# A CONTEXT AWARE APPROACH FOR ENHANCING GESTURE RECOGNITION ACCURACY ON HANDHELD DEVICES

A THESIS

SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCE

OF BİLKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Hacı Mehmet Yıldırım

August, 2010

# ABSTRACT

## A CONTEXT AWARE APPROACH FOR ENHANCING GESTURE RECOGNITION ACCURACY ON HANDHELD DEVICES

Hacı Mehmet Yıldırım
M.S. in Computer Engineering
Supervisor: Asst. Prof. Dr. Tolga K. Çapın
August, 2010

Input capabilities (e.g. joystick, keypad) of handheld devices allow users to interact with the user interface to access the information and mobile services. However, these input capabilities are very limited because of the mobile convenience. New input devices and interaction techniques are needed for handheld devices. Gestural interaction with accelerometer sensor is one of the newest interaction techniques on mobile computing.

In this thesis, we introduce solutions that can be used for automatically enhancing the gesture recognition accuracy of accelerometer sensor, and as a standardized gesture library for gestural interaction on touch screen and accelerometer sensor.

In this novel solution, we propose a framework that decides on suitable signal processing techniques for acceleration sensor data for a given context of the user. First system recognizes the context of the user using pattern recognition algorithm. Then, system automatically chooses signal filtering techniques for recognized context, and recognizes gestures. Gestures are also standardized for better usage.

In this work, we also present several experiments which show the feasibility and effectiveness of our automated gesture recognition enhancement system.

*Keywords:* Gestural interaction, gesture, computer graphics, accelerometer sensor, signal processing.

# ÖZET

## ÇEVRE FARKINDALIĞI TABANLI YAKLAŞIMLA TAŞINABİLİR BİLGİSAYARLARDA İŞARET TANIMASI HASSASİYETİNİN ARTTIRILMASI

Hacı Mehmet Yıldırım
Bilgisayar Mühendisliği, Yüksek Lisans
Tez Yöneticisi: Asst. Prof. Dr. Tolga K. Çapın
Ağustos, 2010

Mobil cihazların tuş takımı ve kontrol kolu gibi veri girişini sağlayan kabiliyetleri, kullanıcının kullanıcı ara birimi yoluyla bilgiye ve mobil servislere ulaşmasını sağlar. Fakat bu veri girişi kabiliyetleri mobil kullanımdan dolayı sınırlıdır. Mobil cihazlar için yeni veri giriş cihazları ve teknikleri gerekmektedir. İvme ölçer kullanarak işaretlerle etkileşim mobil cihazlarda en yeni etkileşim yöntemlerinden biridir.

Bu tezde önerilen çözüm, otomatik olarak ivme ölçer işaretlerinin tanınmasının iyileştirilmesinde, dokunmatik ekran ve ivme ölçer işaretlerinin standart kütüphanesi oluşturulmasında kullanılabilir.

Sunulan çözümde, ivme ölçer işaretlerini tanımak için uygun olan sinyal işleme yöntemleri otomatik olarak belirlenmektedir. Öncelikle, sistem örüntü tanıma algoritması kullanarak kullanıcının hareketini tanır. Daha sonra, sistem otomatik olarak kullanıcı hareketine uygun veri işleme yöntemini seçer ve işaretleri algılar. Ayrıca işaretler daha iyi kullanım için standart hale getirilmişlerdir.

Bu çalışmada ayrıca, önerilen otomatik işaret tanıma iyileştirme sisteminin etkili ve uygulanabilir olduğunu gösteren birkaç kullanıcı testine de yer verilmektedir.

*Anahtar sözcükler*: İşaretlerle etkileşim, işaretler, bilgisayar grafiği, ivme ölçer, sinyal işleme.

# Acknowledgement

First of all, I would like to express my sincere gratitude to my supervisor Asst. Prof. Dr. Tolga K. Çapın for his endless support, guidance, and encouragement.

I would also like to thank to jury members, Prof. Dr. Bülent Özgüç and Asst. Prof. Dr. Tolga Can for spending time for read and evaluate this thesis.

Finally, I would like to thank to TÜBİTAK-BİDEP and EU 7th Framework "All 3D Imaging Phone" Project for financial support during my M.S. study.

Thanks to Abdullah Bülbül and Zeynep Çipiloğlu for their support during application phase of this study.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Modern handheld devices enable users to access a wide variety of information and communication services. Handheld devices are used by people of all ages, occupations and abilities, who care about acquiring access to the information and services, anytime anywhere. Input capabilities (e.g. joystick, touch screen, keypad, accelerometer sensor and camera) of these devices allow users to interact to access the information and services. These input capabilities are very limited because of the mobile convenience.

Handheld devices have been pervasive in our daily lives because devices are more capable of understanding the needs of the user and respond to them in a more sensible manner. Their computational power, long battery life, video processing capability, advanced display, GPS hardware, touch screen sensor, acceleration sensor and camera have given the user a great opportunity to use them across an increasing range of places and contexts. As mobile phones have become more ubiquitous, they have also gained features such as internet browsing, route navigation, music/video playing and office application capabilities. These various kinds of capabilities, available in any contexts, allow user to enjoy the service benefits anywhere they want. People interact with their mobile devices any time, even while waiting for a bus on the platform, talking with friends on a cafe, crossing a road on foot, cycling and travelling in car.

For mobile convenience, handheld devices do not have keyboard and touchpad developed for notebooks, instead they have keypad, stylus pen or touch screen. Modern handheld devices also provide advanced input capabilities. Multiple touch screen and camera are the most popular and common advanced input technologies. However, input capabilities of these devices are not limited to these input devices. Accelerometer sensor can also be used as an input device to interact with the handheld device, because of its ubiquitous support on handheld devices.

The prevalence of the accelerometer is attributed to its numerous advantages over other sensors. It is lightweight, small, and inexpensive. It consumes small amounts of energy. It is self-operable, meaning that it does not require infrastructure for operation. This information can be used for contextual information regarding user's movement (e.g. walking, cycling, running) and the size and moving directions detected from acceleration sensor can be used to classify user gestures.

**Motivation**

The PC form factor cannot be used in smaller sizes. Hardware, input device, user interface and interaction techniques need to be designed for mobile usage. User cannot pay attention to the provided information by handheld device and interact with the handheld device in a dynamic and complicated mobile environment. Therefore, user interface, input devices and interaction techniques of the device should be changed to have an effective usage while user is moving.

Desktop and notebook PCs' input devices (e.g. mouse, keyboard, keypad and touchpad) have standardized functions on the same input type. User does not need to learn new interaction techniques when she changes the keyboard. Right arrow key of all the keyboards have the same function on the user interface or all the right button of all the mice have the same function. On the other hand currently interaction techniques for handheld devices do not allow users to interact with the device intuitively. Current interaction models for handheld devices depend on buttons. This provides the user the functionality to use simple menus but this is not very easy to use and it is not intuitive. To provide a more intuitive and easy to use interaction system for handheld devices touch screen and acceleration sensor are used. Touch screen and accelerometer are used as gestural input devices. These gestures are not standardized. User needs to learn every gesture when changing the handheld device. Every handheld device has own gesture library and gesture meanings; so, interaction with accelerometer sensor and touch screen is not easy and learning the gesture library takes long time.

Unlike ordinary desktop and notebook PCs, handheld devices are not used on the pre-assumed situations. When a desktop or a notebook PC is developed, developers assume that these devices are used in a fixed state on a table, or on a surface that is not moving and by people who is paying full attention. In spite of the fact that handheld devices interaction techniques and user interfaces are inherited from an ordinary desktop PC, handheld devices are not used in fixed surface or in a stable state as in pre-assumed for desktop PC. For example a handheld device user is in a situation that he is walking, maintaining awareness of the surroundings, avoiding obstacles and using a device that is itself in motion. To make the handheld device more user friendly, the gap between an assumption for ordinary PCs (e.g. input by typing on a large keyboard which is fixed on the desk) and an actual use case on the handheld device (e.g. input by typing on a tiny keypad on the train) has to be closed.

**Overview of the System**

Handheld devices' compact form has the advantages of mobility, but also imposes limitations on the interaction methods that can be used. With handheld device's compact form, keyboards are so small, difficult and even impossible to use. On the other hand, because of their compact form displays are so small, which makes it harder to interact with the touch screen with fingers. Therefore, button based and touch screen interaction is not sufficient and intuitive. In this study acceleration based interaction method is examined and tested. This interaction method is very intuitive because of the metaphor of realistically responding physical objects on the user interface.

Compared to desktop computers the use of the handheld devices is more intimate because handheld devices have become a part of daily life. People use handheld devices in many different, dynamic and complicated mobile environments so designers do not have the luxury of forcing the user to assume the position to work with these devices, as is the case with desktop computers. On the other hand using acceleration based interaction techniques will fail on the dynamic environment because device could not recognize the command of the user among the noisy environment. When the user needs to enter an acceleration based gesture command to the device when running, device needs to have the ability to identify the rhythmic movement due to running and the intended user command. In this study we have proposed a context aware signal filtering technique to use accelerometer based interaction technique in a dynamic environment.

**Challenges**

Developing a standardized gesture library for handheld devices using touch screen and accelerometer sensor is a challenging job due to several reasons. First of all, since the possible user range is wide, gesture library should be simple and intuitive enough for the users with basic knowledge; however it should satisfy the needs of a complex user at the same time. Secondly, all the gestures in gesture library should be mapped into the real physical world. For example move gestures in the library should be mapped to real world moving movements: gravity forces objects to slide towards to the centre of the earth, this is a natural phenomenon. When a table has an angle to the right, objects slide to the right due to the gravity forces. In a handheld device, screen has an angle to the right to make the objects on the screen slide to the right.

Developing a gesture recognizer with accelerometer sensor for handheld devices is also a challenging job due to the noisy data of the acceleration sensor. First of all, the user of the handheld device is not a stable state. Handheld device is used when walking, running, cycling, and etc. Due to the movement of the device, signals of the accelerometer sensor are noisy. Secondly, accelerometer sensors are not accurate; they produce noisy data, even if they are in a stable state. Therefore, recognizing gestures by using these noisy data is challenging.

**Summary of the Contributions** The contributions of this thesis can be summarized as follows:

- A survey on context aware systems, signal processing of the acceleration sensor, and gestural mode interaction techniques with handheld devices,
- A standardized gesture library for accelerometer sensor and touch screen to combine all the known gestures and proposed functions of these gestures,
- An activity recognition system for handheld devices,
- A signal processing algorithm to automatically determine the proper signal processing method according to the context of the user,
- A tool, for application developers, to recognize gestures by using signal processing according to user's context,
- An experimental study to evaluate the effectiveness of the proposed algorithms.

**Outline of the Thesis**

- Chapter 2 presents a comprehensive investigation of the previous work on the topic of User's Context Aware Systems, signal processing of acceleration data, and gestural mode and interaction techniques with handheld devices.
- In Chapter 3, architecture of our proposed system and subsystems for context awareness and signal processing are explained in detail.
- In Chapter 4, our proposed subsystems for gesture design and gesture recognition are explained in detail.
- Chapter 5 contains the result of an experimental evaluation of the proposed system.
- Chapter 6 concludes the thesis with a summary of the current system and future directions for the improvements on this system.

# Chapter 2

# Background

This chapter is mainly divided into three sections. In the first section, context awareness systems are investigated, while the second section presents the signal processing of the acceleration sensor signals. In the third section, gestural mode of interaction with handheld devices is investigated.

## 2.1 Context Awareness

### 2.1.1 Definition of the word "Context"

The word *context* is defined as "the situation within which something exists or happens, and that can help explain it" [23]. However, this definition cannot help to understand the concept in computer engineering. The term context is used in different areas of computer science, such as context sensitive user interface, context search, contextual perception, and so on. In this section, only the context used by applications in mobile computing is examined. Researchers have attempted to define the term in computer engineer point of view. In literature the term context aware appeared in Schilit and Theimer [65] for the first time. The authors described the term as location, identities of nearby people, objects and changes to those objects. In another study, Schmidt et al. define context as "the knowledge about user's and IT device's state, including surroundings, situation, and to a less extent, location" [66]. Dey defines context as [2]:

> "Any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and application itself."

Researchers divide the term "context" into categories for better understanding. Schilit divides context into three categories [64]:

- Computing context, such as network bandwidth, network connectivity, network cost and nearby resources such as printers.
- User context, such as user's location, social situation, mood, people nearby.
- Physical context, such as light level, noise level, temperature.

Chen adds the fourth category in context [15]:

- Time context, such as time of the day, week, month, season and year.

### 2.1.2 Location Aware Systems

We focus on user context and in this subsection context aware systems are investigated. Early work in context awareness focused on location aware systems. The history of context aware systems started when Want et al. introduced the Active Badge Location System [75], which is considered to be one of the first context aware applications. Want et al. have designed a system presenting the location of the users to the receptionist. The receptionist forwards the phone calls to the user's nearest telephone, by tracking the user's location. This study shows that context aware systems are very useful and could be used for the practical usage.

After Active Badge system with the help of digital telephony, systems that automatically forward the phone calls have been designed. After using the system, designers have observed that people want to have more control on forwarded calls. For example, people prefer not to take calls when they are having a meeting; thus more complex and intelligent context aware systems are needed.

Researchers have developed a similar system to the Active Badge System that forwards messages instead of phone calls. Munoz et al. have presented a context aware system that support information management of a hospital [52]. All personal of the hospital is equipped with mobile devices with the context aware system to write messages. These messages are sent when specified circumstances are occurred. For example a user can identify a message that should be delivered to the first doctor who enters room number 115 between 8:00 am and 9:00 am. The contextual elements of the system are location, time, and role of the user.

After the success of Active Badge System, Bennet et al. have designed a new system called Teleporting [10], which dynamically maps the user interface onto the resources of the surrounding computer systems. This system is based on the Active Badge System and can track the user location while they move around. A new version of the system has been developed by Harter et al. that uses a new location tracking system called the Bat [33]. The Bat uses both ultrasonic and radio signals to precisely locate the user. Pham et al. have also developed a similar context aware system that is aimed to augment mobile devices with the computational power of the surrounding resources. This system is called Composite Device Computing Environment [58], uses location aware application to locate the mobile device.

After the Bat system, researchers have tried to find new precise and effective location detection systems. One of these researhers is Want. Want et al. have designed a system that detects the location of the user, by the wireless network system and faces of the users are displayed at the location of the people on the map [76], [77]. This map is updated in every few seconds to locate people easily. Another similar approach to the location aware systems is a shopping asistant. Asthane et al. have introduced a system that uses a wireless location awareness system to assist to the shoppers [6]. The wireless location aware system detects the location of the shopper within the store and guides the shopper through the store. System helps user to locate items, provide details about the items, and points the items on sale. Another asistant has been developed by Dey et al [22]. This asistant is not for shoppers, instead of them the system serves for conference presenters. The asistant uses the user's current location, current time and schedule of presentations to examine the conference schedule, user's current location and user's research interest to suggest the presentations to attend. When the user enters the presentation room, the asistant automatically displays information about the presenter and the presentation. Integrated audio and video recorders record the presentation for later retrieval.

Tourist guides are one of the applications of location aware systems. Long et al. have designed and developed a tourist guide called the Cyberguide [46]. The system is an electronic tourist guide that is equipped with a context aware system. The guide uses GPS and infrared sensors for indoor and outdoor location awareness, and provides information services for users about the current location and suggests places. The guide also keeps a trip diary using the location and time information of the user. Abowd et al. have also developed a tourist guide [1] that provides information services about the current location. this guide uses location and time awareness. Another tourist guide is the GUIDE system [19] that developed by Davies et al. at the University of Lancester. GUIDE is an electronic tourist guide for visitors to the city of Lancester , England. The system is context sensitive. For the museums, small scaled indoor versions of the tourist guides are developed [9], [69], [55]. In these small scaled versions, location and orientation information is used to understand the context of the users.

One of the popular application of location aware systems is logging the users' activities. Researchers at University of Kent at Canterbury [56], [57] have developed a system that automatically record information about workers. System uses user's location, time information and displays workers location on a map. Another logging application is ComMotion [50]. Marmasse et al. have developed a system that is called the ComMotion [50]. The system uses both user's location and current time. When a user arrives at a pre-entered destination a reminder message is created. The ComMotion is also used for logging the location of the users. The message delivered via voice synthesis without requiring the user to hold the device and read the message on screen.

### 2.1.3 Activity Aware Systems

After location aware systems, the need to know about the activity of the user has appeared and researchers focused on activity aware systems. One of the first studies on activity aware systems is based on an office assistant. Yan et al. [81] have developed a system that is an assistant for the offices. System uses office owner's current activity and schedule. System interacts with the visitors and manages the office owner's schedule. System uses pressure sensitivity sensors to detect visitors, which are activated when a visitor is approaching. It adapts the system according to contextual information such as identity of the visitor, office owner's schedule, and office owner's status.

Activity aware systems are very useful and accurate when they are mobile. One of the first mobile activity aware systems have designed by Randell. The system uses a single biaxial accelerometer to identify the activity of user [60]. System is worn in a trouser pocket. Randell have tried to minimize number of accelerometer devices needed so a single biaxial accelerometer sensor is used. To identify the activity of the user neural network analysis is used. The recognizer calculates the RMS and integrated values over the last two seconds for both axes to recognize context of the user. The system identifies six activities: walking, running, sitting, walking upstairs, downstairs, and standing. Mantyjarvi et al. [48] have also applied neural networks to human motion recognition. Their feature vector is created with principle component analysis and independent component analysis from a pair of triaxial acceleration sensor attached to the left and right hips. Gyorbiro et al. have developed another mobile system that recognizes and records user's activities using a mobile device [31]. The context awareness is used for life logging. The system uses wireless device called MotionBand [38]. Main applications are life logging and calculating energy consumption for sportsmen.

Gyorbiro is not the only one who studies about activity aware systems in mobile devices. Siewiorek has also designed and implemented mobile context aware systems. The system is called SenSay (sensing and saying). Sensay is a context aware mobile phone that adapts to dynamically changing environment and physiological states [67]. Accelerometer sensor, light sensor, and microphone are located various parts of the user to identify user's context. Sensay adjusts ringing volume level, or vibration according to user's context. The system also provides the caller with feedback of the current status of the user.

One of the important areas of motion recognition is healthcare. Jin et al. have developed a health care system that uses an accelerometer sensor to identify the context of the user [36]. Context of the user is used to recognize emergency situations. System uses an arm band that is embedded with an accelerometer. Arm band collects accelerometer sensor data containing the longitudinal acceleration average, the transverse acceleration average, the longitudinal acceleration mean of absolute difference. Chen et al. [16] also have implemented a mobile device based system for multiple vital signs monitoring. The system can detect if a monitored patient falls using a wireless acceleration sensor. The system can alert care provider. Another healthcare study has been done by Mantyjarvi [51]. In this study, accelerometers are used to detect symptoms of Parkinson's disease. Researchers have realized that activity aware systems are very useful for patients, and patients need to be observed by nurses. Lustrek et al. have developed a fall detection system that do the observation of patients automatically, called the Confidence. The system uses user's activity. The Confidence monitors the user's activity and raises an alarm if a fall is detected, system warns of changes in behavior that may indicate a health problem [47].

Lately, wireless accelerometer sensors have become available, enabling measurements in more comfortable settings. Bao et al. have presented a study about activity recognition from user annotated acceleration data. In this study context awareness algorithms are developed and evaluated to detect physical activities. System uses five small biaxial accelerometer sensors worn simultaneously on different parts of the body [7]. Decision table, instance based learning, C4.5 decision tree, and naive bayes classifier from Weka Machine Learning Algorithms Toolkit are used to classify acceleration sensor data. Twenty everyday activities are tried to be detected. Utilizing the same toolkit but only a single triaxial accelerometer worn in the pelvic region, Ravi et al. [61] have studied the performance of base level classifier algorithms.

### 2.1.4 Hybrid Aware Systems

Information about location or activity of the user is important. However, knowing only one of them is not enough for some applications. Application need to know about activity of the user, location of the user, nearby people, and nearby objects at the same time. One of the first studies about hybrid aware systems is about publishing user information. Voelker et al. have developed a system called Mobisaic [73]. Mobisaic is an extended standard web browser that supports active documents. Active documents are the web pages with the embedded environment variables. Environment variables are location, activity, nearby people, and nearby objects. Whenever the environment variables change, the system updates the webpage of the user.

Brown et al. [13] have also designed and developed a context aware system that can detect user's location, nearby people and nearby objects. The system can send messages to people who don't have a paging device. The system can detect the closest person and route the message to the closest person with a paging device. Brown et al. [12] also developed a system that can locate a book and broadcast a message to the nearer people to the book, whoever encounters this book will pick it up for the requester. This context aware system uses user's location, nearby people and objects.

Another hybrid context aware system has been designed and developed Schmidt et al. The system is called Technology for Enabling Awareness (TEA) [66] system. Application detects user's activity, light level, pressure and proximity of other people. In a mobile device, system adapts the font size to the user activity. A larger font size is used when user is moving and a smaller font size is used when user is stationary. TEA also changes the current profile according to the user's context. Mobile device adjusts the ring volume level, vibrate, silent depending on whether the mobile device is in hand, on a table, or in a suitcase. Lee and Mase [43] have also developed an activity and location aware system using combination of biaxial accelerometer, compass and gyroscope. The classification technique is based on a fuzzy logic reasoning method.

## 2.2 Signal processing of acceleration sensor data

Modern handheld devices enable users to access a wide variety of information and communication services. Handheld devices are used by people of all ages, occupations and abilities, who care about acquiring access to the information and services anytime anywhere. Input capabilities (e.g. joystick, touch screen, keypad, accelerometer sensor and camera) of these devices allow users to interact with the user interface to access to the information and services. However, these input capabilities are very limited and some of them are not very accurate. This section examines the studies on how to treat the signal of input devices to recognize gestures in an accurate way.

One of the important studies in signal processing of acceleration data has been presented by Jang et al. [35], focusing on how to recognize gestures precisely by signal processing. They have proposed a system that is used in mobile devices and interaction of the system is made by acceleration gestures. The purpose of using gesture recognition in mobile devices is to provide easy and convenient interfaces. The application uses low pass filtering, thresholding, high pass filtering, boundary filtering, debouncing filtering to filter acceleration data to precisely recognize gestures.

Wieringen et al. have developed a fall detection and medical patient monitoring device [72]. The system monitors the user's activity and raises an alarm or calls the operator for help if a fall is detected. The goal of this research is to develop a wearable sensor device that uses an accelerometer for monitoring the movements of the user to detect falls. The data coming from accelerometer sensor is processed in real time. In the system, low pass filtering and thresholding is used to filter the accelerometer data. On another study, Marinkovic et al. [49] have presented another fall detection and medical patient monitoring system. The fall detection is done by monitoring subsystem in the form of body area sensor network. System uses thresholding and high pass filtering to filter the acceleration sensor data.

Liu et al. have presented a new technique to recognize acceleration gestures [45]. This new technique is called uWave. The uWave is an efficient gesture recognition algorithm that uses a triaxial acceleration sensor. Unlike statistical methods, uWave requires a single training sample for each gesture pattern. The system allows the users to employ personalized gestures and physical manipulations. Thresholding is used for signal processing of the acceleration sensor data. Liu et al. have also proposed a system that allows users to authentication on uWave with a single triaxial acceleration sensor [44]. In the study, authors report a series of user studies that evaluate the feausibility and usability of user authentication with a single triaxial acceleration sensor. The authentication system also uses thresholding for signal filtering.

Agrawal et al. have presented a system called PhonePoint Pen [4] that uses the built in accelerometer in mobile device to recognize human hand writing. A user can write short messages and draw simple diagrams in air by holding the mobile device like a pen. The system uses a single triaxial acceleration sensor. To recognize user's handwriting system needs to filter the background noise. To filtering noisy data the PhonePoint Pen system smooth the accelerometer readings by applying a moving average over the last n readings (n is seven in the study).

Tanviruzzaman et al. have proposed an adaptive solution to secure the authentication process of mobile devices. The system is called ePet [70]. Gait and location information are used for authentication process. The mobile device learns the attributes of the owner like his voice, hand geometry, user's daily walking patterns, user's specific gestures and remembers those to continually check against the stealing. The ePet uses single triaxial accelerometer and GPS module to record gait and location information. Authors used high-pass filtering on the accelerometer's raw data to get rid of the gravitational effects and find out the instantaneous movement of the device.

Zang et al. have presented a novel gesture recognition method [84]. It is designed and implemented for keyless handheld devices. The system contains a trixial accelerometer sensor and a single chip microcontroller which is used for gesture recognition. User moves the mobile device in air like a pen and initializes gesture or writes letters. System uses a basic signal filtering technique. Accelerometer raw data is filtered by a threshold. The system also keeps the first peak that appeared in the sequence of the raw acceleration data and filters the second or third peaks which will output an unwanted result.

## 2.3 Gestural mode and interaction techniques in mobile devices

### 2.3.1 Experimental User Interfaces

Handheld devices of these days, are more capable of understanding the needs of the user and respond them in a more sensible manner. Their computational power and hardware properties like acceleration sensors, cameras, graphic hardware, etc. give the designer and programmer a great opportunity to design much more user friendly user interfaces. The proposed solution to the problem is developing a dynamic user interface with gestural mode of interaction. Related studies about dynamic user interfaces and gestural mode of interaction are discussed below.

In a study by Kane [40], contextual factors are taken into account and a walking user interface (WUI) is designed. The author claims that contextual factors such as walking reduce the user performance. According to this assumption, he changed the button size and layout of the UI, and tested on different users for dynamic button sizes and different layouts. When in walking mode, buttons are extended in size. Results show that using dynamic UI increases the usability.

In another study by Kane [39], a conceptual study is done. In this study, he uses device sensors to get environmental factors. For different environmental factors, different UI components are launched. Environmental factors can be crowded places, bumpy bus, noisy places or device in pocket. Different solutions for different environments are proposed such as increasing font size, or voice activation enabling.

Yamabe et. al. [79] studied on a similar topic. In his study, he detects the movement by acceleration sensor and uses it in two different applications. The first application uses the font size. If the user is stationary then font size is small and user can see more text on the screen. If the user starts to walk then font size increases, therefore, readability increases. A similar method is used for an image viewer. If the user is stationary, the image is small and if the user is moving then the image becomes bigger. This study covers only font size, and a complete UI has not been studied.

Yamabe et. al. [80] has reported a second study that needs less attention on car map navigation systems. This study is a good example of experimental interfaces. UI displayed for the user what to do next instead of a complex map and navigation information.

### 2.3.2 Experimental Interaction Techniques

As handheld devices become more popular, studying direct manipulation interfaces for different applications becomes important. Although applications with a direct manipulation interface are highly used by many people on PCs, still some problems exist due to the limitations of mobile environments. One of the proposed solutions is developing a gesture based direct manipulation interface. This interface serves the purpose of navigating over user interface elements and documents more efficiently.

One of the first researches on interaction techniques is the Chameleon System [27], [28]. This system uses accelerometer sensors to understand the physical orientation and movement of a small palmtop. The system consists of a small palmtop and a workstation. The palmtop device has accelerometers so the interaction is done via the palmtop and calculations are carried on the workstation side. Design of the system allows detection of x, y, z coordinates and pitch, yaw, roll rotations of the palmtop.

After Chameleon system, more advanced interaction devices have been announced. One such system is the peephole metaphor system by Yee [82]. This system uses a handheld device with a small display. With this small display, users can see large documents, maps and images by moving the device. Device is seen like a window to the document and the user moves this window to see the unseen part of the document. This device has a display and a mouse-like control and input device to understand the two-axis movement of the display. This system does not use accelerometer sensors, but it has a new interaction technique to see larger documents than the screen itself.

Another interaction technique was proposed by Eslambolchilar [25]. The technique is called Speed Dependent Automatic Zooming (SDAZ). This system has a handheld device to display the document and an accelerometer sensor to understand the orientation of the system. The system uses a dynamic approach to manipulate the documents. The level of detail that is seen by the users is not the same when the document is moving and standing. In other words, the system decreases the level of detail of the document when it is scrolled fast. Decreasing the level of detail is done by zooming out and when user stops scrolling, the zoom level is increased to allow the user to view the document.

Another study carried by Decle and Hachet [20] showed that using touch screen as a trackball can be efficient for users to control 3D data in screen. They used touch screen to get the thumb movement performed on screen and use it as a virtual trackball. As an example, user can directly manipulate a 3D sculpture around its axis.

Another study, using tilt control to navigate over documents has been done by Hinckley [34]. Hinckley made an experiment to examine various interaction techniques. Users of the tilt control system learned the device, and mastered the control of the device in a short time but they had complaint about missing the target and the low accuracy of the system. The tilt control users have found the system difficult to use for longer periods.

### 2.3.3 Gesture Based Interaction

There have been various proposed solutions for gesture based interaction. These studies generally are carried for mouse, accelerometer sensor or touch screen devices. The following studies are capable of handling the needs of handheld devices. One of the first studies on using accelerometer as an input device is tilting operations by Rekimoto [62]. He uses tilting and button pressing for menu selection and map browsing. Instead of using classical input method such as keyboard and mouse, tilting is used as an input method. Cylindrical menu and pie menu have been designed and implemented to test the tilting operations. Also, he has tested the tilting operations on a map browser to see how it is on navitagating on a large 2D space. Tilting gesture is designed to use the mobile device only with one hand so only one hand is required to hold and control the device. In this study acceleration sensor is used like a joystick. Acceleration sensor is used as an external input device not an integrated one. Only four basic joystick movement gestures are implemented. The system is also used for displaying 3d objects in a 2d screen. This system understands the rotation of the device and user can examine the various sides of the virtual 3d object on the handheld display.

Joselli et al. have proposed a framework for touch and accelerometer gesture recognition called gRmobile [37]. The gRmobile uses hidden Markov model for recognition of gestures. This study tries to fulfill a gap on user interaction by providing a framework for gesture recognition through touch input or motion input. The gRmobile can be used for games and programs. In order to test the accuracy of the recognition, a data set of gestures has been created and tested by four different users.

Wobbrock et al. have presented a simple gesture recognizer to use in input devices such as touch screen or accelerometer [78]. The system is a geometric template matcher that means given gesture is compared with predefined gestures. The main idea of the study is developing a simple recognizer and in this study very simple and cheap recognizer is developed. The system is all hundred lines of code. After developing the system a user study has been performed with ten subjects.

On gesture interaction there are several studies targeting stylus usage or mouse usage. Bhandari [11] has proposed a research for gesture usage on mobile devices. This work is more an investigation and experiment whether the gestures are usable and efficient for mobile systems. The study used an approach inspired by participatory design, which allows end users to choose the correct gestures for different tasks. In the study a low fidelity but high resolution prototype of a mobile device is creates. This mobile device with a big screen and no key is developed and used as an experimental device. Bhandari focuses on studying the effects of gestures on camera operation and picture management operations.Results were promising, since gestures were found very usable by participants.

A number of research efforts have explored to use of accelerometer sensors to provide additional input degree of freedom for navigation tasks on mobile devices. Small and Harrison [68], [32] have proposed systems that the user can interact with the display of the system to manipulate the document. Input of the systems is the orientation and the position of the display.

Harrison et al. [32] have also used accelerometer sensor as an input device to control the user interface of the mobile device. This study is a conceptual one to understand the basics of tilt control. In this study, acceleration sensor and device is examined together so user tilts the device not an external acceleration sensor. To test the tilt control a book viewer and a sequential list are implemented. Only four basic joystick movement gestures are implemented. After the user studies, paper claims that tilt control is a natural way to interact with the computers and tilt control provide a real-world experience. Similar to Harrison's study, Small and Ishii proposes a spatially aware portable display which use movements and rotations in a physical space to control navigation in the digital space within [68]. In the study a virtual window is designed and implemented to navigate on a virtual space. Accelerometer sensor is used to understand the movement of the device. Only rotation movements are used as gestures.

An advanced study about the built-in sensor devices has been made by Bartlett [8]. Bartlett proposed a handheld electronic photo album that uses panning and tilting gestures to interact with the album. A mobile device is designed, implemented and constructed on the study. This device has an integrated acceleration sensor to understand the rotational movement of the device and shake gesture. Users can browse the photos by changing the rotation of the device. User can navigate through the menus and select the menu item by gestures. Bartlett made a study with the implementation of the system. Some participants of the study found the device very natural, whereas other participants found the sensor based approach more confusing.

Due to the limitations in the user interface, designing a single buttoned game is extremely difficult. One of the new interaction techniques is investigated by Chemini and Coulton. Chemini et al. have presented a new interaction mechanism that uses accelerometer sensor [14]. The system uses gestures for interaction. Authors present the design and user trials for a novel motion controlled 3d multiplayer space game. The results show that the experience of using an accelerometer as an input device is seen fun and intuitive for expert and beginner users.

Another accelerometer based interaction technique has designed by O'Neill. O'Neill et al. have presented a patient information system based on gesture interaction [54]. They developed a gestural input system that provides a common interaction technique across mobile and wearable computing devices. Gestural input system is combined with speech output. The system is tested whether or not the absence of a visual display impairs usability in kind of multimodel application and the system is tested whether or not usability of the gestural mode of interaction is enough to interact with the devices. As a result of the study, gestural mode of interaction is found very intuitive and much more easy to learn and do. Lantz et al. have also presented a system that supports gestural interaction with mobile devices [42]. A pocket pc with a triaxial accelerometer sensor is used as the experimental platform. Dynamic movement primitives are used to learn the limit cycle behavior associated with the rhythmic gesture.

Applications using accelerometer have been studied by researchers. One of the examples is a photo album browser. Cho et al. have presented a photo album browser on mobile devices to browse, search and view photos efficiently by gestural input [17], [18]. The system enables users to browse and search photos more fluently and efficiently by gestural interaction. The system uses continuous input from a triaxial accelerometer sensor to create tilting gestures and a multimodal (visual, audio and vobrotactile) display. Cho et al. compare this tilt based interaction method with a button based interaction method by a quantitative usability criteria and subjective experience. The proposed gestural input improves the usability. Users understand the interaction with tilt based input easily because of the metaphor of realistically responding physical objects. Another application using accelerometer is a musical instrument. Essl et al. proposed an accelerometer sensor based integrated mobile phone instrument [26]. The system is called ShaMus. ShaMus is a sensor based approach to turning mobile device into a musical instrument. ShaMus has two sensors, accelerometer and magnetometer. Accelerometer sensor is used a gesture recognizer. The gestures that are entered to the device are interpreted like musical notes. Three kind of gestures (striking, shaking, and sweeping) are designed and implemented.

Not all the applications use accelerometer. One example is TinyMotion. Wang et al. have presented TinyMotion [74], which is an application that recognizes hand movement of the user. TinyMotion is a software approach for detecting a mobile phone user's hand movements. Recognition is done in real time by analyzing image sequences captured by the built in camera. TinyMotion is used to capture entered text on air. System uses handwriting recognition to recognize text on air, and TinyMotion is used as a joystick for gaming purposes. The system is also used as gesture recognition system for controlling the mobile device. Wang et al. design and implement a contact viewer application, a map browser application, a tetris game, a snake game, and a handwriting recognition application. A user study with 17 participants is done after implementing the applications.

# Chapter 3

# System for Enhancing Gesture Recognition Accuracy

## 3.1 Architecture

In this work, we propose a gesture recognition system that understands the context of the handheld device's user and adjusts different signal filtering technique to acceleration sensor data according to the context of the user. In this system, a standard gesture library is also designed to have a user friendly device. The system automatically understands the activity of the user by pattern recognition algorithm. The system we propose presents an algorithm for automatically selecting the proper signal filtering technique for the current context of the user and the filtering algorithm that provide better and more accurate noise filtering.

In the system, while automatically selecting the proper signal filtering technique for the current context of the user, we consider the acceleration on x, y and z coordinate of the handheld device. The system takes the above items as an input and calculates the current activity of the user. According to the current activity of the user, proper signal filtering techniques are determined. Hence, our system can be considered as a context aware system. Context aware systems are described in Chapter 2.

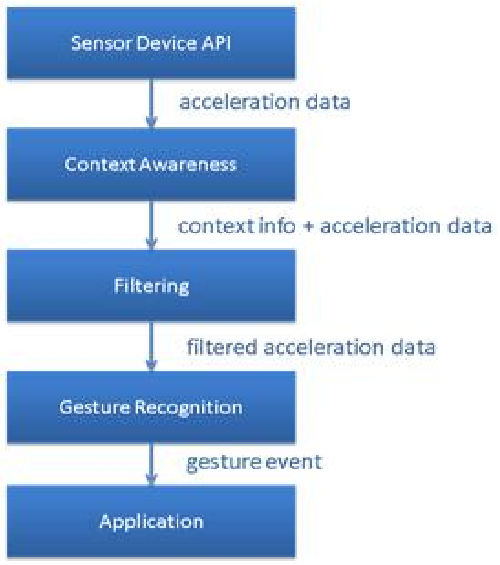


Figure 3.1: Overall System Architecture.

The general architecture of the automatical signal filtering enhancement process can be seen in Figure 3.1. Our approach first determines the current activity of the user on acceleration data with the help of pattern recognition algorithms. The next stage is to decide on proper signal filtering technique. After selecting the proper signal filtering technique, we apply these techniques to acceleration data on x, y and z coordinates. In the next stage, by using filtered acceleration data, gestures are recognized. Recognized gestures are turned to gesture events and sent to the user interface of application.

Our approach first determines the current activity of the user on acceleration data with the help of the pattern recognition algorithm. Context information provided from context awareness part of the system is used to decide on proper signal filtering technique to enhance gesture recognition accuracy. Using context information on gesture recognition part of the system to change the state machines to enhance gesture recognition accuracy is an alternative way. The alternative way has three basic problems. The first problem is about the noisy data of the accelerometer sensor. Accelerometers are not accurate devices; they produce noisy data, even if they are in a stable state. Due to the movement of the user, signals of the accelerometer are also noisy. To enhance the gesture recognition rate noisy data need to be filtered. The second problem with this method is feasibility. Gestures are recognized in a standard way, independent from activity of the user, because gestures are always the same. The user performs the gesture always in same movements. While walking, running or standing, the user does not need to change the way of performing the gesture. Therefore, changing the state machines of gesture recognition part does not enhance the recognition rate, and lower the usability of the device. The last problem with the alternative way is about the performance. Using context information in higher level of the system reduces the performance of the system. The performance in filtering is 5 ms, despite all the filters are active. The performance in gesture recognition is lower than filtering (20 ms) despite there is no complicated state machine in this stage. If context information is used in gesture recognition state, complicated state machines are needed. Changing simple state machines to the complicated ones dramatically reduces the performance.

## 3.2 Context Awareness

Our approach to the context awareness problem is modeling of the contexts with statistics from training data. Samples of sensor data taken from each context are examined to estimate probability densities corresponding to contexts. Once the densities are calculated, the probability of being in a certain context can be computed. To classify the acceleration data into distinct classes we employ a Bayesian classifier. In this section we introduce the activity labels, feature selection, and we examine the classification algorithm we use.

The architecture of the context awareness system can be seen in Figure 3.2. Our context awareness system first performs feature extraction on acceleration data. The next stage is to classify the extracted features. After classification process, the system decides the context information. Input of the context aware system is acceleration data and the output is context information.

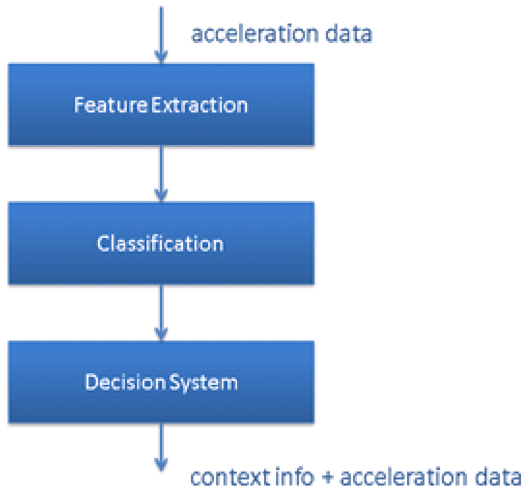


Figure 3.2: Architecture of the Context Awareness Step.

### 3.2.1 Activity Labels

Seven activities are studied. These are standing still, walking, sitting, lying, running, stairing up and stairing down activities. These activities are selected to include a range of common everyday activities. Acceleration values over time for listed activities can be found in Figure 3.3, 3.4 3.5, 3.6, 3.7, 3.8, 3.9.

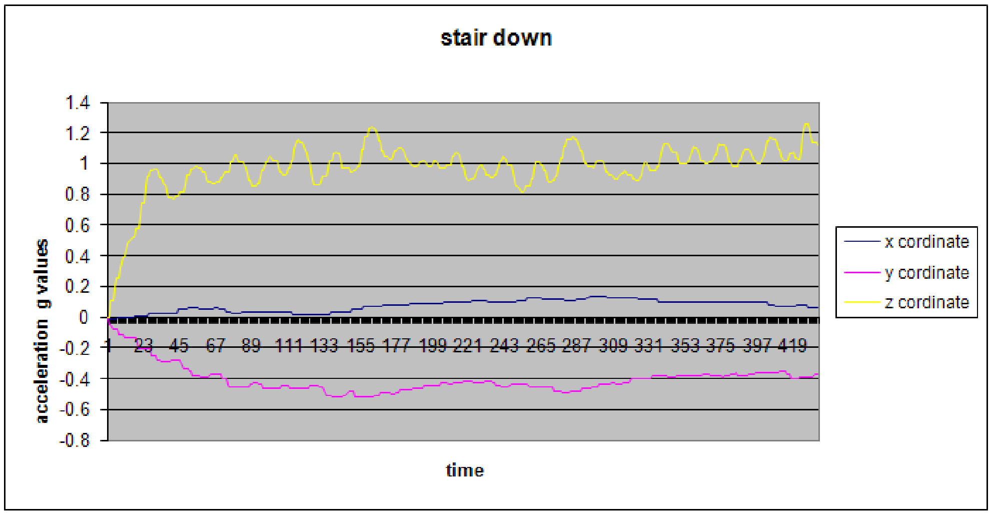


Figure 3.3: Acceleration over time in stair down activity.

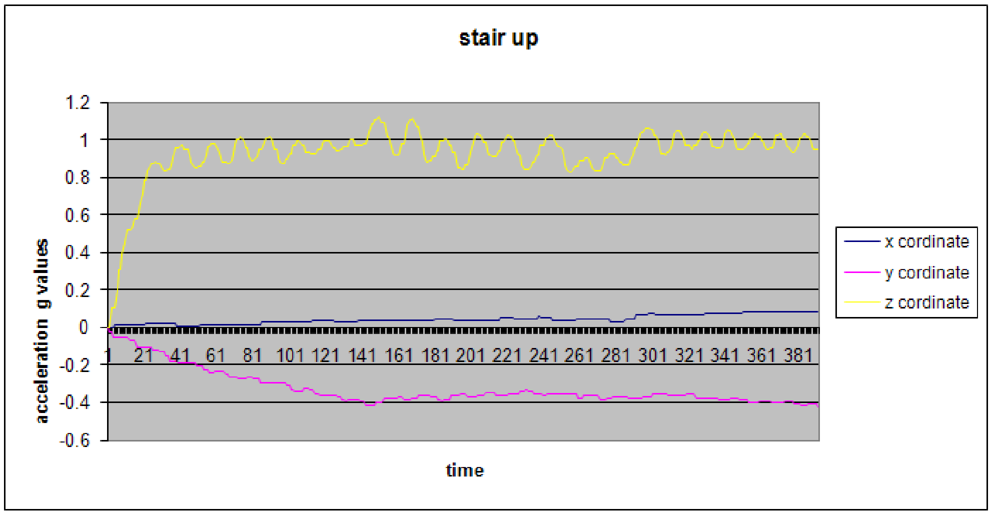


Figure 3.4: Acceleration over time in stair up activity.

### 3.2.2 Features

The Bayesian classification algorithm does not work well when only acceleration raw data samples are used [30]. Algorithm performance can be considerably increased with the use of appropriate features. As features, we use running mean and covariance. The mean of acceleration is used because it is one of the most accessible measures of time series data [83]. Mean acceleration is composed of three axial components, the means of x-axis ($\mu_x$), y-axis ($\mu_y$) and z-axis ($\mu_z$) movements. Use of mean of acceleration features has been shown to result in accurate recognition of activities [29], [5]. Covariance is selected because it is different in all the activities [71], [41]. Mean ($\mu$) and covariance ($\Sigma$) equations [24] can be seen on Eq. 3.1 and Eq. 3.2.

$$\hat{\mu} = \frac{1}{n} \sum_{k=1}^{n} x_k \tag{3.1}$$

$$\hat{\Sigma} = \frac{1}{n} \sum_{k=1}^{n} (x_k - \hat{\mu})(x_k - \hat{\mu})^T \tag{3.2}$$

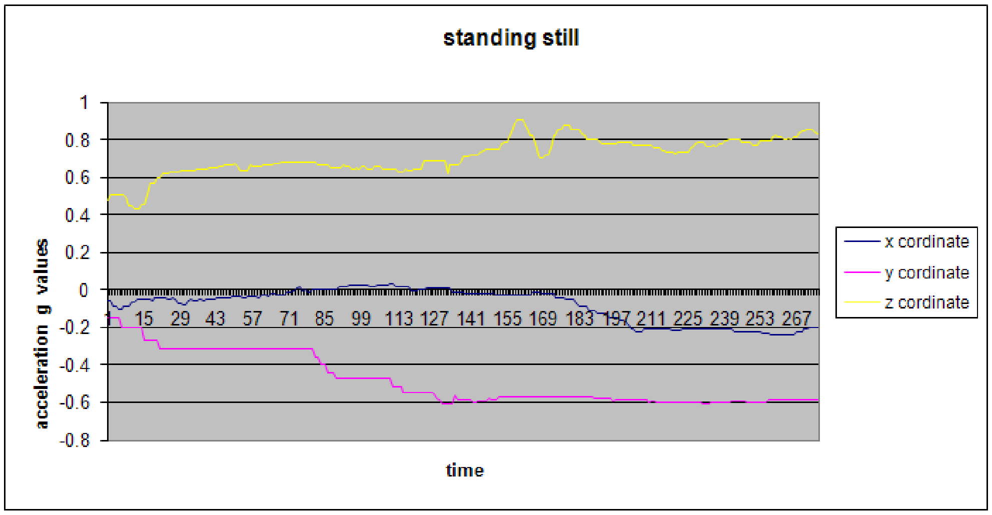


Figure 3.5: Acceleration over time in standing activity.

It is important to stress that the strength of these features discussed are person specific. These features allow recognition of user's activities. However, one person walking can give the same results as another person running [60].

Features are computed on 256 sample windows of acceleration data with 128 samples overlapping between consecutive windows. At a sampling frequency of 30 Hz, each window represents 8.5 seconds. Mean and covariance features are extracted from the sliding windows signals for activity recognition. Feature extraction on siding windows with %50 overlap has demonstrated success in the past works [21], [71]. A window of several seconds is used to sufficiently capture cycles in activities such as walking, running, or stairing down. The 256 sample window size enabled fast computation of mean and covariance.

### 3.2.3 Modelling Activities

To classify the acceleration data into distinct classes we employ a naive Bayes classifier. In this part we examine the naive Bayes classification algorithm. Naive Bayes classification is based on Bayes' rule from basic probability theory. Other, more complex, classifiers are available. However, we chose naive Bayes as our machine learning method because it is effective at classifying acceleration data

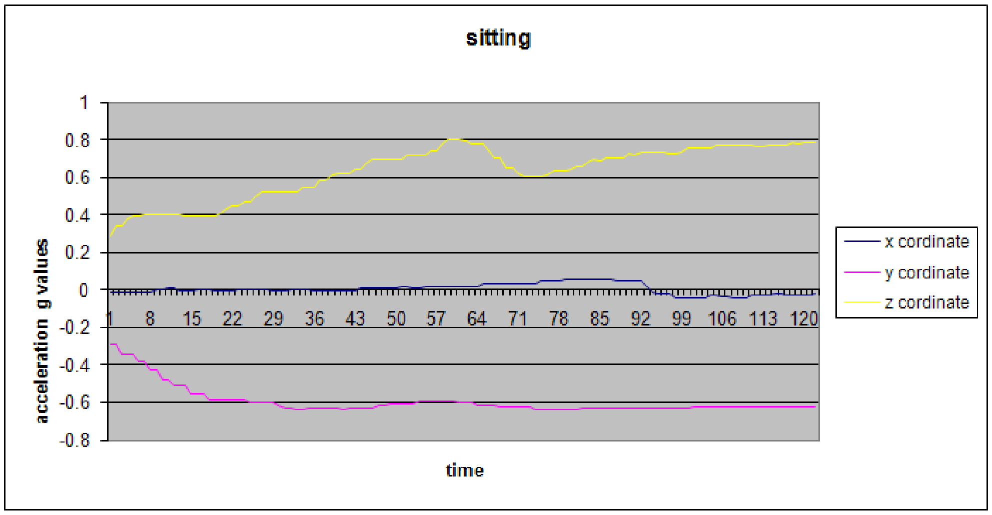


Figure 3.6: Acceleration over time in sitting activity.

[63] and classification using a trained model is computationally inexpensive [63]. Naive Bayes classifier also requires a small amount of training data to estimate the parameters necessary for classification. Because independent variables are assumed, only the variances of the variables for each class need to be determined and not the entire covariance matrix. Naive Bayes classifiers have worked quite well in many complex real-world situations. The Naive Bayes algorithm affords fast, highly scalable model building and scoring. It scales linearly with the number of predictors and rows.

Bayes' rule states that the probability of a given activity and an $n$-dimensional feature vector $x = \langle x_1, ....x_n \rangle$ can be calculated as in Eq. 3.3.

$$p\left(a|x\right) = \frac{p(x|a)}{p(x)} \tag{3.3}$$

$p\left(a\right)$ denotes the a-priori probability of the given activity. The a-priori probability $p\left(x\right)$ of data is just used for normalization. Since we are not interested in the absolute probabilities but rather the relative likelihoods, we can neglect $p\left(x\right)$. Assuming that the different components $x_i$ of the future vector $x$ are independent, we obtain a naive Bayes classifier which can be written as in Eq. 3.4.

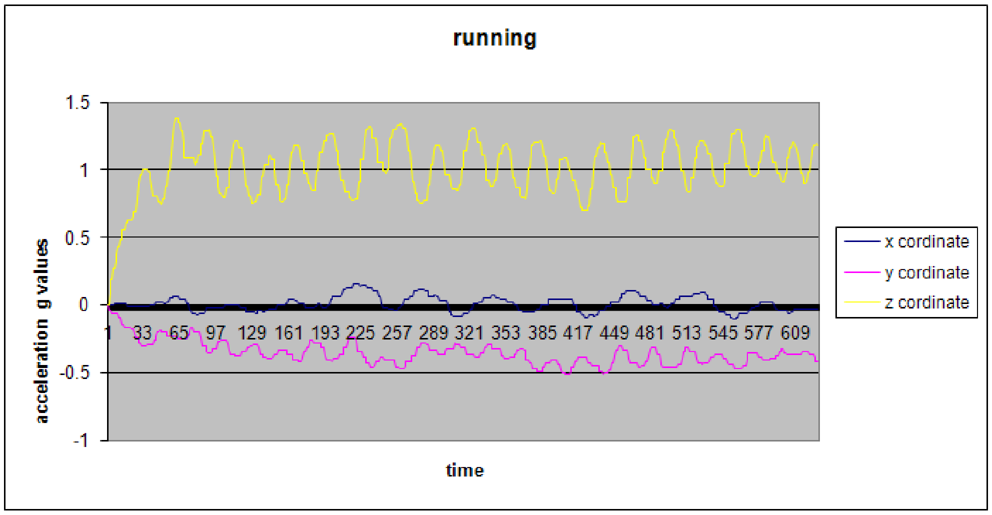


Figure 3.7: Acceleration over time in running activity.

$$p\left(a|x\right) = \prod_{i=1}^{n} p\left(x_i|a\right) \tag{3.4}$$

To calculate the statistical model, we use the calculations similar to the ones in Cakmakci's study [71]. We will assume that each context can be characterized with normal density with mean ($\mu$) and covariance matrix ($\Sigma$) (Eq. 3.1 and Eq. 3.2). Once we have $\mu$ and $\Sigma$ that estimates representing the distribution of each sample class, we can compute the Mahalanobis distance in order to classify sensor data in relation to the modeled data. We will choose the context class where the sensor data has a maximal probability of belonging that class. Eq. 3.5 [71] shows the calculation of the context for a given sensor input.

$$max(p(context_k|sensordata)) = max(p(sensordata|context_k)p(context_k)) \tag{3.5}$$

We substitute previously assumed normal density and take the logarithm of both sides and classify by taking the maximum for the log-like likelihood of each

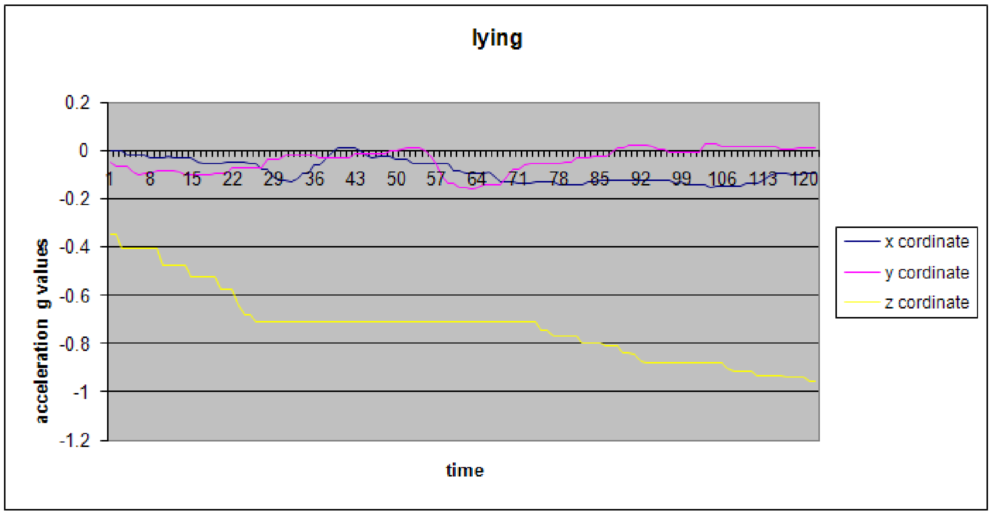


Figure 3.8: Acceleration over time in lying activity.

context class. This will make the mathematical manipulation easier since a Gausian is in form $e^x$ in the following Eq. 3.6 [71]:

$$max(p\left(context_k|sensordata\right)) = ln\left(\frac{1}{(2\pi)^{\frac{d}{2}}\left|\Sigma_{x_i}\right|^{\frac{1}{2}}}e^{-\frac{1}{2}(x_i-\mu_k)^T\Sigma^{-1}(x_i-\mu_k)}p(context)\right) \tag{3.6}$$

Which leads to the following Eq. 3.7 [71]:

$$\begin{aligned} max(p\left(context_k|sensordata\right)) &= max(-\frac{1}{2}(x_i-\mu_k)^T\Sigma^{-1}(x_i-\mu_k)) \\ &-max(-\frac{d}{2}ln(2\pi)-\frac{1}{2}ln\left|\Sigma_k\right|+ln(p(context_k))) \end{aligned} \tag{3.7}$$

We can ignore the additive constants such as the $\frac{d}{2}ln(2\pi)$ and $ln(p(context_k)))$ in the case where all contexts are equally likely in Eq. 3.7. We are left with the maximum of the negative Mahalanobis distance plus the negative logarithm of the determinant, which is equal to the minimum of the Mahalanobis distance with the logarithm of the covariance's determinant added in the following Eq. 3.8 [71]:

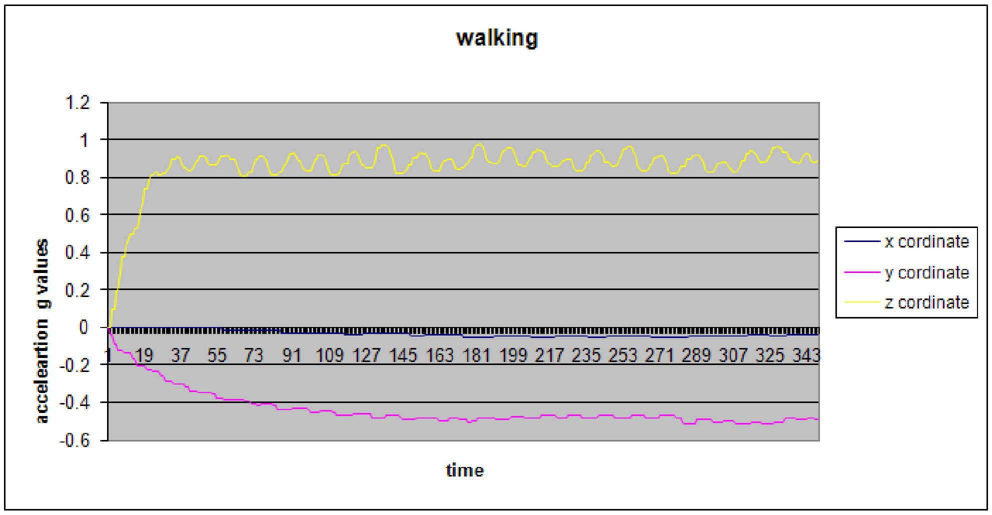


Figure 3.9: Acceleration over time in walking activity.

$$max(p(context_k|sensordata)) = min(ln\,|\Sigma_k| + (x_i - \mu_k)^T \Sigma^{-1}(x_i - \mu_k)) \quad (3.8)$$

## 3.3 Signal Filtering

The processing capacity of the handheld device is very crucial. Therefore, filtering algorithms should not be CPU intensive and number of the accelerometers should be minimum (one in this paper). The signal filtering techniques that are used are similar to Jang's work [35]. The signal filtering system can be seen in Figure 3.10.

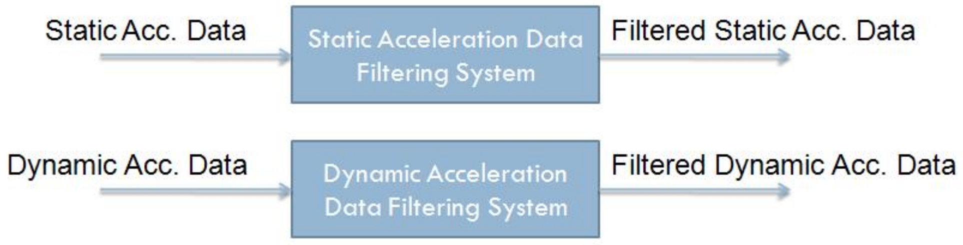

Static Acc. Data
Static Acceleration Data Filtering System
Filtered Static Acc. Data
Dynamic Acc. Data
Dynamic Acceleration Data Filtering System
Filtered Dynamic Acc. Data

Figure 3.10: Signal Filtering System Architecture.

There are two kinds of data that we use for gesture recognition. One is dynamic acceleration (vibration of the device) and the other is static acceleration (gravity or tilt of the device). We will use two kinds of approach: one signal processing approach for static acceleration, another for dynamic acceleration.

### 3.3.1 Static Acceleration

Static acceleration can detect position changes from device's early status. Static acceleration decides which direction the position has changed, compared to its original position. Two filters are used for static acceleration: low pass filtering and thresholding. The steps of signal processing for static acceleration can be seen in Figure 3.11.

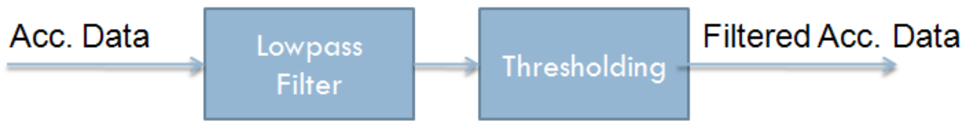


Figure 3.11: The steps of signal processing for static acceleration.

#### Low pass filtering

Low pass filtering is a process to make signal changes consecutive by processing trivial movement. First order Butterworth filter is used to filter the data because calculations in the filter are not CPU intensive and this filter is an effective one [35]. The gain $G(w)$ of an n-order Butterworth low pass filter is given in Eq. 3.9.

$$G^2(w) = \frac{G_0^2}{1 + (\frac{w}{w_c})^{2n}} \tag{3.9}$$

Where n is order of filter, $w_c$ is the cutoff frequency, and $G_0$ is the DC gain (gain at zero frequency). Cutoff frequency changes according to the context information. Emprically calculated cutoff frequency of the activity labels can be seen in Table 3.1.

### Thresholding

If detected signals are lower than a threshold value, then such signals are ignored. A threshold range is defined and only input data outside this range is considered. This threshold range is $\pm 0.2G$. Jang and Park define [35] this threshold range to be $\pm 0.2G$ based on applied experiments and simply discard the acceleration data inside this range. This $\pm 0.2G$ range should be changed according to the context information. Emprically calculated threshold values of the activity labels can be seen in Table 3.1.

Table 3.1: Threshold and cutoff frequency values of the activity labels.

| **Activity Label** | **Threshold Value(G)** | **Cutoff Frequency(G)** |
|---|---|---|
| Standing Still | 0.20 | 1.20 |
| Walking | 0.30 | 1.40 |
| Sitting | 0.10 | 1.00 |
| Lying | 0.15 | 1.10 |
| Running | 0.40 | 1.70 |
| Stairing Up | 0.35 | 1.50 |
| Stairing Down | 0.35 | 1.50 |

## 3.3.2 Dynamic Acceleration

Dynamic acceleration happens when a sudden movement or a short shock is transmitted to accelerator. Three filters are used for dynamic acceleration: high-pass filtering, boundary filtering and debouncing filtering. The steps of signal processing for dynamic acceleration are shown in Figure 3.12.

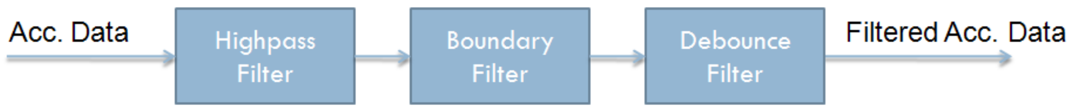


Figure 3.12: The steps of signal processing for dynamic acceleration.

### High pass filtering

A highpass filter is the just opposite of a lowpass filter: to offer easy passage of a high frequency signal and difficult passage to a low frequency signal. Butterwoth Filter is used for high pass filtering. The gain $G(w)$ of an n-order Butterworth filter is given in Eq. 3.9. Cutoff frequency changes according to the context information. Emprically calculated cutoff frequencies according to the activity labels can be seen on Table 3.2.

### Boundary Filtering

It is a process to eliminate trivial signals to prevent unwanted gestures arising from slight vibration or movements. Between $\alpha - \beta$ are eliminated. This $\alpha - \beta$ range changes according to context information. Emprically calculated $\alpha - \beta$ range according to the activity labels can be seen in Table 3.2.

### Debouncing Filtering

This is a process not to recognize several peaks as gestures. It ignores multiple peaks. The Intensity of this process changes according to context information. Emprically calculated intensity values are shown in Table 3.2.

Table 3.2: $\alpha - \beta$ range, cutoff frequency values and intensity of debouncing filter of the activity labels.

| **Activity Labels** | $\alpha - \beta$ **Range** | **Intensity** | **Cutoff Frequency** |
|---|---|---|---|
| Standing Still | -0.20 to 0.20 | 0.2 | 0.2 |
| Walking | -0.30 to 0.30 | 0.3 | 0.3 |
| Sitting | -0.10 to 0.10 | 0.1 | 0.1 |
| Lying | -0.15 to 0.15 | 0.1 | 0.1 |
| Running | -0.40 to 0.40 | 0.35 | 0.3 |
| Stairing Up | -0.35 to 0.35 | 0.25 | 0.2 |
| Stairing Down | -0.35 to 0.35 | 0.25 | 0.2 |

# Chapter 4

# Gesture Design and Gesture Recognition

## 4.1 Gesture Design

Despite the fact that accelerometer sensor and touch screen are used as gestural input devices, the gestures defined for these devices are not standardized. The user needs to learn every gesture when changing the handheld device. Every handheld device has own gesture library and gesture meanings so interaction with touch screen and accelerometer sensor are not easy to learn and learning the gesture libraries takes long time. Since the possible user range is wide, gesture library should be simple and intuitive enough for the users with basic knowledge; however it should satisfy the needs of a complex user at the same time. In this study we proposed a standardized gesture library for accelerometer sensor and touch screen to combine all the gestures and name the functions of these gestures.

In this study, gestures are studied as an interaction mode adding more meaning to usage of a handheld device. When designing a gesture we need to find a real world example of using the handheld device. The abstraction of the user interface of the handheld devices has a wide gap. To narrow this gap, meaningful gestures need to be designed.

### 4.1.1 Gesture Design Principles

When we are designing the accelerometer gestures, we use the design principles in Prekopcsak's work [59]. Four design principles are examined for an everyday gesture interface: ubiquity, unobtrusiveness, adaptability and simplicity. The design of the accelerometer gestures is an iterative process, because new ideas rise after every finished development step. The following is the four design principles that we use when designing the accelerometer gestures.

#### Ubiquity

The gesture interface should be available everywhere, not restricted to time and place. Most gesture recognizer systems are usually based on video cameras, which makes them immovable and they only works well in controlled environments. Developed gestures are designed to use everywhere and anytime, when walking, running, lying, and etc.

#### Unobtrusiveness

Unobtrusive or inconspicuous means that we barely notice that we are using an interface[59]. The gestures should be used without special controllers or gloves and with everyday clothes. Our gestures are very intuitive and do not need special devices or clothes.

One of the most important features is to cover interface details from the user. Thus, we need to avoid special rules for the use of the system. Most accelerometer based prototypes have a button with which the user presses at the start and at the end of the gesture. It is extremely reliable but can be disturbing. In our design, there is no need for a button as the gestures are automatically extracted from the continuous sensor data stream.

Another very important feature is the response time. Usability engineering books suggest that response time should be as low as 100ms[53]. Our gesture recognition time is between 1-100 ms.

#### Adaptability

All possible functions of the gestures need to be considered when designing a gesture interface, therefore the users can use the gestural interface in every application in a single meaning. Designing a universal gesture set is good design principle because users do not need to learn new gestures when they change the application or device. The users also do not need to learn about the meaning of all the gestures because meanings are same independent from application.

**Simplicity**

The interface should be easy to learn and use in a few minutes. The system shouldn't have high expectations about the user, and provide feedback about successful gesture recognition. As gesture recognition will never reach 100%, it is important to inform the user about what is happening with the system. Successful recognition can be signed with a visual feedback, so the user instantly realizes if it is needed to repeat the gesture.

### 4.1.2 Accelerometer Gestures

We have designed five gesture types. These are 'primitive', 'move', 'page', 'global' and 'other' gesture types. Primitive acceleration events are joystick movements. When we design the primitive acceleration events, we map the mobile device to a joystick. User moves the stick of the joystick device to have a move event, this movement is same on the mobile device. Mobile device is thought as the stick of the joystick. When user moves the mobile device, move events are created. Also primitive acceleration events are designed considering the gravity forces. Gravity forces objects to slide towards to the centre of the earth, this is a nature phenomenon. If a table has an angle to the right, objects on the table slide to the right due to the gravity forces. In joystick movements, screen has an angle to the right/left/up /down to make the objects on the screen slide to the right/left/up /down. Primitive acceleration events can be seen in Table 4.1

Table 4.1: Joystick Movements

| **Gesture Type** | **Gesture Name** | **Accelerometer Movements** |
|---|---|---|
| Acc Event | Primitive Event | In X-Z plane, angle to +x. |
| Acc Event | Primitive Event | In X-Z plane, angle to -x. |
| Acc Event | Primitive Event | In X-Z plane, angle to +y. |
| Acc Event | Primitive Event | In X-Z plane, angle to -y. |

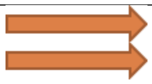
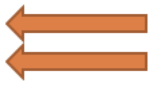
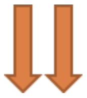
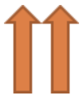
The move gesture types are the gestures that we are using to map the arrow keys of the keyboard. These gestures designed to move the indicator. When user needs to move an object in a specified direction, she throws it to that direction. Like in a real world, move gestures have the same principle. User throws indicator to move it. These gestures are throwing actions, user throws indicator to desired direction. So indicator jumps to that direction. Move gesture type can be seen in the following Table 4.2

Table 4.2: Move Gesture Type

| **Gesture Type** | **Gesture Name** | **Accelerometer Movements** | **Plane** |
|---|---|---|---|
| Move | GSRight | → | X-Y |
| Move | GSLeft | ← | X-Y |
| Move | GSDown | ↓ | X-Y |
| Move | GSUp | ↑ | X-Y |

The page type of gestures is designed to handle more complex gestures than simple movement gestures. They are designed to map keyboard's page up, page down, end and home buttons and extended to the current usage of applications. The gestures in this type generally map to complex movements in applications. This type of gestures are basically same as the move gestures but not a single move gesture is enough; the user needs to do move gesture more than one to initialize the page gesture. Page gesture can be seen on Table 4.3

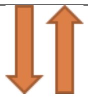
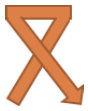
Table 4.3: Page Gesture Type

| **Gesture Type** | **Gesture Name** | **Accelerometer Movements** | **Plane** |
|---|---|---|---|
| Page | GSPageRight |  | X-Y |
| Page | GSPageLeft |  | X-Y |
| Page | GSPageDown |  | X-Y |
| Page | GSPageUp |  | X-Y |

The global type of gestures is designed to handle global functions of the applications such as enter, exit. The enter gesture is one of the global gestures. 'Click' on the mouse, 'tap' on the touch screen is a standard way to enter. On the accelerometer gesture system, one standard gesture is needed to map the 'click' and 'tap'. When user taps the touch screen, device has a movement to back and forward. So this motion is used to map the enter gesture.

The next global gesture is exit gesture. 'X' is a common character to indicate exit and/or close functions. When user draw 'X' to the air with the mobile device, it means user want to call exit gesture. Global gesture type can be seen on following Table 4.4

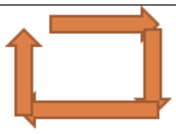
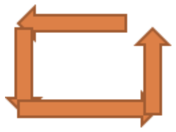
Table 4.4: Global Gesture Type

| Gesture Type | Gesture Name | Accelerometer Movements | Plane |
|---|---|---|---|
| Global | GSEnter |  | Y-Z |
| Global | GSExit |  | X-Z |

The last type of the gesture is other type of gestures. This type of gestures is designed to handle other type of functions such as zoom in and zoom out. This type of gestures makes users feel they are screwing/unscrewing or opening up/closing up the pictures and documents to make them bigger/smaller [11] and make users feels they are pulling/pushing the pictures or documents from each side, to make them expands/smaller under the pulling/pushing forces.

The last gesture is isFront gesture. This gesture indicate that phone is upside down or else. Other type of gestures can be seen on Table 4.5

Table 4.5: Other Gesture Type

| Gesture Type | Gesture Name | Accelerometer Movements | Plane |
|---|---|---|---|
| Other | GSSquare |  | X-Z |
| Other | GSCcSquare |  | X-Y |
| Other | GSIsFront | When device is turned front or back, this gesture is triggered. | X-Y |

### 4.1.3 Touch Screen Gestures

The very basic touch screen data is coordinates on a plane (screen) and the value of whether an object is touching on the surface or not. By using these primitive data we can build complex input systems such as continuous movement events and gestures libraries. To activate the basic touch screen functionality, the user simply touches the screen and the system understands the point of touch, is the object still pressed or not, the object is moved on the screen without releasing it. By this way our data is generated.

Primitive touch screen events are created according to the direct manipulation principles. When a user wants to point, activate or perform whatever action is possible at that time, he/she just touches the screen where the object is on the screen. Since touch screen is capable of 2D input and we have to use a touched/not touched state the only primitive event generated by touch screen is the current location generated by a touch of an object on the screen. Primitive acceleration events can be seen in Table 4.6

Table 4.6: Touch Screen Event

| **Gesture Type** | **Gesture Name** | **Touch Screen Action** |
|---|---|---|
| Touch Event | Primitive Touch Event | Touch on a location of the screen |

The move gesture types are the gestures that we are using to map the arrow keys of the keyboard. These gestures designed to move the indicator. Drawing a line to a direction to represent a gesture is a natural drawing habit. Since usage of touch screen is a reflection to drawing or writing, this gesture is selected to reflect everyday habits most naturally.

User can draw line to right, left, up and down direction to specify related gesture. Move gesture type of touch screen can be seen on the following Table 4.7

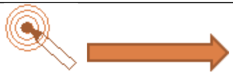
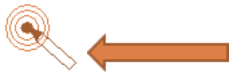
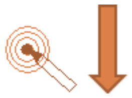
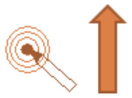
Table 4.7: Move Gesture Type

| **Gesture Type** | **Gesture Name** | **Touch Screen Action** |
|---|---|---|
| Move | GSRight | → |
| Move | GSLeft | ← |
| Move | GSDown | ↓ |
| Move | GSUp | ↑ |

This type of gestures is the easiest of two-level gestures since it requires least effort. They are designed to map keyboard's page up, page down, end and home buttons and extended to the current usage of applications. Tapping is natural since it reminds clicking. Combining it with direct one-way movement gives us less effort two level gesture. It also reminds one level direction gestures and so this gesture can be used in similar works.

User can perform this type of gestures in four directions, right, left, up and down direction to specify related gesture. Page gesture type of touch screen can be seen on the following Table 4.8

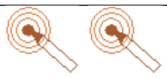
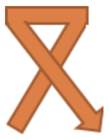
Table 4.8: Page Gesture Type

| **Gesture Type** | **Gesture Name** | **Touch Screen Action** |
|---|---|---|
| Page | GSPageRight |  |
| Page | GSPageLeft |  |
| Page | GSPageDown |  |
| Page | GSPageUp |  |

The global type of gestures is designed to handle global functions of the applications such as enter, exit. GSEnter gesture is to simulate enter situations or double clicking situations for mouse. Since mouse is fairly close to touch screen, describing double clicking with double tapping is reasonable. On the other hand, in most operating systems, double tapping is the way to use entering.

The second gesture in this type is GSExit. Cross symbol is generally accepted to express exit situations. So the idea while defining GSExit gesture like this was to represent common sense. Also exit includes an x letter in it. This strengthens the feeling of drawing an X to exit. Global gesture type can be seen on following Table 4.9

Table 4.9: Global Gesture Type

| **Gesture Type** | **Gesture Name** | **Touch Screen Action** |
|---|---|---|
| Global | GSEnter |  |
| Global | GSExit |  |

The last type of the gesture is other type of gestures. This type of gestures is designed to handle other type of functions such as Tap, Erase, Putting Space, etc.

GSTap gesture is directly driven from touch screen enabled operating systems. It generally is used to click left button of mouse. Since it requires the least effort, it is the easiest gesture to perform. GSTapnWait gesture is directly driven from touch screen enabled operating systems. It generally is used to demonstrate the right button of mouse. Since it requires the least effort after just tapping, GSTapnWait is selected to perform options property.

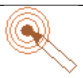
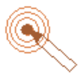
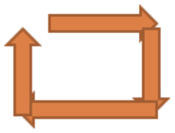
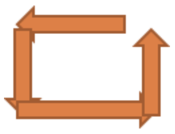
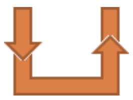
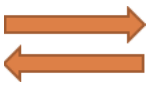
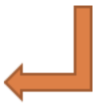
Besides one or two level gestures we also thought that we need higher level gestures to identify complex inputs. These gestures should not be confused with low-level gestures. Square was the easiest four-level gesture since it includes both directions in a smooth manner. GSSquare and GSCcSquare are defined that way.

GSSpace gesture is defined to express common space symbol in writing. It also gives the expression to separate things. So the reason to define GSSpace this way was both for describing a routine or giving the feeling to put a space.

GSErase is defined to express common scratch out movement in writing. It also gives the expression to blacken on something. So the reason to define GSErase this way was both for describing a routine or giving the feeling of scratching out.

GSNewLine gesture is defined to handle new line input. It is defined as the symbol of new line in computer keyboard. It is easy to perform since it is a two level gesture. A set of all other gesture type can be seen on Table 4.10

Table 4.10: Other Gesture Type

| **Gesture Type** | **Gesture Name** | **Touch Screen Action** |
|---|---|---|
| Other | GSTap |  |
| Other | GSTapnWait |  |
| Other | GSSquare |  |
| Other | GSCcSquare |  |
| Other | GSSpace |  |
| Other | GSErase |  |
| Other | GSNewLine |  |

### 4.1.4 Standardized Gesture Library

This part of the section explains the core of the study. Previous gesture systems do not include a combined gesture library to abstract different types of inputs. For a point of view of a designer or a coder, it is very important to see the input as one type of event instead of different device events. Different type of input methods can be intended to create the same type of events. So combining these two inputs allows the designer to deal with the problem only and not to consider different type of devices. He/she only deals with the intended action. For example a right draw in touch screen and a right move in acceleration sensor can mean the same input. So instead of dealing with two libraries and getting the input separately, the coder just gets the intended action from the event queue.

As mentioned earlier, the system is thought as a complete library. This gives us the opportunity to enhance adaptability. We also worked on the grouping of the gestures. This was to help the designer or coder to adapt the intended gestures and eliminate the others.

While combining these different gestures the primary concern was the compatibility. Both inputs from different input devices should be intended to perform the same type of functionality. With this manner, we consider which actions and movements are understood as synonym and as a result we created a general gesture as a mapping of the both real life action. Some gestures for different input devices do not have a synonym gesture. In that case, a general gesture still is created but only that device can create that general gesture. Thus two different types of gestures are combined in a general and abstract gesture library. More input devices can be added to the system. A list of Move gestures for both touch screen and acceleration sensor can be seen on Table 4.11

Table 4.11: Move Gesture Type

| **Standardized Gesture** | **Touch Screen** | **Acceleration Sensor** | **Acc. Plane** |
|---|---|---|---|
| GSRight | → | → | X-Y |
| GSLeft | ← | ← | X-Y |
| GSDown | ↓ | ↓ | X-Y |
| GSUp | ↑ | ↑ | X-Y |

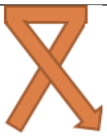
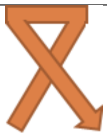
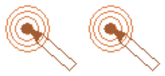
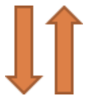
A list of Page gestures for both touch screen and acceleration sensor can be seen on Table 4.12

Table 4.12: Page Gesture Type

| **Standardized Gesture** | **Touch Screen** | **Acceleration Sensor** | **Acc. Plane** |
|---|---|---|---|
| GSPageRight | ◎ → | ⇉ | X-Y |
| GSPageLeft | ◎ ← | ⇇ | X-Y |
| GSPageDown | ◎ ↓ | ⇊ | X-Y |
| GSPageUp | ◎ ↑ | ⇈ | X-Y |

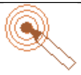
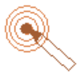
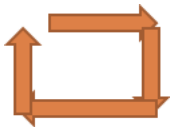
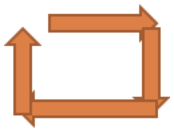
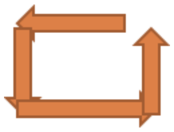
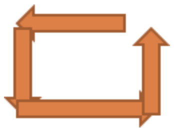
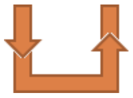
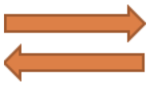
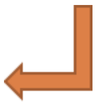
A list of Global gestures for both touch screen and acceleration sensor can be seen on Table 4.13

Table 4.13: Glocal Gesture Type

| **Standardized Gesture** | **Touch Screen** | **Acceleration Sensor** | **Acc. Plane** |
|---|---|---|---|
| GSExit |  |  | X-Z |
| GSEnter |  |  | Y-Z |

A list of Other gestures for both touch screen and acceleration sensor can be seen on Table 4.14 Remember that some of the gestures cannot be performed with one of the input devices.

Table 4.14: Other Gesture Type

| **Standardized Gesture** | **Touch Screen** | **Acceleration Sensor** |
|---|---|---|
| GSTap |  | Not Available |
| GSTapnWait |  | Not Available |
| GSSquare |  |  |
| GSCcSquare |  |  |
| GSSpace |  | Not Available |
| GSErase |  | Not Available |
| GSNewLine |  | Not Available |
| GSIsFront | Not Available | When device is turned front or back. |

### 4.1.5 Application Function Suggestion

Application designers are not limited to design functions to assign for the gestures but we suggest some functions for the gestures. In the following Tables 4.15, 4.16, 4.17, 4.18, 4.19, 4.20 the function suggestions for the specified applications can be found.

Table 4.15: The Function Suggestions for the Specified Applications on Primitive Events

| **Gesture Type** | **Gesture Name** | **Application Function Suggestion** |
|---|---|---|
| Acc Event | Primitive Right | Games: Move object to right.<br>Operating Systems: Move cursor to the right. |
| Acc Event | Primitive Left | Games: Move object to left.<br>Operating Systems: Move cursor to the left. |
| Acc Event | Primitive Up | Games: Move object to up.<br>Operating Systems: Move cursor to the up. |
| Acc Event | Primitive Down | Games: Move object to down.<br>Operating Systems: Move cursor to the down. |

Table 4.16: The Function Suggestions for the Specified Applications on Move Gestures

| **Gesture Type** | **Gesture Name** | **Application Function Suggestion** |
|---|---|---|
| Move | GSRight | Media Player: Fast forward. |
| | | Web Browser: Move page right. |
| | | Photo Album: Move image right. |
| | | MMI: Move cursor right. |
| | | Calendar: Move cursor right. |
| | | Contacts: Move cursor right. |
| Move | GSLeft | Media Player: Rewind. |
| | | Web Browser: Move page left. |
| | | Photo Album: Move image left. |
| | | MMI: Move cursor left. |
| | | Calendar: Move cursor left. |
| | | Contacts: Move cursor left. |
| Move | GSUp | Media Player: Volume up. |
| | | Web Browser: Move page up. |
| | | Photo Album: Move image up. |
| | | MMI: Move cursor up. |
| | | Calendar: Move cursor up. |
| | | Contacts: Move cursor up. |
| Move | GSDown | Media Player: Volume down. |
| | | Web Browser: Move page down. |
| | | Photo Album: Move image down. |
| | | MMI: Move cursor down. |
| | | Calendar: Move cursor down. |
| | | Contacts: Move cursor down. |

Table 4.17: The Function Suggestions for the Specified Applications on Page Gestures

| **Gesture Type** | **Gesture Name** | **Application Function Suggestion** |
|---|---|---|
| Page | GSPageRight | Media Player: Next track. |
| | | Web Browser: Page left. |
| | | Photo Album: Go to last image. |
| | | MMI: -. |
| | | Calendar: Next month. |
| | | Contacts: Page right. |
| Page | GSPageLeft | Media Player: Previous track. |
| | | Web Browser: Page right. |
| | | Photo Album: Go to first image. |
| | | MMI: -. |
| | | Calendar: Previous month. |
| | | Contacts: Page left. |
| Page | GSPageUp | Media Player: -. |
| | | Web Browser: Page up. |
| | | Photo Album: Previous image. |
| | | MMI: Previous page of icons. |
| | | Calendar: Previous year. |
| | | Contacts: Page up. |
| Page | GSPageDown | Media Player: -. |
| | | Web Browser: Page down. |
| | | Photo Album: Next image. |
| | | MMI: Next page of icons. |
| | | Calendar: Next year. |
| | | Contacts: Page down. |

Table 4.18: The Function Suggestions for the Specified Applications on Global Gestures

| **Gesture Type** | **Gesture Name** | **Application Function Suggestion** |
|---|---|---|
| Global | GSEnter | Media Player: Play/Stop. |
| | | Web Browser: Enter (For selected forms). |
| | | Photo Album: Full screen/Normal window. |
| | | MMI: Enter |
| | | Calendar: Enter |
| | | Contacts: Enter |
| Global | GSExit | Media Player: Exit. |
| | | Web Browser: Exit. |
| | | Photo Album: Exit. |
| | | MMI: Exit. |
| | | Calendar: Exit. |
| | | Contacts: Exit. |

Table 4.19: The Function Suggestions for the Specified Applications on Other Gestures

| Gesture Type | Gesture Name | Application Function Suggestion |
|---|---|---|
| Other | GSTap | Media Player: Pause.<br>Web Browser: Click (On widget).<br>Photo Album: -.<br>MMI: Click.<br>Calendar: Click.<br>Contacts: Click. |
| Other | GSTapnWait | Media Player: -.<br>Web Browser: Request context menu.<br>Photo Album: Request context menu.<br>MMI: Request context menu.<br>Calendar: Request context menu.<br>Contacts: Request context menu. |
| Other | GSSquare | Media Player: -.<br>Web Browser: Zoom in.<br>Photo Album: Zoom in.<br>MMI: -.<br>Calendar: Year¿Month¿Day view switch.<br>Contacts: Show detailed view. |
| Other | GSCcSquare | Media Player: -.<br>Web Browser: Zoom out.<br>Photo Album: Zoom out.<br>MMI: -.<br>Calendar: Day¡Month¡Year view switch.<br>Contacts: Show general view. |

Table 4.20: The Function Suggestions for the Specified Applications on Other Gestures

| **Gesture Type** | **Gesture Name** | **Application Function Suggestion** |
|---|---|---|
| Other | GSSpace | Media Player: -. |
| | | Web Browser: Page down. |
| | | Photo Album: Slide show. |
| | | MMI: -. |
| | | Calendar: -. |
| | | Contacts: -. |
| Other | GSErase | Media Player: -. |
| | | Web Browser: -. |
| | | Photo Album: Delete. |
| | | MMI: -. |
| | | Calendar: Delete item's meetings. |
| | | Contacts: Delete contact. |
| Other | GSNewLine | Media Player: -. |
| | | Web Browser: -. |
| | | Photo Album: Photo shoot. |
| | | MMI: -. |
| | | Calendar: Shortcut to add new meeting. |
| | | Contacts: Shortcut to add new contact. |
| Other | GSIsFront | Media Player: Mute/Unmute. |
| | | Web Browser: -. |
| | | Photo Album: -. |
| | | MMI: -. |
| | | Calendar: -. |
| | | Contacts: -. |

## 4.2 Gesture Recognition

The gesture recognizer should:

- be resilient to variations in sampling due to movement speed or sensing,
- support optional and configurable rotation, scale, and position invariance,
- require no advanced mathematical techniques (e.g., matrix inversions, derivatives, integrals),
- be easily written in few lines of code,
- be fast enough for interactive purposes.

With these goals in mind, we describe the gesture recognizer. For primitive acceleration gesture type, static acceleration data is used. Therefore, we need to calculate the angle of the device from static acceleration data. Changing of the angle of the device creates primitive gesture types. A simple trigonometric calculation based on the unit circle (Figure 4.1) is used when trying to determine the tilt angle of the device. Although gravitational acceleration is limited to 1G by definition, the accelerometer we have used (Phidget Accelerometer [3]) measures both static and dynamic acceleration up to 3G, and can easily output values greater than 1G during a tilt measurement if the device is in motion. Accelerations in excess of 1G are not valid during tilt measurements; values should be limited to this maximum to ensure that the formula below remains appropriate [3].

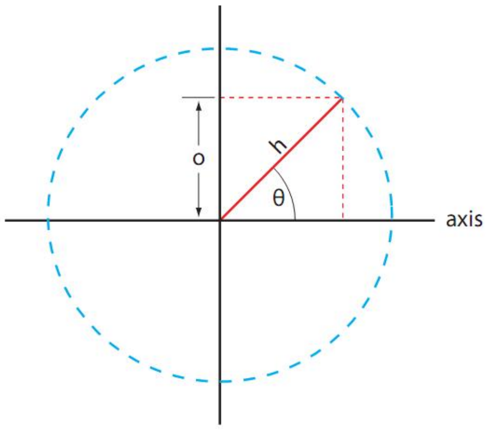


Figure 4.1: The unit circle representation of acceleration data.

The unit circle represents the maximum gravitational acceleration of 1G. A line drawn from the origin out to the unit circle then represents the acceleration vector of the device in a given axis. For that axis, the angle Θ, which represents the tilt angle of the device in that axis, can be calculated using the length of the hypotenuse and the distance of the end point of the hypotenuse from the axis in question (labeled o, which represents the value of acceleration reported by the device). The formula [3] can be seen on can be seen on Eq. 4.1.

$$\Theta = arcsin(o/h) \tag{4.1}$$

In an example, the device reports the acceleration of axis 0 to be 0.7071G. The calculation of $arcsin(0.7071G/1G)$ yields the result of 45 degree.

To recognize the move gesture type, we use dynamic acceleration data. We use a finite state machine. Move gesture recognizer finite state machine can be seen in Figure 4.2.

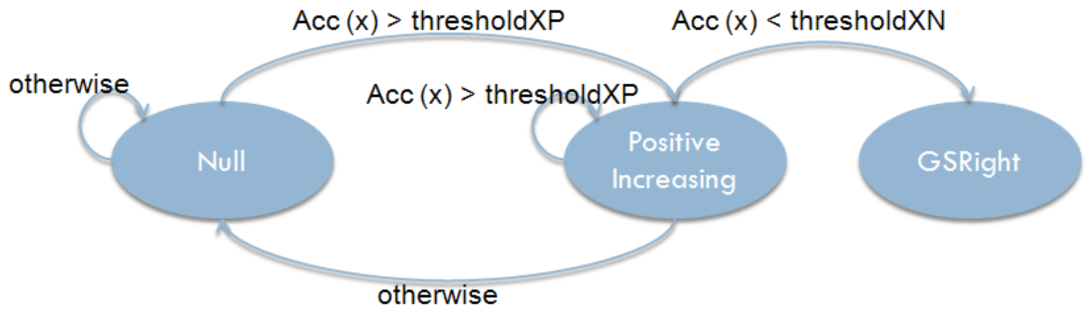


Figure 4.2: Move type GSRight gesture recognizer finite state machine.

To recognize move gesture type we examine dynamic acceleration data on X and Y coordinates. Acceleration data on x axis need to be positive increasing and then need to be positive decreasing to recognize GSRight. States and coordinate axis that are needed to recognize move gesture type can be seen on Table 4.21. Page gestures are multi level move gestures. When two sequential gsright gestures are entered, system identifies the gesture as GSPageright. And this recognition is same for all the page gestures. Two sequential GSLeft is GSPageleft. Two sequential GSDown is GSPagedown and two sequential GSUp is GSPageup.

Table 4.21: States and coordinate axis that are needed to recognize move gesture type.

| **Move gesture Type** | **Axis** | **States that are needed** |
|---|---|---|
| GSRight | X | positive increasing then positive decreasing. |
| GSLeft | X | negative increasing then negative decreasing. |
| GSUp | Y | positive increasing then positive decreasing. |
| GSDown | Y | negative increasing then negative decreasing. |

To recognize the enter gesture type, we use dynamic acceleration data. We use a finite state machine. Enter gesture recognizer finite state machine can be seen in Figure 4.3. GSEnter is a tab gesture. Since system needs to identify the positive increasing and positive decreasing sequentially on Z axis.

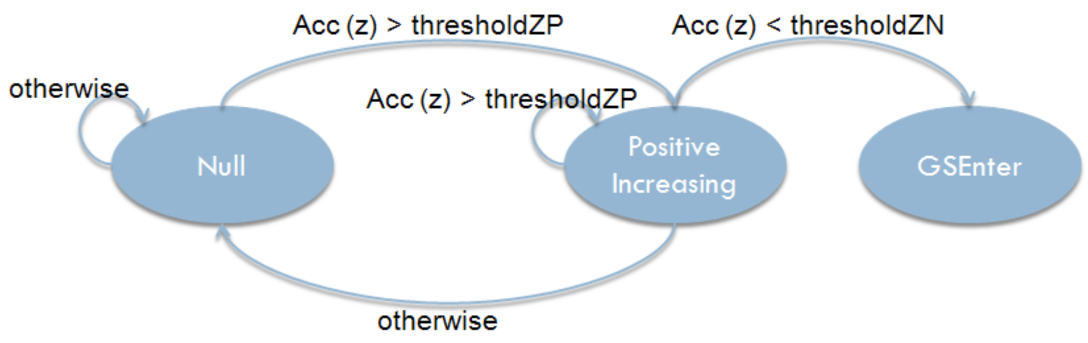


Figure 4.3: Global type GSEnter gesture recognizer finite state machine.

To recognize the exit gesture type, we use dynamic acceleration data. We use a finite state machine. Exit gesture recognizer finite state machine can be seen in Figure 4.4. GSExit is more complicated then move gestures or enter gesture. The gesture is on X axis. System recognizes the exit gesture by the sequential movements of negative increasing, negative decreasing, positive increasing and positive decreasing.

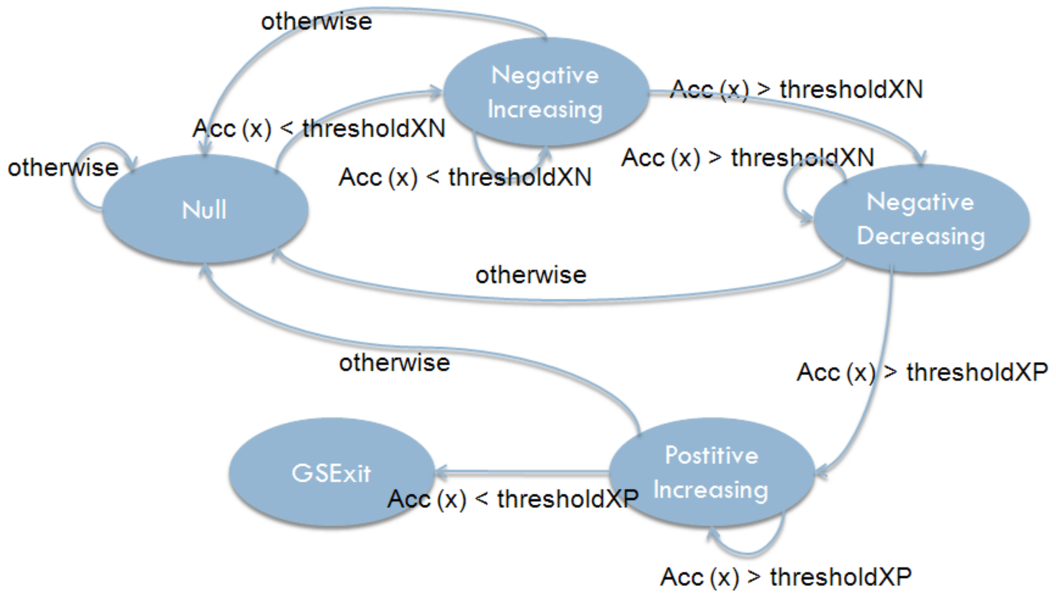


Figure 4.4: Global type GSExit gesture recognizer finite state machine.

To recognize the square gesture type, we use dynamic acceleration data. We use a finite state machine. Square gesture recognizer finite state machine can be seen in Figure 4.5. Recognition of square and counter clockwise square gesture type is different from the other gestures. Square and counter clockwise square is recognized on both X and Y axis. System recognizes the square gesture by the sequential movements of negative increasing on X axis, negative increasing on Y axis, positive increasing on X axis and positive increasing on Y axis. States and coordinate axis that are needed to recognize square and counter clockwise square can be seen on Table 4.22.

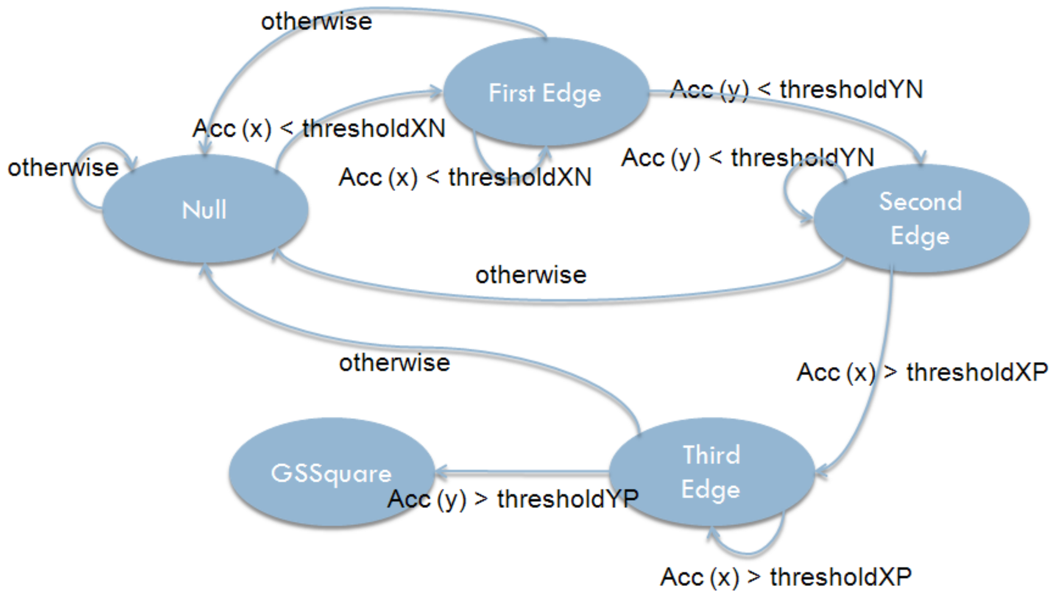


Figure 4.5: Other type GSSquare gesture recognizer finite state machine.

Table 4.22: States and coordinate axis that are needed to recognize other gesture type.

| **Move gesture Type** | **Axis** | **States that are needed** |
|---|---|---|
| GSSquare | X and Y | negative increasing on X axis, then negative increasing on Y axis, then positive increasing on X axis, then positive increasing on Y axis. |
| GSCCSquare | X and Y | positive increasing on X axis, then negative increasing on Y axis, then negative increasing on X axis, then positive increasing on Y axis. |

Recognition of Isfront gesture is relatively easy. If static value Z axis of the device is positive, gesture is created with boolean front = true. If static value Z axis of the device is negative, gesture is created with boolean front = false.

# Chapter 5

# Experiments and Evaluations

In order to evaluate the success of the proposed gesture recognition accuracy enhancement system, we have performed a number of objective and subjective experimental studies. In this chapter, we discuss these experimental studies and their results in detail.

In this study, we selected two objective and one subjective experiment. One experiment is to evaluate the accuracy of the context aware part of the system, the other is to test the gesture recognition accuracy enhancement system and the last one is to evaluate the interaction method, subjectively. The following sections present detailed information about these experiments.

## 5.1 Context Aware Experiment

The purpose of this experiment is to test the accuracy of the context aware part of the system. The details of this experiment are explained in the following subsections.

**Subjects**

The objective experiment was performed on 9 subjects: 6 males and 3 females with a mean age of 24.8. They were voluntary graduate and undergraduate students with computer science background. The purpose of the experiment was not explained to the subjects.

#### Procedure

In this experiment, an experimental setup similar to the one in Bao's study [7] is used. On two consecutive days each test person participated in a data collection session. In each session they performed each activity once for approximately two minutes while they are reading text on the screen. We trimmed several seconds off the start/end to balance the data for our analysis and we used 5 window size data in the middle. The data collected in the first session is used for training the context aware part of the system and the data collected in the second session is used for testing the context aware part.

#### Results and Discussion

Context aware classifier was trained and tested using two protocols: within-person and cross-person protocols. Under the within person protocol, for each of the nine people, we built a model using their own first session data and tested their second session data. This user-specific training protocol was repeated for all nine subjects. The mean accuracy was 92.0%. The classification accuracy results on within person protocol can be seen on Table 5.1. Aggregate confusion matrix for context aware part of the system under the within-person protocol can be seen on Table 5.2.

Table 5.1: Classification accuracy results on within-person and cross-person protocol.

| **Activity Label** | **Accuracy on Within-Person** | **Accuracy on Cross-Person** |
|---|---|---|
| Lying | 97.7% | 95.8% |
| Running | 88.8% | 77.7% |
| Sitting | 100% | 97.7% |
| Standing | 97.7% | 97.7% |
| Stair Down | 82.2% | 77.7% |
| Stair Up | 86.6% | 80.0% |
| Walking | 91.1% | 84.4% |
| Overall | 92.0% | 87.6% |

Table 5.2: Aggregate confusion matrix for context aware part of the system under the within-person protocol.

| **a** | **b** | **c** | **d** | **e** | **f** | **g** | **classified as** |
|---|---|---|---|---|---|---|---|
| 44 | 00 | 00 | 01 | 00 | 00 | 00 | a = lying |
| 00 | 45 | 00 | 00 | 02 | 03 | 00 | b = running |
| 00 | 00 | 45 | 00 | 00 | 00 | 00 | c = sitting |
| 00 | 00 | 01 | 45 | 00 | 00 | 00 | d = standing |
| 00 | 00 | 00 | 00 | 37 | 07 | 01 | e = stair down |
| 00 | 00 | 00 | 00 | 03 | 39 | 03 | f = stair up |
| 00 | 00 | 00 | 00 | 00 | 04 | 41 | g = walking |

Under the second protocol, cross-person protocol, we built models based only on other people. For each of the nine people, we trained the classifier using data from the first session of the other eight people, and tested on person's second session data. The mean accuracy under the second protocol is 87.6%. The classification accuracy results on cross-person protocol can be seen on Table 5.1.

Aggregate confusion matrix for context aware part of the system under the cross-person protocol can be seen in Table 5.3.

Table 5.3: Aggregate confusion matrix for context aware part of the system under the cross-person protocol.

| **a** | **b** | **c** | **d** | **e** | **f** | **g** | **classified as** |
|---|---|---|---|---|---|---|---|
| 43 | 00 | 00 | 02 | 00 | 00 | 00 | a = lying |
| 00 | 36 | 00 | 00 | 04 | 05 | 00 | b = running |
| 00 | 00 | 44 | 01 | 00 | 00 | 00 | c = sitting |
| 00 | 00 | 01 | 44 | 00 | 00 | 00 | d = standing |
| 00 | 00 | 00 | 00 | 35 | 08 | 02 | e = stair down |
| 00 | 00 | 00 | 00 | 04 | 36 | 05 | f = stair up |
| 00 | 00 | 00 | 00 | 02 | 05 | 38 | g = walking |

The results show that within-person model achieved a mean classification accuracy of 92.0%. This accuracy means that highly accurate classification is possible with only a small amount of training data from the subjects. Cross-person model had a mean accuracy of 87.6%. This shows that accurate activity classification could be achieved without the need to collect training data from new users.

## 5.2 Gesture Recognition Experiment

We have also performed an experiment to test the accuracy of the recognition. In this experiment, the subjects were asked to perform gestures when they are walking, running, stairing up, stairing down, lying, standing and sitting.

### Subjects

For the gesture recognition task, 9 subjects: 6 males and 3 females with a mean age of 24.8 were participated in the experiment. The subjects were among the voluntary graduate and undergraduate students who have computer science background. They were not informed about the purpose of the experiment.

### Procedure

For this experiment, subjects were asked to participate in seven consecutive test sessions for two days. In day one, system was running with context aware filtering and in the second day system was running with a fixed filtering. In fixed filtering the values of threshold, cutoff frequency, and the intensity of debouncing filter were the average values of the context aware filtering. The parameters of fixed filtering can be seen on Table 5.4. In each session they were asked to perform all the listed acceleration gestures on Table 5.5 for five times while they are reading text on the screen. Before all the sessions, subjects were given free time to practice gestures. In the first session, subjects were performed gestures while they were sitting. On the other sessions, while they were standing, lying, stairing up, stairing down, running, and walking. Each session took approximately 4 minutes. In each session, each subject perform 19 gesture for 5 times. There are seven sessions, and 9 subjects so 5985 performed gesture is examined in this test.

Table 5.4: Paramaters of the fixed filter.

| **Parameter** | **Value** |
|---|---|
| Threshold Value(G) | 0.25 |
| Lowpass Filter Cutoff Frequency(G) | 1.35 |
| $\alpha - \beta$ Range | -0.25 to 0.25 |
| Intensity of Debouncing Filter | 0.22 |
| Highpass Filer Cutoff Frequency | 0.20 |

Table 5.5: Tested gestures.

| **Gesture Type** | **Gesture Name** |
|---|---|
| Joystick | JRight |
| | JLeft |
| | JUp |
| | JDown |
| Move | GSRight |
| | GSLeft |
| | GSUp |
| | GSDown |
| Page | GSPageRight |
| | GSPageLeft |
| | GSPageUp |
| | GSPageDown |
| Global | GSEnter |
| | GSCopy |
| | GSPaste |
| | GSExit |
| | GSSquare |
| | GSCcSquare |
| | GSIsfront |

## Results and Discussion

The recognition accuracy for each activity label is shown on Table 5.6 and 5.7. These results show a high fidelity recognition rate of our gesture recognition enhancement system with an average recognition rate of 79.81%. The system increases the accuracy of gesture recognition between 8% and 32% according to the activity. These results also show that the system has the lowest recognition rate of 71.58% (32% higher than fixed filtering) in running, and system has the highest recognition rate of 87.02% (8% higher than fixed filtering) in standing.

The enhancement system has a recognition rate of 88% while the user is standing and 84% while sitting. The recognition rates in sitting and standing is higher than the other activities. Therefore, the system should be used when the user is sitting or standing. The system has the lowest recognition rate (71%) while running. This recognition rate means that gesture based interaction is not very accurate while the user is running, walking, stairing up, or stairing down.

The recognition enhancement system has better results on primitive acceleration gesture type (98%) and move gesture type (90%). This means that primitive and move gesture type are very accurate to use in everyday life. Page, global and other gesture types have a recognition accuracy of 73%. This accuracy rate means these gestures are not suitable to use in applications.

Table 5.6: The recognition accuracy with context aware filtering for activity labels.

| | **Recognition Accuracy (%)** | | | | |
|---|---|---|---|---|---|
| **Activity Labels** | **Joystick** | **Move** | **Page** | **Global** | **Average** |
| Walking | 96.67 | 89.44 | 85.56 | 59.05 | 78.95 |
| Standing | 100.0 | 96.67 | 93.89 | 70.16 | 87.02 |
| Stair Up | 97.22 | 86.11 | 81.67 | 58.10 | 77.19 |
| Stair Down | 97.78 | 84.44 | 81.11 | 57.46 | 76.61 |
| Running | 93.33 | 77.78 | 73.33 | 54.60 | 71.58 |
| Sitting | 99.44 | 97.22 | 94.44 | 63.17 | 84.56 |
| Lying | 97.22 | 93.89 | 92.22 | 62.86 | 82.81 |
| Overall | 97.38 | 89.36 | 86.31 | 60.75 | 79.81 |

Table 5.7: The recognition accuracy with fixed filtering for activity labels.

| | **Recognition Accuracy (%)** | | | | |
|---|---|---|---|---|---|
| **Activity Labels** | **Joystick** | **Move** | **Page** | **Global** | **Average** |
| Walking | 79.44 | 72.22 | 63.33 | 42.86 | 64.46 |
| Standing | 90.56 | 87.22 | 84.44 | 60.63 | 80.71 |
| Stair Up | 80.56 | 71.11 | 65.00 | 43.49 | 65.04 |
| Stair Down | 79.44 | 66.67 | 65.00 | 42.22 | 63.33 |
| Running | 71.11 | 58.33 | 51.67 | 36.19 | 54.32 |
| Sitting | 92.22 | 89.44 | 86.67 | 55.87 | 81.05 |
| Lying | 85.00 | 82.78 | 80.56 | 49.52 | 74.46 |
| Overall | 82.61 | 75.39 | 70.95 | 47.25 | 69.05 |

When subjects are standing, there is less noise data and subjects are more comfortable when they are moving their hands so recognition rate of the system is the highest. When subjects are running, there is more noisy data because of the running movements and system has the lowest recognition rate. Stair up and stair down movements has nearly same recognition rate because these two activity labels are very similar according to acceleration data. From our experiment, it is clear that our system performs very well for gesture recognition, recognizing them at more than 79% accuracy overall. Another finding is that our enhancement system performs well even when user is moving.

When subject are performing the listed gestures, the system was monitored to analyse the system performance. The system consists of three main parts, which are context awareness, filtering, and gesture recognition. The time that is spent during the gesture recognition can be seen on Table 5.8.

Table 5.8: The system performance (in milliseconds) during gesture recognition in activities standing(a), sitting(b), lying(c), stairing down(d), stairing up(e), running(f), and walking(g).

| **Part of the system** | **a** | **b** | **c** | **d** | **e** | **f** | **g** | **Average** |
|---|---|---|---|---|---|---|---|---|
| Context Awareness | 160 | 165 | 170 | 180 | 190 | 200 | 195 | 180 |
| Filtering | 3 | 4 | 5 | 5 | 6 | 7 | 5 | 5 |
| Gesture Recognition | 18 | 20 | 17 | 20 | 20 | 22 | 23 | 20 |

## 5.3 Subjective Experiment

We have also performed a subjective experiment to evaluate the usage of the system. In this experiment, the subjects were asked to perform all the gestures While they were walking, running, etc. and answer the survey questions.

**Subjects**

For the subjective experiment, 7 subjects: 4 males and 3 females with a mean age of 23.7 participated in the experiment. The subjects were among the voluntary graduate and undergraduate students who have computer science background.

**Procedure**

For this experiment, subjects were asked to participate in seven consecutive test sessions. In each session they were asked to perform all the listed acceleration gestures on Table 5.5. In the first session, subjects were performed gestures while they were sitting. In the other sessions, while they were standing, lying, stairing up, stairing down, running, and walking. Each session took approximately 4 minutes. After the session, they were asked the following questions:

- How friendly is the interaction method? (1: least 5: most)

- Which gesture type(s) would you prefer to use in a device? (Joystick, move, page, global or other gesture type)
- In which activities do you prefer to use the system? (Walking, standing, running, stair up, stair down, lying or sitting)
- In which activities do you prefer to use the system least? (Walking, standing, running, stair up, stair down, lying and sitting)

**Results and Discussion**

According to the users' answers to the questions stated above, the users found the sensor-based interface 80% user-friendly. Six of seven users prefer to use joystick and move gestures in an application. One user prefers joystick, move and page gestures in an application. All the users prefer to use the system when they are sitting or standing. Five users did not like to use the system when they are running. Two users did not like to use the system when they are running, stairing up and stairing down.

Six of seven users prefer to use primitive and move gesture types because the recognition enhancement system has better recognition results on primitive (98% accuracy) and move (90% accuracy) gesture types. This means better recognition accuracy enhance usage of the device. Therefore, the users prefer to use accurate gestures. All the users prefer to use system while they are sitting or standing, because the system has a recognition rate of 88% while the user is standing and 84% while sitting. The recognition rates in sitting and standing are higher than other activities. Five users did not like to use the system while they were running. This is because the system has the lowest recognition rate of 71% while running.

## 5.4 Applications

In this part we present two applications that we integrated the gesture recognition accuracy enhancement system to show example usage of acceleration sensor based interaction technique.

**Photo Album Application**

Displaying a large photo on a small screen has been a problem on handheld devices. Small screen makes the usage of scroll bars difficult. We propose to use accelerometer based gesture to handle the inputs for scrolling events. Therefore, we have implemented a photo album application. Controlling this application is intuitive. Gravity forces objects to slide towards to the centre of the earth, this is a nature phenomenon. If a table has an angle to the right, objects on the table slide to the right due to the gravity force. In this application when your device has an angle to the right (primitive gesture type) photo moves to the right like a real object.

We specifically focus on the application for navigating over several photo thumbnails in a photo album and browsing the photo selected. Screen shots of the photo album application can be seen in Figure 5.1, Figure 5.2. This application provides a showcase of modalities provided by accelerometer based gestures. It aims to enable the users to view the photos taken by high resolution cameras in their original dimensions. The same approach can easily be mapped to other similar applications such as map viewing.

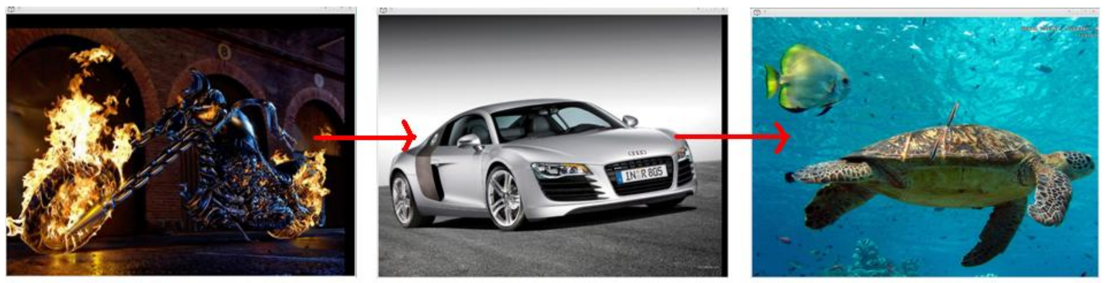


Figure 5.1: Photo album application is in thumbnail mode.

Photo album application has two modes, one is to see thumbnail of the photos (thumbnail mode) and the other one is to see the actual photo (full-view mode). In thumbnail mode, GSRight, GSLeft gestures are used to navigate over photos. GSEnter is used to change the mode to the full-view mode. GSExit is used to close the application. In full-view mode, JRight, JLeft, JUp and JDown is used to navigate over photo. Gsexit is to use to change the mode to thumbnail mode.

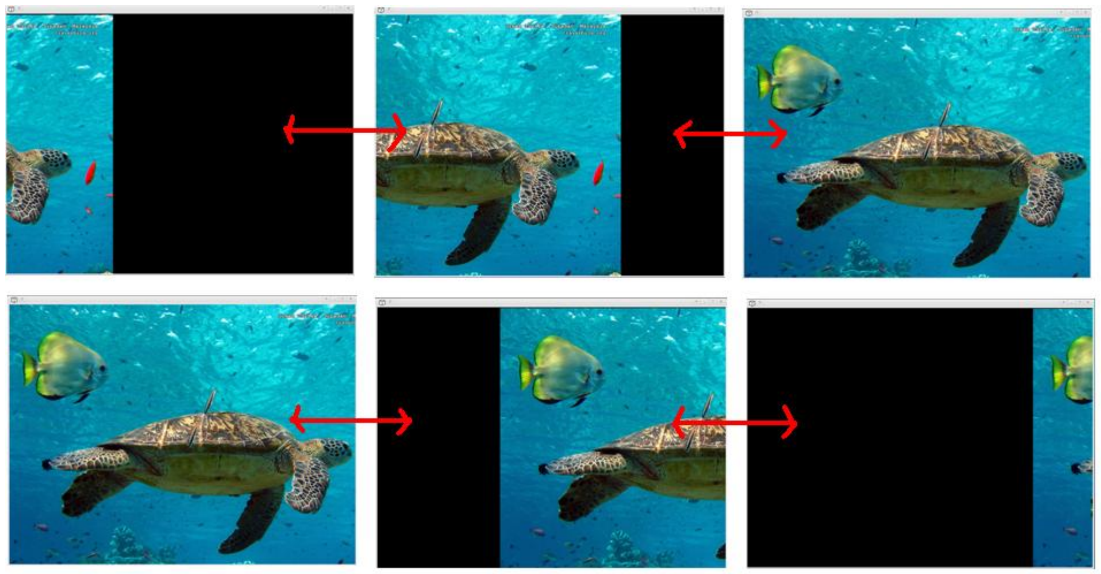


Figure 5.2: Photo album application is in full-view mode.From left to right, image is moving right.

### Media Browser Application

When defining gestures, we have tried to find more realistic and general purpose gestures that can be mapped to other applications as well. As an illustration, the gestures can be mapped to a media browser application. Instead of viewing thumbnails of photos, users can see different album covers. A specific album can be selected just like photo selection and the user can revert to album covers like reverting to thumbnails of photos. Finally, the gestures for scrolling a photo can be used navigate over songs in the selected album. This simple example shows that the proposed solution can be generalized to developing various applications for mobile devices.

In this application, the user can navigate over media files such as music files and photos. In the main screen of the application the user can select music files or photos with GSLeft and GSRight gestures. In photo mode of the application, user can navigate over photos with GSUp and GSDown gestures and navigate over photo album with GSPageLeft, GSPageRight, GSPageUp and GSPageDown gestures. GSExit is to used to exit to the main screen and gsenter is used to select the media file. Media browser application screen shot can be seen in Figure 5.3.

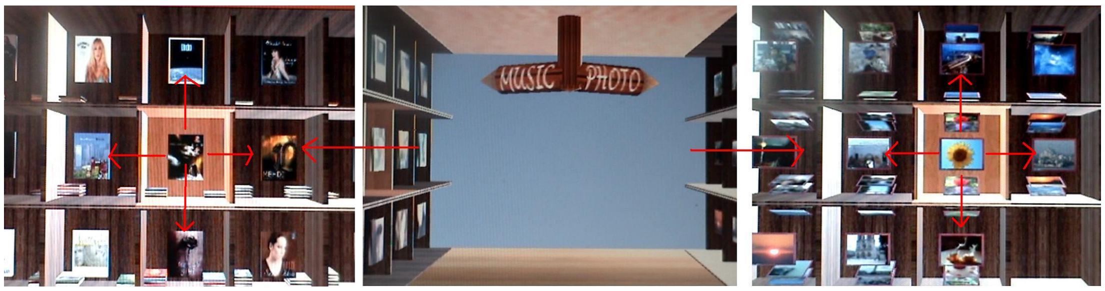


Figure 5.3: Media browser application. From left to the right, music album view, main screen and photo album view.

# Chapter 6

# Conclusion and Future Work

In this work, we proposed a framework that enhances the gesture recognition accuracy in a given content. For this purpose, we have developed a system that automatically decides on the suitable signal processing techniques for acceleration sensor data for a given context of the user. First system recognizes the context of the user using pattern recognition algorithm. Then, system automatically chooses signal filtering techniques for recognized context, and recognizes gestures. Gestures are also standardized for better usage. In this automatic gesture recognition accuracy enhancement framework, we consider several factors: context of the user, and signal type of the acceleration data.

We tested our system by the help of objective experimental studies. We had two experiments to test the accuracy of the context aware part and to test the accuracy of gesture recognition. To test the context aware part of the system we used within-person model and cross-person model. According to the results of the experiments, within-person model achieved a mean classification accuracy of 92.0%. This accuracy means that highly accurate classification is possible with only a small amount of training data from the subjects. Cross-person model had a mean accuracy of 87.6%. This shows that accurate activity classification could be achieved without the need to collect training data from new users. To test the accuracy of gesture recognition, we have examined nearly 6000 acceleration gestures. The result of the test shows a high fidelity recognition rate of the gesture recognition enhancement system with an average recognition rate of 79.81%. These results also show that system has the lowest recognition rate of 71.58% in running, and system has the highest recognition rate of 87.02% in standing activity.

We have designed 17 gestures for accelerometer sensor based interaction. The average recognition rate of the system is 80%. However, the recognition enhancement system has better results on primitive acceleration gesture type (98%) and move gesture type (90%). This means that primitive and move gesture type are very accurate to use in everyday life. Page, global and other gesture types have a recognition accuracy of 73%. This accuracy rate means these gestures are not suitable to use in applications.

We have tested the enhancement system in seven activity conditions. These are standing, sitting, lying, walking, stairing up, stairing down and running. The average recognition rate of the system is 80%. The enhancement system has a recognition rate of 88% while the user is standing and 84% while sitting. The recognition rates in sitting and standing is higher than the other activities. Therefore, the system should be used when the user is sitting or standing. The system has the lowest recognition rate (71%) while running. This recognition rate means that gesture based interaction is not very accurate while the user is running, walking, stairing up, or stairing down.

One possible future direction for our system is to implement more pattern recognition algorithms for context aware part of the system such as neural networks, k-nearest neighbor and hidden Markov model. Moreover, the signal processing part should be extended and more signal filtering methods should be implemented. Another idea for the future work is to train the classifier with more labels such as cycling and travelling by car and analyze the accuracy of the system under these activities. Furthermore, the current system is designed to enhance only gesture recognition which means that context information is used only for signal processing of acceleration data. Hence, it can be extended for user interface adaptation for different contexts to increase viewability of displayed information.

# Bibliography

[1] G. Abowd, C. Atkeson, J. Hong, S. Long, R. Kooper, and M. Pinkerton. Cyberguide: A mobile context aware tour guide. *Wireless Networks*, 3(5):421–433, October 1997.

[2] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles. Towards a better understanding of context and context-awareness. In *HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, pages 304–307, London, UK, 1999. Springer-Verlag.

[3] P. Accelerometer. http://www.phidgets.com/products.php?productid=1059, 2010.

[4] S. Agrawal, I. Constandache, S. Gaonkar, and R. R. Choudhury. Phonepoint pen: using mobile phones to write in air. In *MobiHeld '09: Proceedings of the 1st ACM workshop on Networking, systems, and applications for mobile handhelds*, pages 1–6, New York, NY, USA, 2009. ACM.

[5] K. Aminian, P. Robert, E. Buchser, B. Rutschmann, D. Hayoz, and M. Depairon. Physical activity monitoring based on accelerometry: validation and comparison with video observation. *Medical and Biological Engineering and Computing*, 37(1):304–308, January 1999.

[6] A. Asthana, M. Crauatts, and P. Krzyzanowski. An indoor wireless system for personalized shopping assistance. In *WMCSA '94: Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications*, pages 69–74, Washington, DC, USA, 1994. IEEE Computer Society.

[7] L. Bao and S. S. Intille. Activity recognition from user-annotated acceleration data. *Pervasive Computing*, pages 1–17, 2004.

[8] J. F. Bartlett. Rock 'n' scroll is here to stay. *IEEE Comput. Graph. Appl.*, 20(3):40–45, 2000.

[9] B. B. Bederson. Audio augmented reality: a prototype automated tour guide. In *CHI '95: Conference companion on Human factors in computing systems*, pages 210–211, New York, NY, USA, 1995. ACM.

[10] F. Bennett, T. Richardson, and A. Harter. Teleporting - making applications mobile. In *WMCSA '94: Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications*, pages 82–84, Washington, DC, USA, 1994. IEEE Computer Society.

[11] S. Bhandari and Y.-K. Lim. Exploring gestural mode of interaction with mobile phones. In *CHI '08: CHI '08 extended abstracts on Human factors in computing systems*, pages 2979–2984, New York, NY, USA, 2008. ACM.

[12] P. Brown. Triggering information by context. *Personal and Ubiquitous Computing*, 2:18–27, 1998. 10.1007/BF01581843.

[13] P. J. Brown, J. D. Bovey, and X. Chen. Context-aware applications: from the laboratory to the marketplace. *Personal Communications, IEEE [see also IEEE Wireless Communications]*, 4(5):58–64, 1997.

[14] F. Chehimi and P. Coulton. Motion controlled mobile 3d multiplayer gaming. In *ACE '08: Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*, pages 267–270, New York, NY, USA, 2008. ACM.

[15] G. Chen and D. Kotz. A survey of context-aware mobile computing research. Technical report, Dartmouth College, Hanover, NH, USA, 2000.

[16] W. Chen, D. Wei, S. Ding, M. Cohen, H. Wang, S. Tokinoya, and N. Takeda. A scalable mobile phone-based system for multiple vital signs monitoring and healthcare. *International Journal of Pervasive Computing and Communications*, 1:157 – 163, 2005.

[17] S.-J. Cho, C. Choi, Y. Sung, K. Lee, Y.-B. Kim, and R. Murray-Smith. Dynamics of tilt-based browsing on mobile devices. In *CHI '07: CHI '07 extended abstracts on Human factors in computing systems*, pages 1947–1952, New York, NY, USA, 2007. ACM.

[18] S.-J. Cho, R. Murray-Smith, and Y.-B. Kim. Multi-context photo browsing on mobile devices based on tilt dynamics. In *MobileHCI '07: Proceedings of the 9th international conference on Human computer interaction with mobile devices and services*, pages 190–197, New York, NY, USA, 2007. ACM.

[19] N. Davies, K. Cheverst, K. Mitchell, and A. Friday. 'caches in the air': Disseminating tourist information in the guide system. In *WMCSA '99: Proceedings of the Second IEEE Workshop on Mobile Computer Systems and Applications*, page 11, Washington, DC, USA, 1999. IEEE Computer Society.

[20] F. Decle and M. Hachet. A study of direct versus planned 3d camera manipulation on touch-based mobile phones. In *MobileHCI '09: Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services*, pages 1–5, New York, NY, USA, 2009. ACM.

[21] R. W. Devaul, B. Clarkson, and A. S. Pentland. The memory glasses: Towards a wearable, context aware, situation-appropriate reminder system. MIT Media Laboratory, 2000.

[22] A. K. Dey, D. Salber, G. D. Abowd, and M. Futakawa. The conference assistant: Combining context-awareness with wearable computing. In *ISWC '99: Proceedings of the 3rd IEEE International Symposium on Wearable Computers*, pages 21+, Washington, DC, USA, 1999. IEEE Computer Society.

[23] C. Dictinonary. http://dictionary.cambridge.org/, 2010.

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

[25] P. Eslambolchilar and R. Murray-Smith. Tilt-based automatic zooming and scaling in mobile devices on a state-space implementation. *Mobile Human-Computer Interaction on MobileHCI 2004*, pages 120–131, 2004.

[26] G. Essl and M. Rohs. Shamus - a sensor-based integrated mobile phone instrument. In *IN PROCEEDINGS OF THE INTERNATIONAL COMPUTER MUSIC CONFERENCE (ICMC)*, pages 27–31, 2007.

[27] G. W. Fitzmaurice. Situated information spaces and spatially aware palmtop computers. *Commun. ACM*, 36(7):39–49, 1993.

[28] G. W. Fitzmaurice, S. Zhai, and M. H. Chignell. Virtual reality for palmtop computers. *ACM Trans. Inf. Syst.*, 11(3):197–218, 1993.

[29] F. Foerster, M. Smeja, and J. Fahrenberg. Detection of posture and motion by accelerometry: a validation study in ambulatory monitoring. *Computers in Human Behavior*, 15(5):571–583, September 1999.

[30] A. R. Golding and N. Lesh. Indoor navigation using a diverse set of cheap, wearable sensors. In *ISWC '99: Proceedings of the 3rd IEEE International Symposium on Wearable Computers*, pages 29+, Washington, DC, USA, 1999. IEEE Computer Society.

[31] N. Györbíró, A. Fábián, and G. Hományi. An activity recognition system for mobile phones. *Mob. Netw. Appl.*, 14(1):82–91, 2009.

[32] B. L. Harrison, K. P. Fishkin, A. Gujar, C. Mochon, and R. Want. Squeeze me, hold me, tilt me! an exploration of manipulative user interfaces. In *CHI '98: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 17–24, New York, NY, USA, 1998. ACM Press/Addison-Wesley Publishing Co.

[33] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. The anatomy of a context-aware application. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 59–68, New York, NY, USA, 1999. ACM.

[34] K. Hinckley, J. Pierce, M. Sinclair, and E. Horvitz. Sensing techniques for mobile interaction. In *UIST '00: Proceedings of the 13th annual ACM symposium on User interface software and technology*, pages 91–100, New York, NY, USA, 2000. ACM.

[35] I. J. Jang and W. B. Park. Signal processing of the accelerometer for gesture awareness on handheld devices. *Robot and Human Interactive Communication, 2003. Proceedings. ROMAN 2003. The 12th IEEE International Workshop on*, pages 139–144, November 2003.

[36] G. H. Jin, S. B. Lee, and T. S. Lee. Context awareness of human motion states using accelerometer. *J. Med. Syst.*, 32(2):93–100, 2008.

[37] M. Joselli and E. Clua. grmobile: A framework for touch and accelerometer gesture recognition for mobile games. *Games and Digital Entertainment, Brazilian Symposium on*, 0:141–150, 2009.

[38] L. K, P. T, S. S, and V. A. Wireless motion bands. 2005.

[39] S. K. Kane. Context-enhanced interaction techniques for more accessible mobile phones. *SIGACCESS Access. Comput.*, (93):39–43, 2009.

[40] S. K. Kane, J. O. Wobbrock, and I. E. Smith. Getting off the treadmill: evaluating walking user interfaces for mobile devices in public spaces. In *MobileHCI '08: Proceedings of the 10th international conference on Human computer interaction with mobile devices and services*, pages 109–118, New York, NY, USA, 2008. ACM.

[41] N. Kern, B. Schiele, and A. Schmidt. Multi-sensor activity context detection for wearable computing. In *In Proc. EUSAI, LNCS*, volume 2875, pages 220–232, 2003.

[42] V. Lantz and R. Murray-Smith. Rhythmic interaction with a mobile device. In *NordiCHI '04: Proceedings of the third Nordic conference on Human-computer interaction*, pages 97–100, New York, NY, USA, 2004. ACM.

[43] S.-W. Lee and K. Mase. Activity and location recognition using wearable sensors. *IEEE Pervasive Computing*, 1(3):24–32, 2002.

[44] J. Liu, L. Zhong, J. Wickramasuriya, and V. Vasudevan. User evaluation of lightweight user authentication with a single tri-axis accelerometer. In

*MobileHCI '09: Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services*, pages 1–10, New York, NY, USA, 2009. ACM.

[45] J. Liu, L. Zhong, J. Wickramasuriya, and V. Vasudevan. uwave: Accelerometer-based personalized gesture recognition and its applications. *Pervasive Mob. Comput.*, 5(6):657–675, 2009.

[46] S. Long, R. Kooper, G. D. Abowd, and C. G. Atkeson. Rapid prototyping of mobile context-aware applications: the cyberguide case study. In *MobiCom '96: Proceedings of the 2nd annual international conference on Mobile computing and networking*, pages 97–107, New York, NY, USA, 1996. ACM Press.

[47] B. Lustrek, Mitja; Kaluza. Fall detection and activity recognition with machine learning. *The Free Library*, http://www.thefreelibrary.com/Falln

[48] J. Mantyjarvi, J. Himberg, and T. Seppanen. Recognizing human motion with multiple acceleration sensors. volume 2, pages 747–752 vol.2, 2001.

[49] S. Marinkovic, R. Puppo, R. L. C. Pan, and E. Popovici. Implementation of a secure fall detection system for body area networks. pages 38–47, 2010.

[50] N. Marmasse and C. Schmandt. Location-aware information delivery with commotion. *Handheld and Ubiquitous Computing*, pages 361–370, 2000.

[51] S. A. Mntyjrvi J, Alahuhta P. Wearable sensing and disease monitoring in home environment. *Workshop on ambient intelligence technologies for wellBeing at New York*, pages 45–51, 2004.

[52] M. A. Muñoz, V. M. Gonzalez, M. Rodríguez, and J. Favela. Supporting context-aware collaboration in a hospital: An ethnographic informed design. pages 330–344. 2003.

[53] J. Nielsen. *Usability Engineering*. Morgan Kaufmann, 1st edition, September 1993.

[54] E. ONeill, M. Kaenampornpan, V. Kostakos, A. Warr, and D. Woodgate. Can we do without guis? gesture and speech interaction with a patient information system. *Personal Ubiquitous Comput.*, 10(5):269–283, 2006.

[55] R. Oppermann and M. Specht. A context-sensitive nomadic exhibition guide. In *HUC '00: Proceedings of the 2nd international symposium on Handheld and Ubiquitous Computing*, pages 127–142, London, UK, 2000. Springer-Verlag.

[56] J. Pascoe. Adding generic contextual capabilities to wearable computers. pages 123–132, Los Alamitos, CA, USA, 1998. IEEE Computer Society.

[57] J. Pascoe, D. Morse, and N. Ryan. Developing personal technology for the field. *Personal and Ubiquitous Computing*, 2(1):28–36, March 1998.

[58] T.-L. Pham, G. Schneider, and S. Goose. Exploiting location-based composite devices to support and facilitate situated ubiquitous computing. In *HUC '00: Proceedings of the 2nd international symposium on Handheld and Ubiquitous Computing*, pages 143–156, London, UK, 2000. Springer-Verlag.

[59] Z. Prekopcsák, P. Halácsy, and C. Gáspár-Papanek. Design and development of an everyday hand gesture interface. In *MobileHCI '08: Proceedings of the 10th international conference on Human computer interaction with mobile devices and services*, pages 479–480, New York, NY, USA, 2008. ACM.

[60] C. Randell and H. Muller. Context awareness by analyzing accelerometer data. In *ISWC '00: Proceedings of the 4th IEEE International Symposium on Wearable Computers*, page 175, Washington, DC, USA, 2000. IEEE Computer Society.

[61] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman. Activity recognition from accelerometer data. In *IAAI'05: Proceedings of the 17th conference on Innovative applications of artificial intelligence*, pages 1541–1546. AAAI Press, 2005.

[62] J. Rekimoto. Tilting operations for small screen interfaces. In *UIST '96: Proceedings of the 9th annual ACM symposium on User interface software and technology*, pages 167–168, New York, NY, USA, 1996. ACM.

[63] T. Saponas, J. Lester, J. Froehlich, and J. Fogarty, J.and Landay. ilearn on the iphone: Real-time human activity classification on commodity mobile phones. Technical report, W-CSE-08-04-02, 2008.

[64] B. Schilit, N. Adams, and R. Want. Context-aware computing applications. In *WMCSA '94: Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications*, pages 85–90, Washington, DC, USA, 1994. IEEE Computer Society.

[65] B. N. Schilit and M. M. Theimer. Disseminating active map information to mobile hosts. *Network, IEEE*, 8(5):22–32, 1994.

[66] A. Schmidt, K. A. Aidoo, A. Takaluoma, U. Tuomela, K. Van Laerhoven, and W. Van de Velde. Advanced interaction in context. In *HUC '99: Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, pages 89–101, London, UK, 1999. Springer-Verlag.

[67] D. Siewiorek, A. Smailagic, J. Furukawa, N. Moraveji, K. Reiger, and J. Shaffer. Sensay: A context-aware mobile phone. pages 248–249. IEEE Computer Society, 2003.

[68] D. Small and H. Ishii. Design of spatially aware graspable displays. In *CHI '97: CHI '97 extended abstracts on Human factors in computing systems*, pages 367–368, New York, NY, USA, 1997. ACM.

[69] M. Specht. Adaptive support for a mobile museum guide. In *Proceedings of Interactive Applications of Mobile Computing 98*, pages 24–25, 1998.

[70] M. Tamviruzzaman, S. I. Ahamed, C. S. Hasan, and C. O'brien. epet: when cellular phone learns to recognize its owner. In *SafeConfig '09: Proceedings of the 2nd ACM workshop on Assurable and usable security configuration*, pages 13–18, New York, NY, USA, 2009. ACM.

[71] K. Van Laerhoven and O. Cakmakci. What shall we teach our pants? In *The Fourth International Symposium on Wearable Computers.*, pages 77–83, New York, 2000.

[72] M. Van Wieringen and J. Eklund. Real-time signal processing of accelerometer data for wearable medical patient monitoring devices. *Conference proceedings . Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference, London, UK*, 8:2397–2400, 2008.

[73] G. M. Voelker and B. N. Bershad. Mobisaic: An information system for a mobile wireless computing environment. In *WMCSA '94: Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications*, pages 185–191, Washington, DC, USA, 1994. IEEE Computer Society.

[74] J. Wang, S. Zhai, and J. Canny. Camera phone based motion sensing: interaction techniques, applications and performance study. In *UIST '06: Proceedings of the 19th annual ACM symposium on User interface software and technology*, pages 101–110, New York, NY, USA, 2006. ACM Press.

[75] R. Want, A. Hopper, V. Falc ao, and J. Gibbons. The active badge location system. *ACM Trans. Inf. Syst.*, 10(1):91–102, 1992.

[76] R. Want, B. N. Schilit, N. I. Adams, R. Gold, K. Petersen, D. Goldberg, J. R. Ellis, and M. Weiser. An overview of the parctab ubiquitous computing experiment. *Personal Communications, IEEE [see also IEEE Wireless Communications]*, 2(6):28–43, 1995.

[77] R. Want, B. N. Schilit, N. I. Adams, R. Gold, K. Petersen, D. Goldberg, J. R. Ellis, and M. Weiser. *The Parctab Ubiquitous Computing Experiment*, volume 353, chapter Chapter 2, pages 45–101. Springer US, Boston, MA, 1996.

[78] J. O. Wobbrock, A. D. Wilson, and Y. Li. Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes. In *UIST '07: Proceedings of the 20th annual ACM symposium on User interface software and technology*, pages 159–168, New York, NY, USA, 2007. ACM.

[79] T. Yamabe and K. Takahashi. Experiments in mobile user interface adaptation for walking users. In *IPC '07: Proceedings of the The 2007 International*

*Conference on Intelligent Pervasive Computing*, pages 280–284, Washington, DC, USA, 2007. IEEE Computer Society.

[80] T. Yamabe, K. Takahashi, and T. Nakajima. Design issues and an empirical study in mobility oriented service development. In *MobMid '08: Proceedings of the 1st workshop on Mobile middleware*, pages 1–6, New York, NY, USA, 2008. ACM.

[81] H. Yan and T. Selker. Context-aware office assistant. In *IUI '00: Proceedings of the 5th international conference on Intelligent user interfaces*, pages 276–279, New York, NY, USA, 2000. ACM.

[82] K.-P. Yee. Peephole displays: pen interaction on spatially aware handheld computers. In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 1–8, New York, NY, USA, 2003. ACM.

[83] J. S. Yi, Y. S. Choi, J. A. Jacko, and A. Sears. Context awareness via a single device-attached accelerometer during mobile computing. In *MobileHCI '05: Proceedings of the 7th international conference on Human computer interaction with mobile devices & services*, pages 303–306, New York, NY, USA, 2005. ACM.

[84] S. Zhang, C. Yuan, and Y. Zhang. Self-defined gesture recognition on keyless handheld devices using mems 3d accelerometer. In *ICNC '08: Proceedings of the 2008 Fourth International Conference on Natural Computation*, pages 237–241, Washington, DC, USA, 2008. IEEE Computer Society.