide/topics/sensors/sensors_motion.html

[35] M. Barkat, *Signal Detection and Estimation.* Artech House, 2005.

[36] S. Boyd and L. Vandenberghe, *Convex Optimization.* Cambridge University Press, 2004.

[37] A. Ben-Tal and A. Nemirovski, *Lectures on modern convex optimization: Analysis, algorithms, and engineering applications.* SIAM, 2001.

[38] Y. Nesterov and A. Nemirovski, *Interior-Point Polynomial Algorithms in Convex Programming.* SIAM, 1994.

[39] S. Kay, *Fundamentals of Statistical Signal Processing: Detection theory.* Prentice-Hall, 1998.

[40] L. Scharf and C. Demeure, *Statistical Signal Processing: Detection, Estimation, and Time Series Analysis.*

[41] H. Van Trees, K. Bell, and Z. Tian, *Detection Estimation and Modulation Theory, Detection, Estimation, and Filtering Theory.*

[42] J. Moragues, L. Vergara, J. Gosálbez, and I. Bosch, "An extended energy detector for non-gaussian and non-independent noise," *Signal Processing*, vol. 89, no. 4, pp. 656 – 661, 2009.

[43] H. Urkowitz, "Energy detection of unknown deterministic signals," *Proceedings of the IEEE*, vol. 55, no. 4, pp. 523–531, April 1967.

[44] ——, "Energy detection of a random process in colored gaussian noise," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-5, no. 2, pp. 156–162, March 1969.

[45] B. Polyak, "Convexity of quadratic transformations and its use in control and optimization," *Journal of Optimization Theory and Applications*, vol. 99, no. 3, pp. 553–583, 1998.

[46] E. Alpaydin, *Introduction to Machine Learning.* MIT Press, 2010.

[47] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics).* Springer-Verlag, 2006.

[48] D. MacKay, *Information Theory, Inference and Learning Algorithms.* Cambridge University Press, 2003.

[49] M. C. Grant and S. P. Boyd, *Recent Advances in Learning and Control*. Springer, 2008, ch. Graph Implementations for Nonsmooth Convex Programs, pp. 95–110.

[50] S. Watanabe, *Pattern recognition: Human and mechanical*. Wiley, 1985.

[51] S. Särkkä, *Bayesian Filtering and Smoothing*. Cambridge University Press, 2013.

[52] L. Vandenberghe and S. Boyd, "Semidefinite programming," *SIAM Review*, vol. 38, no. 1, pp. 49–95, Mar. 1996.

[53] T. Nguyen and S. Sanner, "Algorithms for direct 0–1 loss optimization in binary classification," in *Proceedings of the 30th International Conference on Machine Learning*, 2013, pp. 1085–1093.

[54] L. N. Trefethen and D. Bau, *Numerical Linear Algebra*. SIAM, 1997.