INTERACTIONS WITH SMARTPHONES AND SMARTWATCHES: CONTEXT-AWARENESS, TEXT ENTRY INTERFACES, AND INPUT BEYOND TOUCH

Rajkumar Darbar

INTERACTIONS WITH SMARTPHONES AND SMARTWATCHES: CONTEXT-AWARENESS, TEXT ENTRY INTERFACES, AND INPUT BEYOND TOUCH

Thesis submitted to the
Indian Institute of Technology Kharagpur
for award of the degree

of

Master of Science (MS) - by Research

by

Rajkumar Darbar

Under the guidance of

Dr. Debasis Samanta

Computer Science and Engineering
Indian Institute of Technology Kharagpur
Kharagpur - 721 302, India
July 2016

CERTIFICATE OF APPROVAL

00/00/0000

Certified that the thesis entitled **Interactions with Smartphones and Smartwatches: Context-Awareness, Text Entry Interfaces, and Input Beyond Touch** submitted by **Rajkumar Darbar** to the Indian Institute of Technology, Kharagpur, for the award of the degree Master of Science has been accepted by the external examiners and that the student has successfully defended the thesis in the viva-voce examination held today.

(Member of DAC) (Member of DAC) (Member of DAC)

(Supervisor)

(Internal Examiner) (Chairman)

CERTIFICATE

This is to certify that the thesis entitled **Interactions with Smartphones and Smartwatches: Context-Awareness, Text Entry Interfaces, and Input Beyond Touch** submitted by **Rajkumar Darbar** to Indian Institute of Technology Kharagpur, is a record of bona fide research work under my supervision and I consider it worthy of consideration for the award of the degree of Master of Science (by Research) of the Institute.

Date: 20/07/2016

Dr. Debasis Samanta
Associate Professor
Computer Science and Engineering
Indian Institute of Technology Kharagpur
Kharagpur - 721 302, India

DECLARATION

I certify that

a. The work contained in the thesis is original and has been done by myself under the general supervision of my supervisor.

b. The work has not been submitted to any other Institute for any degree or diploma.

c. I have followed the guidelines provided by the Institute in writing the thesis.

d. I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

e. Whenever I have used materials (data, theoretical analysis, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references.

f. Whenever I have quoted written materials from other sources, I have put them under quotation marks and given due credit to the sources by citing them and giving required details in the references.

Rajkumar Darbar

Dedicated to My Parents

ACKNOWLEDGMENT

During this period of my graduate study there are many people whose guidance, support, encouragement and sacrifice has made me indebted for my whole life. I take this opportunity to express my sincere thanks and gratitude to all these people.

First and foremost I would like to express my deepest gratitude to my revered supervisor *Dr. Debasis Samanta* for his invaluable guidance and encouragement throughout my work. His constant motivation, support and infectious enthusiasm have guided me towards the successful completion of my work. My interactions with him have been of immense help in defining my research goals and in identifying ways to achieve them. His encouraging words have often pushed me to put in my best possible efforts. Above all, the complete belief that he has entrusted upon me and has instilled a great sense of confidence and purpose in my mind, which I am sure, will stand me in good stead throughout my career.

It gives me immense pleasure to thank the head of the department Prof. Rajib Mall for extending me all the possible facilities to carry out the research work. My sincere thanks to all of my departmental academic committee members Prof. P. Mitra, Prof. C. Mandal, and Dr. K. S. Rao, for their valuable suggestions during my research. I sincerely remember the support of my friends and office staffs. Finally, I am forever grateful to my parents and my brother who have always rooted for me.

Rajkumar Darbar

Abstract

Of late, mobile and wearable devices such as smartphones and smartwatches have created a rapid impact on almost every aspect of our social and working life due to their ubiquity, pervasiveness, and portability. Improvements in computing power and sensory capabilities of these devices not only aid in creating the opportunities to implement a much wider variety of applications but also enable the development of smarter and more efficient interactions.

The objective of this dissertation is to enhance the user interactions with smartphones as well as smartwatches. First, we have explored sensor-based mobile interactions in two directions: phone's context-aware mobile interaction and around device interaction for efficient text input. Then we have investigated how text input interface of a smartwatch can be further improved for efficient typing. Finally, we have attempted on the expansion of smartwatch touch interface from touch screen to side of the device using pressure sensors to overcome the limited interaction space (i.e. diminutive screen).

In the case of phone's context-aware mobile interaction, we attempt to recognize different phone's placements (like pant's pocket, on the table etc.) using its inbuilt sensors. We show that different applications can be configured dynamically based on the identified phone's position to enhance mobile interaction.

Next, we propose around device interaction (ADI) based text input mechanism for smartphones to overcome the challenges of virtual keyboard based typing, for example, long visual search time and fat-finger issue.

To enable a better text entry mechanism with smartwatches, we introduce two text entry approaches: ETAO keyboard and 3D typing. In ETAO keyboard, a user can select most frequent characters with a single tap and remaining characters, numbers and symbols with two taps. With 3D typing, users can write text just by drawing character gesture above the watch face. It is a touchless text input approach.

Last, this thesis describes the design, implementation and evaluation of pressure sensors based smartwatch interactions like panning, zooming, rotation, scrolling, etc.

Keywords: Mobile and Wearable Devices, Interactions, Context-Aware Computing, Text Entry Interfaces, Sensors.

Contents

List of Figures

List of Tables

List of Symbols and Abbreviations

List of Symbols

μ	Mean
Δ	Change in the value of accelerometer
σ	Standard deviation
$\leq$	Less than or equal to a value
$\geq$	Greater than or equal to a value
μT	Magnetic flux density

List of Abbreviations

P	Precision
R	Recall
WPM	Words per Minute
ANOVA	Analysis of Variance
CPM	Characters per Minute
WIMP	Windows, Icons, Menus, and Pointer
GUI	Graphical User Interface
IR	Infrared Sensor

IMU	Inertial Measurement Unit
EMG	Electromyography
IG	Information Gain
SVM	Support Vector Machines
FSR	Force Sensing Resistor

Chapter 1

Introduction

Of late, mobile devices like smartphones and wearable devices like smartwatches have become the most popular computing devices at the consumer level. Their usage trend is also increasing at a rapid speed. According to eMarketer statistics[1], there were around 1 billion smartphone users all over the world in 2012 and by 2014, it reached nearly 1.75 billion. It is expected that smartphone adoption will continue on a fast-paced trajectory through 2017. Similarly, NextMarket Insights[2] reported a sale of around 15 million smartwatches worldwide, and this number is expected to increase.

Now-a-days people carry their phones in pocket and wear the watch on their wrist almost round the clock and in all places due to the tremendous affordability, flexibility, accessibility, and portability of these devices. In fact, we may claim that smartphones and smartwatches have truely become ubiquitous in nature. These devices are not only a communication and instant notification management tool, but they are also used for location-based services [26] [27], social activities [28] [29] [30], health care services [31] and so on. There is no doubt that we are gradually moving toward Mark Weiser's envision [32] - *"The technology required for ubiquitous computing comes in three parts: cheap,*

[1] http://www.emarketer.com/article/smartphone-users-worldwide-will-total-175-billion-2014/1010536

[2] http://bgr.com/2013/09/27/smartwatch-sales-forecast-2020/

low-power computers that include equally convenient displays, software for ubiquitous applications and a network that ties them all together."

In 1984, the Motorola DynaTAC 8000x[1] (see Fig.1.1(a)) was the first portable cellular phone. In 1992, IBM came up with a more refined version of the phone and launched IBM Simon[2] ((see Fig.1.1(b))) which was the first touch screen mobile phone. It was able to make and receive calls, send faxes and emails, and more. IBM Simon is the first device that could really be referred to as a smartphone. Revolutionary change in smartphone market came in the year 2007 when Apple Inc. introduced the iPhone[3], one of the first smartphones to use a multi-touch interface and a large touchscreen for direct finger input instead of a stylus and keyboard. Then smartphone based on Android and Windows OS came into the market with many other features like capacitive-touch screen, virtual keyboard, etc.

On the other hand, smartwatch development kicked off in the early 2000s. IBM

[1]https://en.wikipedia.org/wiki/Motorola_DynaTAC
[2]https://en.wikipedia.org/wiki/IBM_Simon
[3]https://en.wikipedia.org/wiki/IPhone

(a) (b) (c)

Figure 1.1: (a) Motorola DynaTac 8000x Cellular Phone, 1984 (b) IBM Simon Touch-Screen Phone, 1992 (c) The IBM WatchPad, 2000

WatchPad[1] ((see Fig.1.1(c))), the first smartwatch, was released by IBM in 2000. This watch included 8MB of memory, 320 x 240 QVGA display, Linux 2.4, an accelerometer, vibration motor, a fingerprint sensor and Bluetooth. In 2010, a decade after IBM's proof-of-concept, Sony SmartWatch was the first commercially available modern smartwatch. Presently, Smartwatches have exploded in popularity with Samsung Gear S2, Moto 360, Pebble Watch, LG G-Watch, Apple Watch and so on [33] [34].

If we look at the evolution of smartphones and smartwatches, one thing is very common - the computing power of these devices increases steadily over the years and it reminds us Moore's law [35] - *"Over the history of computing hardware, the number of transistors in a dense integrated circuit doubles approximately every two years."* Improvements in CPU speed, memory capacity, screen resolution and sensory capabilities of the devices have profoundly affected the development of mobile and wearable user interfaces. These interfaces have shifted from the mainly button- and menu-based interaction styles to Natural User Interfaces (NUIs) which involve rich sensory input and output. For instance, user interfaces for smartphones have transformed from being mostly button-driven, as was the case just a few years ago, toward being mainly multi-touch based today. Likewise, in the case of smartwatches, the user interface is gradually moving toward rotating bezel based interaction[2] from the direct touch input. Moreover, the extension of the sensory capabilities of mobile and wearable devices, such as incorporating accelerometers, gyroscopes, magnetometers, light and distance sensors, not only aid in creating the opportunities to implement a much wider variety of applications but also enables the development of smarter and more effective interactions.

The broad objective of this thesis is to develop a number of novel interaction techniques and user interface concepts for smartphones and smartwatches. We observe that the development of mobile and wearable user interfaces is closely coupled with the evolution of their sensory capabilities. In this dissertation, we have tried to make better

[1] https://www.cs.cmu.edu/ 15-821/CDROM/PAPERS/narayanaswami2002a.pdf
[2] Samsung Gear S2 with rotating bezel control. http://www.samsung.com/global/galaxy/gear-s2/

use of the available sensing capabilities of these devices for improving user interactions. Moreover, we also provide suggestions on the types of sensor technologies that could be added to the future devices in order to enrich their input expressiveness.

The rest of this chapter is organized as follows. Section 1.1 describes the challenges for designing efficient mobile and wearable interactions. The scope and objective of our work are presented in Section 1.2. Section 1.3 discusses the contribution of the thesis and finally, Section 1.4 outlines the organization of the thesis.

1.1 Challenges for Designing Efficient Mobile and Wearable Interactions

The emerging market of smartphones and smartwatches provides a tremendous opportunity to introduce efficient mobile and wearable interactions to the marketplace. While developing a novel interaction technique, we should consider the different challenges which come from device perspective as well as from a user point of view. Several constraints related to an effective interaction design are as follows:

- **Location for storing a mobile phone:** Smartphones are often stored in pant's pocket, inside a bag, etc. [36] [37] and we have to remove them every time in order to access basic functionality like calling, messaging, changing music, etc. It requires a high level of cognitive and visual attention. Moreover, physically retrieving the device is socially disruptive and it also incurs a non-trivial time cost.

- **Small touch screen display:** The latest smartphones have large enough touch screen display which ranges from 4.8 inches to 5.5 inches. But, in the case of smartwatches display size is on an average 1.5 inch [11] [12] [13]. The diminutive screens of these devices mar the user experiences and suffer from visual occlusion and the fat finger problem.

- **Phone holding pattern:** In general, people use their both hands - one hand to hold the phone and another hand to touch the screen, for effective interaction [38] [39]. But, interactions become difficult in the one-handed situation.

- **Tactile feedback:** In touch interactions, users can feel tactile sensation through a little vibration while they touch the phone's display. But, in some situations, users perform touchless interactions. For example, during cooking, they can't touch phones with their messy hands. Providing tactile feedback while users gesture in mid-air will help them to overcome uncertainty about gesture performance. It is really a challenge to design a tactile feedback mechanism for above-device interaction with mobile phones [40] [41].

- **Power consumption:** Energy consumption is a major concern for mobile and wearable devices [42] [43]. Developing an energy efficient interaction technique at hardware and software levels is also a big challenge.

- **User's Context:** The smartphones and smartwatches are more intimate to users because they often carry or even wear these devices throughout their daily routine. Hence, user's context (i.e. location, ambient light condition, sitting/walking/running, surrounding noise level, etc.) is an important factor for efficient interaction design [44] [45].

1.2 Scope and Objectives

In this dissertation, we have aimed to expand interactions with smartphones as well as smartwatches keeping the challenges as mentioned above in mind. First, we have explored mobile interactions in two directions: phone's context-aware mobile interaction and around device interaction for efficient text input. Then we have investigated how text input interface of a smartwatch can be further improved for efficient typing. Finally, we

have attempted to extend the smartwatch interaction space beyond touchscreen input. The scopes of this work are as follows.

- **Phone's context-aware mobile interaction:** Smartphone became ubiquitous in nature and interaction with these devices greatly depends on current context. Presently, mobile devices know little about the context in which they operate. Hence, a user takes responsibility for managing phone's behavior to fit with the current environment. Performing these settings manually is an extra burden to users. To make the mobile interaction more effective, researchers are focusing on the development of context-aware mobile applications. There are two perspectives in context-aware applications: user's context and phone's context. In this dissertation, we are particularly interested in automatic detection of phone's context and how it can be deployed to enhance user interaction with mobile devices. For example, if the phone is on the table, then vibration may cause damage in the phone. In this case, only ringing at the time of an incoming call is enough to draw user attention. Most of the research in this area focus on the development of several software and hardware solutions to detect phone's context (i.e. phone is on the table, inside the backpack, etc.). Further, this area of research can be explored in the following directions: **(1)** designing an energy efficient phone's placement detection algorithm; **(2)** need user study to understand how this context-aware technique helps users to perform the better mobile interaction.

- **Around Device Interaction (ADI) for text entry on smartphones:** An important part of mobile interaction involves entering text. The traditional way of entering text on the smartphone is Qwerty soft keyboard that demands a long visual search time and suffers from fat finger problem. It is also hard to hit the keys while users in mobile scenarios and at the same time, letters are difficult to read, especially when eyesight-impaired [46]. In this dissertation, we are trying to develop magnetic field based around device interaction (ADI) technique for

text input on the smartphone. We want to explore it in the following directions: **(1)** explore ADI techniques to recognize English alphabets and punctuations; **(2)** design algorithm to detect start and end of character gesture automatically; **(3)** development of 3D haptic feedback solution.

- **Text input interfaces on smartwatches:** We are gradually moving toward wearable computing. Recently, smartwatches have gained a lot of public attention as one of the most popular wearable devices. It allows users to interact with several applications (messaging, email, calendar, maps) running on smartphones directly from their wrists, without having to look at their phones. Here, text input is also an essential part of smartwatch interaction and the scope of work in this area are as follows: **(1)** designing a full-fledged soft keyboard with well balance between error rate and typing speed; **(2)** moving towards touchless (i.e. 3D) text entry may be great help to the users as it does not need an accurate and conscious finger pointing on a keyboard.

- **Smartwatch interaction mechanisms beyond touchscreen input:** However, smartwatches are a promising new interactive platform, but their small size makes even basic actions cumbersome. In this scenario, we are trying to extend smartwatch interaction space beyond touchscreen input using minimum hardware instrumentation.

1.3 Contribution of the Thesis

This dissertation contributes a range of sensor-based novel interaction techniques designed for current and future mobile and wearable devices. Our aim is to enhance the input expressiveness of these devices. We have shown that the expressiveness of smartphone's and smartwatches input can be significantly increased using the devices inbuilt sensors or with the help of some external sensors. We have made contributions in three

promising areas: context-awareness, text input interfaces, and input beyond touch. The main contributions of this thesis are discussed as follows.

- In the case of phone's context-aware mobile interaction, we present *SurfaceSense*, a scalable phone's context sensing framework, to recognize 13 different phone's placements using phone's embedded accelerometer, gyroscope, magnetometer, microphone and proximity sensor. Our proposed technique primarily works in two steps. The first step identifies that there is a change in phone's placement. For example, a user is taking the phone placed on the table and putting it in his pant's pocket. Therefore, phone position has changed from on the table to inside the pant's pocket. The second step detects phone's current context.

  To understand the change in phone's placement, we proposed a simple threshold based algorithm on tri-axial accelerometer signal, and it is running continuously as a background service.

  When a change in phone's placement is detected, SurfaceSense triggers the phone's placement sensing mechanism. To sense phone's present surface, it vibrates for four seconds and during the vibration, accelerometer, and gyroscope record motion data, magnetometer records magnetic field strength, proximity sensor measures the presence of nearby objects at different distance levels, and microphone captures phone's vibration echoes. Once sensor data collection is completed, data are processed to identify those surfaces. The surface recognition procedure is basically a two-tier hierarchical classification approach. In the first and second level, a simple 'if-else' rule-based reasoning module is used on the basis of proximity and magnetometer sensor data pattern to categorized surfaces into four subgroups, that is, metal and non-metal inner surfaces, metal and non-metal outer surfaces. Then, extracted features from accelerometer and gyroscope sensor data and recorded vibration echoes are fed into four Random Forest (RF) classifiers to infer surfaces from each group. Note that, there is one RF classifier for each subgroup.

Finally, we develop several example applications to show that how the proposed framework can be used to enhance mobile interactions.

- Next, we propose *MagiText*, around device interaction based text input mechanism, which provides the text entry space beyond physical boundaries of a device. The key idea is to influence the magnetometer sensor by writing character gestures in front of the device using a properly shaped magnet taken in hand. Here, we consider two types of character gestures - Graffiti[1] and EdgeWrite [47]. The movement of this magnet changes the magnetic flux pattern, and it is sensed and registered by the magnetometer sensor. Then, we classify this flux pattern using support vector machine classifier and identify intended character.

- To enable a better text entry mechanism with smartwatches, we propose two text entry approaches: *ETAO keyboard* and *3D typing using hall effect sensors*.

  The ETAO keyboard enables faster and less erroneous text input on ultra-small interfaces of smartwatches. Using our proposed keyboard's user-interface (UI), a user can type most frequent characters (i.e. E, T, A, O, I, N, S, and R) with a single tap and remaining characters, numbers, and symbols with two taps. Here, we use 'tap' as the prime input method because it is really easy to perform while walking in a street. Moreover, this layout easily eliminates 'fat-finger problem' by providing keys with bigger buttons and provides well trade-off between typing speed and error rate.

  With 3D typing, users can write text just by drawing character gesture above the watch face. It is basically a touchless text input approach for smartwatches. To achieve this, we place four Hall sensors[2] in four corners of a watch and map each EdgeWrite letter [47] to four corners. To enter any alphanumeric character, users move their magnet mounted finger over hall effect sensors following the predefined

[1] http://en.wikipedia.org/wiki/Graffiti_(Palm_OS)
[2] https://en.wikipedia.org/wiki/Hall_effect_sensor

corner sequences. This text input technique has several advantages such as (a) it does not require any touch screen space, (b) it is free from occlusion problem, and (c) it does not demand any visual search also.

- To extend smartwatch's input space beyond its tiny touchscreen, we present *PressTact* which augments smartwatches with four Force Sensing Resistors (FSRs) - two sensors on the left side of a watch and another two on the right side. It enables users to input different levels of pressure (light press, medium press, and strong press) in discrete and continuous mode. In this work, we define a rich vocabulary of pressure events that can be mapped to many different actions in a variety of applications. For example, it can be used for bi-directional navigation (panning, zooming, scrolling, rotation, etc.) on smartwatches. Our preliminary user study shows that participants can input different pressure levels with an acceptable accuracy, and it is a promising input modality for future smartwatches.

1.4 Organization of the Thesis

The thesis contains seven chapters including this introductory chapter. This chapter gives a brief introduction to mobile and wearable interactions and discusses different challenges associated with it. Then, scope and objectives of the work are presented in this chapter. Next, we discuss the contribution of the thesis. It also introduces subjects which are described in details in the next chapters.

Chapter 2: Related Work

This chapter provides compendious reviews of existing works related to phone's context-aware mobile interaction, around device interaction for text entry on the smartphone, text input interfaces on smartwatches, and smartwatch interaction mechanisms beyond touchscreen input.

Chapter 3: Phone's Context Sensing Based Mobile Interaction

This chapter describes our approach to detect phone's context. It also highlights different mobile interactions developed on the basis of phone's context.

Chapter 4: Magnetic Field Based ADI for Text Entry in Smartphone

This chapter explains the details of around device magnetic interaction for 3D space text entry in the smartphone.

Chapter 5: Efficient Text Input on Smartwatches

This chapter describes the design and user study of our proposed smartwatch's text input mechanisms - ETAO keyboard and 3D typing hall effect sensors.

Chapter 6: Side Pressure-Based Input for Smartwatch Interaction

This chapter provides the details implementation and user study of the side pressure sensors based smartwatch interaction technique.

Chapter 7: Conclusion and Future Research

Finally, this chapter summarizes the thesis contributions and provides potential future research directions for further exploration.

Chapter 2

Related Work

In this chapter, we survey the existing work related to the interaction techniques planned in our thesis. We report our survey in four subsections. Section 2.1 represents related work on phone's context-aware mobile interaction. Section 2.2 discusses existing work on around device interaction for text entry technique. Section 2.3 highlights related work on smartwatch text input interfaces. Next, Section 2.4 presents existing work on smartwatch interaction mechanisms beyond touchscreen input. Finally, we summarize the all related work sections.

2.1 Phone's context-aware mobile interaction

Phone's context (e.g., whether phone is on the table, inside backpack etc.) sensing is an active research field. So far, various approaches have been proposed using phone's embedded sensors and sometimes, with the help of external hardware. Harrison et al. [48] identified proximity materials of mobile devices using the multi-spectral optical sensor. In their approach, different LEDs (i.e. infrared, red, green, blue and ultraviolet) were used to artificially illuminate the target material and photo-register measured the reflected light properties (i.e. wavelength). The experimental result showed 86.9% placement-detection accuracy for 27 different test placements. This approach consumes

less power (∼20 mA) and faster enough (took 5 sec). In [36], the authors built 'Phoneprioception' model with 85% accuracy using experience sampling method (ESM) to infer phone placements. They also demonstrated that reasonably accurate classification is possible using proximity sensor, light sensor, and multi-spectral sensor. Wahl et al. [49] presented RFID tag based phone placement inferring method. In their experiment, they placed RFID tags at pant, table, jacket and bag. Smartphone's built-in NFC reader automatically scans RFID tags when phone passes these sites and recognizes the places with an average accuracy of 80%.

The methods described in [48] [36] and [49], demand some external hardware setup. But, it would be much better and robust if cell-phone's inbuilt sensors (such as an accelerometer, gyroscope, microphone, proximity sensors, magnetometer etc.) can be used to infer phone surface. Keeping this in mind, Cho et al. [50] proposed 'VibePhone' where extracted Jigsaw and time histogram features from vibration generated acceleration readings are used as input to the SVM classifier that recognized six contract surfaces (i.e. sofas, plastic tables, wooden tables, hands, backpacks, and pants pockets) with 85% accuracy. In [51], Hwang et al. proposed 'VibroTactor', an easy and inexpensive solution, by analyzing smartphone's microphone captured acoustic signal generated when the mobile device vibrates. They derived several characteristics (such as peak count, peak intensity, peak frequency, and skewness) from spectrograms of vibration echoes on different placements. These features are fed to the RBF classifier that achieves the recognition rate of 91% in 12 different real-world placement sets. Kunze et al. [37] proposed a symbolic phone location method based on an active sampling of vibration generated motion data captured by the accelerometer and 'beep' sound signatures. They achieved recognition rates of up to 81% for 35 trained locations and 86% for 12 abstract locations. In [52], Yang et al. demonstrated a low-cost solution using smartphone embedded proximity (IR) and light sensor to detect 'in the pocket', 'in the bag', 'out of pocket' and 'out of the bag'. The average accuracy of their demo prototype is above 98% and it consumes

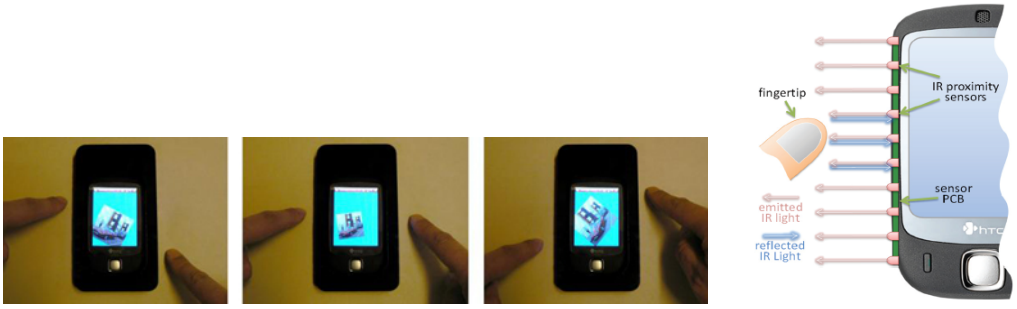
less than ∼6mW power for collecting sensor readings. Recently, Diaconita et al. [53] presented an acoustic probing based approach to detect 'in a backpack', 'on the desk', 'in user's hand' and 'in a user's pocket'. In their approach, mobile phone emits and records short bursts of inaudible audio signals while it is placed at above mentioned positions. The differences in signal attenuation reveal the nature of the material surrounding the mobile phones. They performed this experiment in various environments such as office, bus, train, outdoors etc. For identification purpose, they extracted MFCC, DMFCC and Band Energy (BE) features from the recorded audio signal. Finally, they achieved 97% and 96% accuracy using K-Nearest Neighbors and Random Forest, respectively.

2.2 Around Device Interaction for text entry with smartphones

3D space text entry technique expands the text input space beyond the physical boundaries of the device. For this purpose, researchers have used phone's inertial sensors such as the accelerometer, gyroscope, and magnetometer. For instance, in [1], Agrawal et al. proposed 'PhonePoint-Pen' where a user holds the phone like a pen and writes short messages in the air ((see Fig.2.1(a))). The acceleration due to hand gestures are translated into geometric strokes and compares the sequence of strokes against a grammar tree to identify the air written English alphabets. This process needs no training and recognizes characters with 83% accuracy. Deselaers et al. [2] presented 'GyroPen' that also allows users to hold the smartphone like a pen and write on any surface ((see Fig.2.1(b))). The angular trajectory of the phone's corner is reconstructed from the phone's embedded gyroscope sensor and it is used as input to the online handwriting recognition system to identify the desired English words. In the experiment, novice participants took 37 seconds to write a word, whereas experienced users were able to write it within 3-4 seconds with a character error rate of 18%. However, both types of users feel that holding

(a) (b)

Figure 2.1: (a) PhonePoint-Pen prototype [1] (b) GyroPen prototype [2].

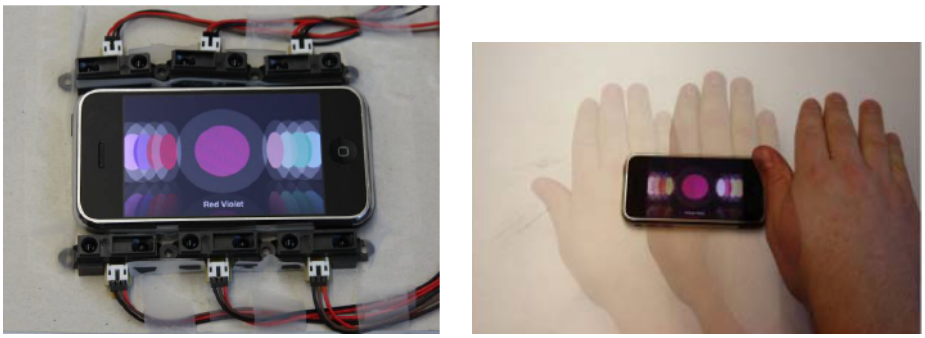


(a) (b)

Figure 2.2: (a) SideSight prototype [3] (b) The working principle behind SideSight.

the phone as a pen is a little bit awkward for writing long words. In [54], authors proposed a stepwise lower-bounded dynamic time warping (DTW) algorithm to recognize user-independent real-time 3D handwritten English alphabets using phone's built-in gyroscope sensor. This approach achieves 91.1% accuracy and it is computationally faster in terms of memory and CPU time.

The 3D space handwritten recognition methods presented in [1], [2] and [54] considered phone as a pen. But, in our work, we used a different framework called around device interaction that utilizes the physical space around the mobile device to provide richer input possibilities. This ADI commonly deals with different types of sensory inputs such as infrared distance sensor, camera, depth sensor, electric field, and magnetic field. For example, Butler et al. [3] proposed 'SideSight' ((see Fig.2.2(a)) & (b))) which uses infra-red (IR) proximity sensors to implement single and multi-touch gestures to



(a) (b)

Figure 2.3: (a) HoverFlow prototype [4] (b) User performs 'sweep-right hand palm' gesture above the mobile device.

Figure 2.4: Recognizing different in-air hand gestures using Phone's RGB camera [5].

the sides of a mobile device when the device is rested on a flat surface. Similarly, 'HoverFlow' [4] also recognizes coarse hand gesture above a mobile device using a set of IR distance sensors (see Fig.2.3(a) & (b)).

In [5], Song et al. developed a novel machine learning algorithm to extend the interaction space around mobile devices by detecting rich gestures performed behind

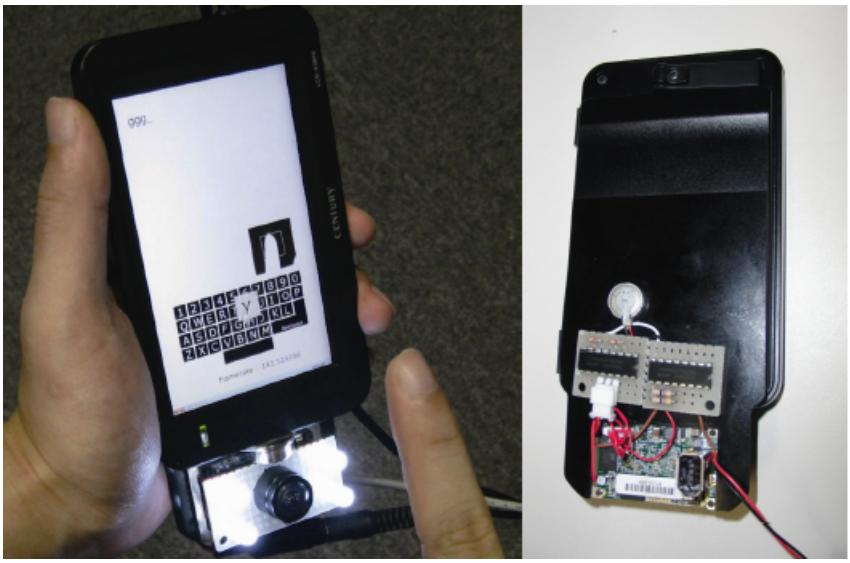


Figure 2.5: Real-time fingertip tracking based in-air typing interface for mobile devices [6].

Figure 2.6: PalmSpace prototype detects different palm poses using depth camera [7].

or in front of the screen (see Fig.2.4). Their algorithm takes input from phone's RGB camera in real-time and recognizes different hand gestures robustly (93% accuracy) in varying lighting conditions. Niikura et al. [6] proposed 3D typing interface by tracking fingertip using a high frame rate camera (see Fig.2.5).

In [7], Kratz et al. presented 'PalmSpace' which detects several single handed palm gestures using depth cameras on mobile devices (see Fig.2.6) and manipulates the orientation of a virtual object on mobile device screen just by mapping those gestures onto it. Goc et al. [8] developed electric field sensing based 3D interaction techniques on mobile

Figure 2.7: Electric field sensing based interactions with mobile devices [8].

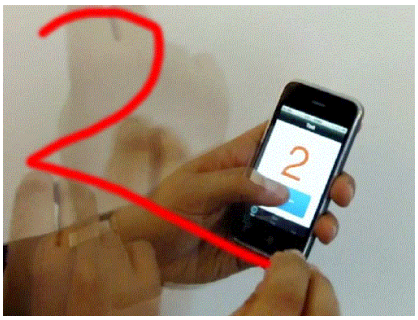


Figure 2.8: MagiWrite prototype [9].

devices (see Fig.2.7).

There are some touchless interaction techniques (such as 'MagiTact' [55], 'MagPen' [56], 'MagNail' [57], 'MagGetz' [58], 'Magnetic Appcessories' [59], 'MagCubes' [60], 'Magnetic Marionette' [61]) using phone's embedded magnetometer sensor. Among all these sensory inputs, magnetic field based ADI is much simpler as (1) its hardware (i.e. magnetometer sensor) is already available in the current mobile devices; (2) it consumes very less power; (3) it does not suffer from illumination variation and occlusion problems like camera based ADI; (4) it can pass through many materials i.e. it enables in-pocket interaction. Considering all these benefits, Ketabdar et al. [9] introduced 'MagiWrite', which supports 3D space digit (i.e. 0 - 9) entry in smartphones using magnetic field based around device interaction technique (see Fig.2.8).

2.3 Text input interfaces on smartwatches

In recent time, text entry on smartwatches is a prospering research area. Tiny touch screen of smartwatch suffers from visual occlusion and the fat finger problem. The fingers obscure on-screen contents and user interface elements during interaction, and as a result it hinders efficient text input on watches. To address these challenges, researchers have investigated various text input methods in last few years.

In [10], Oney et al. proposed 'ZoomBoard' (see Fig.2.9) that uses a miniaturized

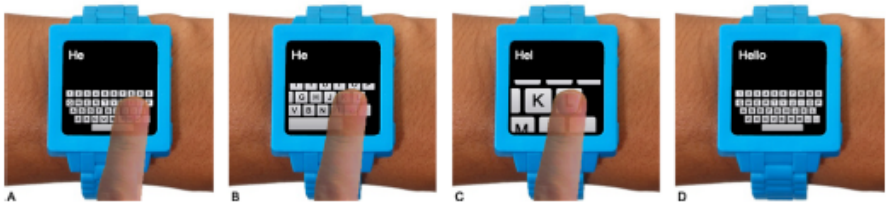


Figure 2.9: (a) Default ZoomBoard layout [10]. (b & c) User taps for iterative zooming. (d) The keyboard resets after entering the desired character.

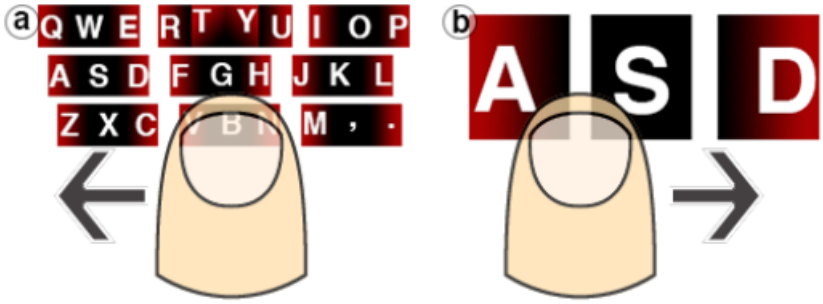
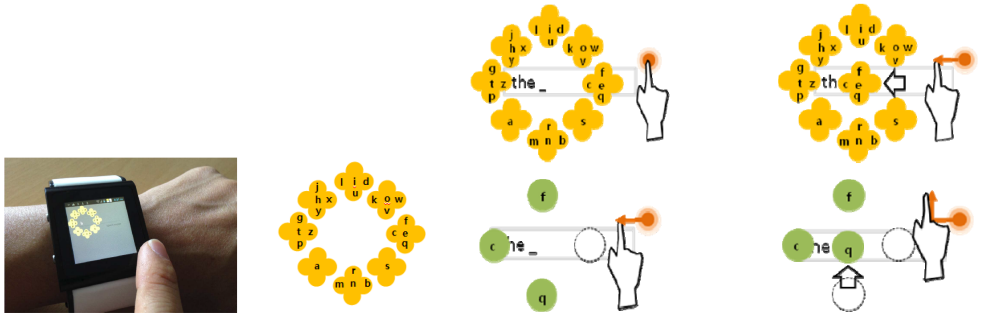


Figure 2.10: Swipeboard [11]: (a) User selects one region out of nine with the first swipe. (b) Using second swipe, user enters specific character. In the figure, swiping right types 'D'.

version of the conventional Qwerty keyboard. The user has to focus on a particular area of keyboard and then tap for zooming into that area. The user can also zoom in further depending upon the number of zoom levels set. Once the zooming is done the user selects the appropriate key by tapping. Although this mechanism seems favorable to the user because of the familiar layout, it still requires two or more careful taps to zoom and select a key. Text entry rates suffer because of these excessive tapping tasks.

The 'Swipeboard' [11] divides the traditional Qwerty keyboard into nine regions and to enter any character, user requires two swipes. Figure 2.10 represents the 'Swipeboard' prototype. Using first swipe, user specifies the desired character's region and the second swipe selects the particular character within that region.

In [12], Cho et al. developed 'DragKey' prototype (see Fig.2.11(a) & (b)) for text entry in wrist-worn watches with tiny touchscreen. It is a circular keyboard composed of 8 ambiguous keys arranged around the text cursor. At most five letters are assigned



(a) (b)

Figure 2.11: (a) DragKeys layout [12]. (b) The interaction sequence to enter 'q' with DragKeys.

(a) (b) (c)

Figure 2.12: The alphabetic ambiguous-keyboard running Sony Smartwatch 2: (a) concept, (b) design, and (c) implementation [13].

to each key. It allows a user to input letters using drag directions regardless of careful touched locations. A user needs lot of time to learn this layout. Furthermore, making continuous drag gestures is quite difficult in walking situations and it is also slower than tapping.

Dunlop et al. [13] proposed alphabetic ambiguous-keyboard for text entry and it is shown in Fig.2.12. They divided the watch screen into seven zones, that is, six big ambiguous keys (three at the top of the screen and three at the bottom) and a center zone for the input entry field. OpenAdaptxt [62] is used for entry disambiguation and input methods like tapping and few swipe gestures are used to change modes (alphabetical/numerical, lower/upper case, punctuation), complete a word or enter a space. Overall, it is good, but a user may face difficulties while trying to enter password and urls.

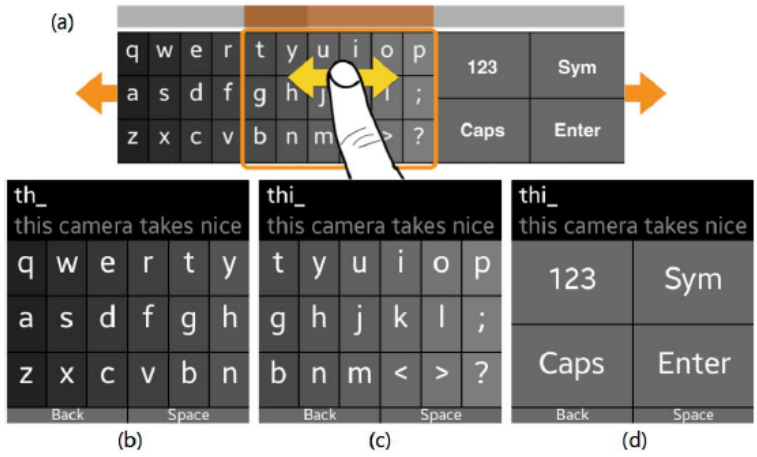


Figure 2.13: (a) The operational concept of the SplitBoard layout [14]. (b, c, & d) The three parts of the SplitBoard. When a user flicks from right to left on the part (b), the screen switches to (c).

Figure 2.14: UniWatch Keyboard: Typing using three strokes ('/', '(', '|') [15].

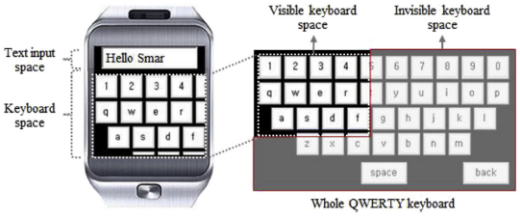


Figure 2.15: The conceptual drawing of the Virtual Sliding Qwerty (VSQ) [16].

Moreover, commercially available prediction based text input techniques like Minuum[1], TouchOne[2], Swipe[3], and Fleksy[4] also suffer from similar kind of problems.

Hong et al. [14] developed 'SplitBoard' (see Fig.2.13) which is a variation of the Qwerty keyboard. Here, Qwerty layout is split into a few layers. The user sees one layer of keys and has to swipe left or right to press keys present in other layers. It is intuitive

[1]http://minuum.com/
[2]http://www.touchone.net/
[3]http://www.swype.com
[4]http://fleksy.com

Figure 2.16: Using a touch-sensitive wristband for text entry on smartwatches [17]. The multi-tap layout is shown on the left and the linear keyboard is shown on the right.

to use as it doesn't require a steep learning curve. But, the key-size of SplitBoard is not large enough to avoid fat-finger problem.

Poirier et al. [15] designed 'UniWatch' (see Fig.2.14) derived from the UniGlyph [63] method and it supports text input on smartwatches using only three keys i.e. diagonal-shape key ('/'), loop-shape key ('(') and straight-shape key ('|').

In [16], J. M. Ckha proposed 'Virtual Sliding Qwerty' (VSQ) keyboard (see Fig.2.15) which utilizes a virtual qwerty layout and a 'Tap-N-Drag' method to move the qwerty keyboard until the target letter is shown on the screen.

The keyboards, discussed so far, require significant amount of space in the watch display. More recently, Funk et al. [17] explored a new text entry method for smartwatches using a touch sensitive wristband (see Fig.2.16). This technique does not need any screen space and thus watch's screen can be used for presenting actual content.

2.4 Smartwatch interactions extending touchscreen input

Touchscreen interaction has become a fundamental means of controlling smartwatches. However, the small form factor of a smartwatch limits the available interactive surface area. In recent past, researchers have proposed several approaches to address these challenges.

Figure 2.17: Interaction with smartwatch using 'NanoStylus' prototype [18].

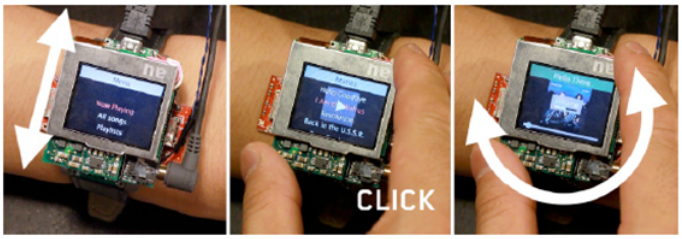


Figure 2.18: Smartwatch with hall-effect joystick sensor supports continuous 2D panning, twist, tilt and click gestures [19].

Baudisch P. and Chu G. [64] presented touch enabled backside of the device for occlusion free interaction. However, rear surface of a wristwatch is inaccessible to users. TouchSense prototype [65] expanded watch's touchscreen input bandwidth by augmenting different areas of human finger with an IMU sensor. Oakley et al. [66] developed beating gestures composed of a rapid pair of simultaneous or overlapping screen taps made by the index and middle finger of one hand. In [18] Xia et al. presented a finger-mounted fine-tip stylus, called 'NanoStylus', that supports high precision pointing on a smartwatch with almost no occlusion (see Fig.2.17).

Utilizing watch's bezel and strap instead of its touchscreen face is a popular technique for efficient interaction. Oakley et al. [67] placed an array of touch sensors on the bevel of a watch to provide high resolution capacitive input. Similarly, the haptic wristwatch [68], made up of a rotatable bezel and touchscreen with haptic feedback, allows for detection of simple, eye-free gestures such as covering the watch, turning the bezel, or swipe over the watch. Xiao et al. [19] moved away from a static bezel and introduced a proof-of-concept to provide mechanical input (such as pan, twist, tilt and click) by moving

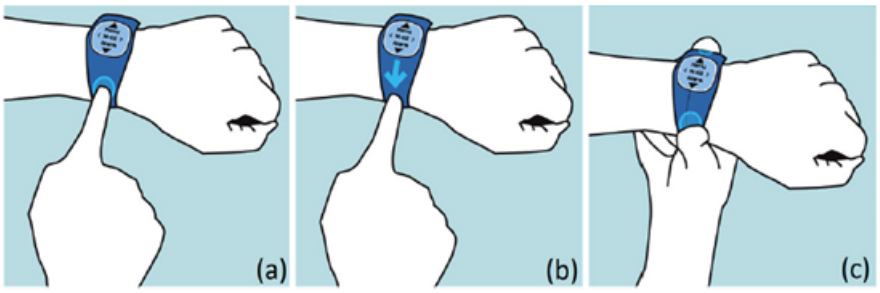


Figure 2.19: The 'WatchIt' prototype extends smartwatch interaction space using its wristband [20].

Figure 2.20: Around device interaction with smartwatches using magnetometer sensor [21].

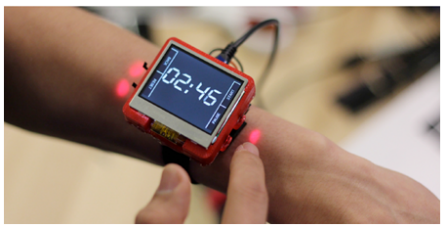


Figure 2.21: The 'Skin Buttons' enables smartwatch interaction using touch-sensitive projected icons [22].

the movable display on a smartwatch (see Fig.2.18). In [20], Perrault et al. presented 'WatchIt' (see Fig.2.19) that uses wristband surface as an input area for occlusion-free selection and scrolling task. Likewise, 'BandSense' [69] allows pressure sensitive multi-touch interaction on a wristband.

In-air gestures based interaction mechanism utilizes the space around the watch for input with minimal screen occlusion. For example, 'Gesture Watch' [70] and 'HoverFlow' [4] augments a watch face with an array of proximity sensors to detect swipe gestures above and around the watch. Abracadabra [21] supports around the watch interaction using magnetometer sensor (see Fig.2.20). In [71], 'Transture' overcomes the spatial constraints of touch gestures on small watch screen by allowing them to continue into the hover state. Knibbe et al. [72] extended the interactive surface of a smartwatch by allowing users to perform single finger, multi finger and whole arm gestures to the back of the hand. They used a combination of infra-red sensors, ultrasound sensors

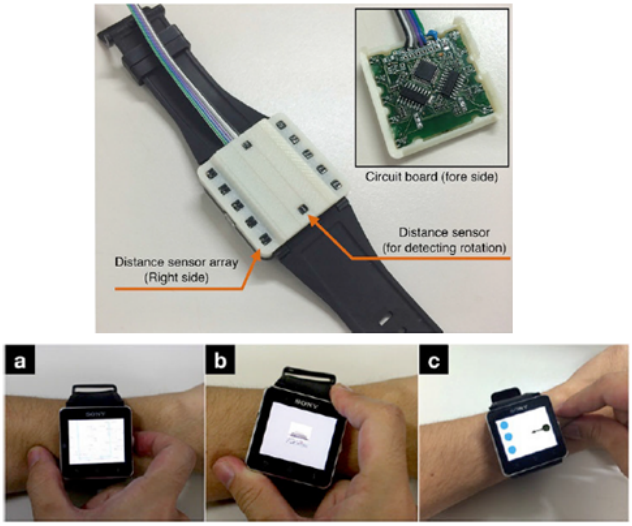


Figure 2.22: The 'SkinWatch' prototype using photo-reflective sensors [23].

and a piezoelectric sensor to recognize six different distinct gestures. In 'Skin Buttons' prototype [22], a user can select the icons projected on the skin by pushing his or her finger on those icons (see Fig.2.21). Basically, it is a projected interface that enables button inputs using laser light and photo sensing techniques.

Other previous works utilized skin, hand, blow, eye-gaze etc. to expand the interaction space of a smartwatch. For instance, 'SkinWatch' [23], an embodied interaction modality, supports gesture input (like pinch, rotation and scroll) by sensing deformation of the skin under wristwatch via photo-reflective sensors (see Fig.2.22).

Interaction with smartwatch demands both hands - dominant and non-dominant. Users face difficulties to interact with watches while carrying something. To address this issue, Akkil et al. [73] presented gaze-gesture based interaction on smartwatches for menu navigation, selecting an item and notification task. 'Blowatch' [24] technique provides blowing as an input for one handed smartwatch interaction (see Fig.2.23). Kerber et al. [25] proposed one-handed, eyes-free smartwatch interactions using electromyography (EMG) armband (see Fig.2.24) and compared its task completion time with respect to touch interactions.

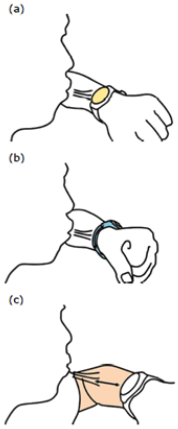


Figure 2.23: The 'Blowatch' supports blowing gestures based smartwatch interactions [24].

Figure 2.24: Control smartwatch using Myo armband [25].

2.5 Summary

In this chapter, we have discussed various approaches related to the context-aware mobile interaction, text entry interfaces, and interaction mechanisms beyond touchscreen input.

In context-aware mobile interaction, earlier work focused on the detection of phone's current position using device's inbuilt sensors or with the help of external hardware. There is no doubt that phone's onboard sensors based placement detection approaches [50] [51] [37] [52] [53] are more appealing than the use of external hardware [48] [36] [49]. Therefore, we are interested in recognizing different surfaces using phone's available sensors. In previous work [50]- [53], proposed approaches are not energy efficient, as these techniques run continuously on the background to sense phone's placement. Moreover, their mechanisms can efficiently detect a limited number of placements. Hence, designing an energy efficient framework which will be able to identify several phone's placement with an acceptable accuracy, is a challenging task.

Around device interaction for mobile devices commonly deals with different types of sensory inputs such as infrared distance sensor [3] [4], RGB camera [5] [6], depth camera [7], electric field sensing [8] and magnetometer [55] [56]. Most of these work

mainly focus on how users can input different gestural command in 3D space to support the natural interaction with mobile devices. However, these techniques demand external hardware as the proposed sensory inputs are not available in the latest generation smartphones. In [55], [56], [57]- [61] researchers deployed phone's inbuilt magnetometer sensor to enhance its interaction space above and around the device. But, they didn't explore the perspective of text entry mechanism. In this thesis, we are mainly trying to use magnetic field based around device interaction technique for text input on the smartphone. It provides the text entry space beyond physical boundaries of a device.

Similarly, designing an efficient text input interface for smartwatches is an important part of the wearable interaction. Most of the earlier research work [10]- [17] related to text entry mechanism on smartwatches tried to fit traditional Qwerty soft keyboard in an intelligent way. The proposed virtual keyboards, which provide good typing accuracy, are slow in nature (i.e. demand high keystrokes per character), and keyboards which support faster typing, are error-prone. However, there are few techniques where researchers used predefined stroke gestures and touch sensitive wristband for typing. In fact, these approaches don't address all challenges regarding smartwatch text input. There is still room for improvements. We can design a text entry interface which will provide a good trade-off between typing speed and error rate. Further, we can also think about touchless (i.e. 3D space) text entry mechanism for smartwatches.

There is another perspective of smartwatch interaction, that is, how can we input different commands or gestures to perform certain actions like zooming, panning, rotation, scrolling, etc. In commercially available smartwatches, we use its tiny touchscreen for interaction. The small screen of the watch suffers from visual occlusion and the fat finger problem. To address these issues, researchers have developed several techniques such as touch-enabled backside of the device interaction [64], NanoStylus [18], watch's bezel [67] [19] and strap [20] [69], projection based interface [22], interaction based on sensing skin deformation [23], eye-gaze [73], and EMG [25]. Most of them require too

much hardware instrumentation, and it is quite difficult to deploy these hardware set-ups into watch form-factor immediately. Hence, we can think about a solution which will be cost-effective and must have the immediate feasibility to fit into smartwatches.

Chapter 3

Phone's Context Sensing Based Mobile Interactions

In recent times, researchers have proposed numerous approaches that allow smartphones to determine user current locations (e.g., home, office, railway station, restaurant, street, supermarket etc.) and their activities (such as sitting, walking, running, bicycling, driving, cutting bread, making coffee, watching television, working at laptop, taking lunch, using water tap, brushing teeth etc.) in real-time. But, to infer much richer story of context-aware applications, it is necessary to recognize the smartphone surfaces - for example on the sofa, inside the backpack, on the plastic chair, in a drawer or your pant pocket. This chapter presents *SurfaceSense*, a two-tier, simple, inexpensive placement-aware technique, that uses smartphone's embedded accelerometer, gyroscope, magnetometer, microphone, and proximity sensor to infer where the phone is placed. It does not require any external hardware and able to recognize thirteen different surfaces with an acceptable accuracy.

The rest of the chapter is organized as follows. Section 3.1 describes the motivation of our work. In Section 3.2, a brief phone placement study has been presented. The Section 3.3 describes SurfaceSense system architecture and Section 3.4 focuses on the

experimental results. The Section 3.5 details the implementation of SurfaceSense as an Android App. In Section 3.6, we show several example applications to enhance mobile interactions using our proposed framework. Finally, Section 3.7 concludes this chapter.

3.1 Motivation

Now-a-days, smartphone became ubiquitous in nature - everyone carries their phone with them almost all time and in all places. Presently, mobile devices know little about the context in which they operate and so, the user takes responsibility for managing their behavior to fit with the current environment. Performing these settings manually is an extra burden to users. For example, if the user enters into a classroom, she has to set her cell-phone into silent mode manually. To provide intelligent supports to users, researchers are focusing on the development of context-awareness into mobile devices such that it should automatically adapt to user's changing environment or context [74] [75]. There are two perspectives in context-aware applications, that is, user's context and phone's context. The user's context mainly focus on detection of user's current location (e.g., home, office, railway station, restaurant, street, supermarket etc.) [26] [27] and their real-time activities (such as sitting, walking, running, bicycling, driving, cutting bread, making coffee, watching television, working at laptop, taking lunch, using water tap, brushing teeth etc.) [28] [29] [30] using smartphone.

So far, it's all about user's context detection. But, to make context-aware applications complete, it is also important to identify phone's context, that is, on the sofa, inside the backpack, on the plastic chair, in a drawer or your pant pocket. Automatic detection of smartphone's context has several advantages. For example, (a) if the phone is in the pocket, then it automatically turns off the display and locks it to prevent pocket-dialing; (b) if cell-phone is in the backpack, then the phone should ring at the highest volume at the time of an incoming call. The similar setting is desirable if it is on the bed or sofa; (c) in pant-pocket location, vibration is sufficient to draw user's attention; (d) if

the phone is on the hard surface like a wooden table or metal cabinet, then vibration may cause damage in the phone. In this case, only ringing is enough; (e) if the phone is put in a drawer, it is reasonable to assume that in near future, the phone will not be used. So, it can go to the power saving mode; (f) if the phone is in a pocket, then don't activate pollution sensor[1]; (g) it also helps to detect user's context. For instance, when a phone is on a table, the microphone may be given the highest priority to estimate user's surrounding environment from ambient sound.

There are two objectives in our work. The first objective is to understand where people generally keep their phones across various contexts such as at home, office-place, driving, sleeping, etc. For this purpose, we carried out a user study and identified 13 different phone placements like a wooden table, sofa-bed, glass table, backpack, plastic chair, cart-box, fabric chair, phone holder, metal chair, wooden-drawer, metal drawer, pant's pocket, and user's hand. In our second objective, we mainly focus on how to identify these locations automatically using phone's inbuilt sensors.

In particular, key contributions of this chapter are summarized as follows:

1. We present the system architecture of SurfaceSense, which follows two-tier hierarchical classification approach, to recognize 13 different phone placements using smartphone's built-in sensors with 91.75% accuracy. The advantage of our proposed method is that it requires a less number of surface candidates to build the classifier module for each subgroup and reduces overall complexity.

2. We propose a simple threshold based algorithm using accelerometer signal to detect the change in phone placement.

3. We implemented SurfaceSense as an Android application on the Samsung Galaxy S4 and analyzed the resource consumption profile (i.e. CPU and Memory usage).

4. Finally, we develop several example applications to show that how the proposed

[1]http://www.gizmag.com/nitrogen-dioxide-sensor/40082/

framework can be used to enhance mobile interactions.

3.2 Phone Placement Study

To understand where people commonly keep their phones in everyday life, we interviewed 92 participants (60 male and 32 female), all aged between 20-38 (Mean = 29). They were primarily graduate, post-graduate students in our university's information technology department. They are all well experienced with smartphones. All of them were given a compensation for their time. We asked them two questions: (*a*) where they normally keep their phones in five different contexts like, in the home while awake, in the home while sleeping, at the office, driving and walking around; (b) how they decide where to put their phone. Table 3.1 shows where people put their phones across various activities. We were surprised that in this study none mentioned belt-pouch as a choice for carrying the mobile phone while they are walking or driving. However, due to the increased size (width and height) of the latest smartphones, it is impractical to carry the phone in belt-pouch. In fact, belt pouches are not commercially available for many latest smartphones. From this phone placement study, we got total 13 surfaces and decision factors for selecting these surfaces are easy to access notifications, phone's safety, physical comfort, minimize distraction, common habit, nearby charger socket and so on.

Table 3.1: Phone placements during different activities.

Activities	**Phone Placements**
Awake at home	wooden table (62%), glass table (17%), in hand (13%),plastic chair (3%), fabric chair (2%), metal chair (2%), cardboard-box(1%)
Sleeping at home	bed (78%), wooden table (9%), glass table (6%), wooden drawer (3%), phone holder (2%), metal drawer (1%)
Working at office	wooden table (81%), pant's pocket (12%), in hand (7%)
Driving	pant's pocket (94%), backpack (6%)
Walking	pant's pocket (91%), backpack (9%)

3.3 SurfaceSense Framework

The SurfaceSense system architecture fundamentally comprises two parts: (1) change in phone placement detection and (2) phone's context sensing. The working of the said two modules is discussed in the following sub-sections.

3.3.1 Change in Phone Placement Detection

Detection of change in phone placement means how the phone will automatically understand that user has changed its current position. For example, a user is taking the phone placed on the table and putting it in his pant's pocket. In this case, phone position has changed from on the table to inside pant's pocket. In this chapter, the proposed change in phone placement detection algorithm will run continuously as a background service. It is a simple threshold based algorithm relying on the captured data of the smartphone's tri-axial accelerometer sensor. Since this algorithm will run continuously, phone's battery consumption is an important concern. To optimize the power consumption of the device, we use only the accelerometer signal as it is the most informative sensor regarding the change in phone placement detection.

Figure 3.1 represents the flow-diagram of change in phone placement detection algorithm. To understand that there is a change in phone placement, we continuously analyze t sec of accelerometer signal (A_x, A_y and A_z) window in real-time. If change in each axis acceleration (ΔA_x, ΔA_y and ΔA_z) exceeds threshold Th_1, then we calculate the norm of the current accelerometer signal as described in Equation (3.1).

$$|A_T| = \sqrt{|A_x|^2 + |A_y|^2 + |A_z|^2} \tag{3.1}$$

Now, check that if $|A_T|$ is within the range of Th_2 and Th_3, that is, $Th_2 \leq |A_T| \leq Th_3$ and two consecutive $|A_T|$ satisfy this criteria within a given time interval of δ msec ($\delta \ll t$), then a counter increases every time and change in phone placement is suspected.

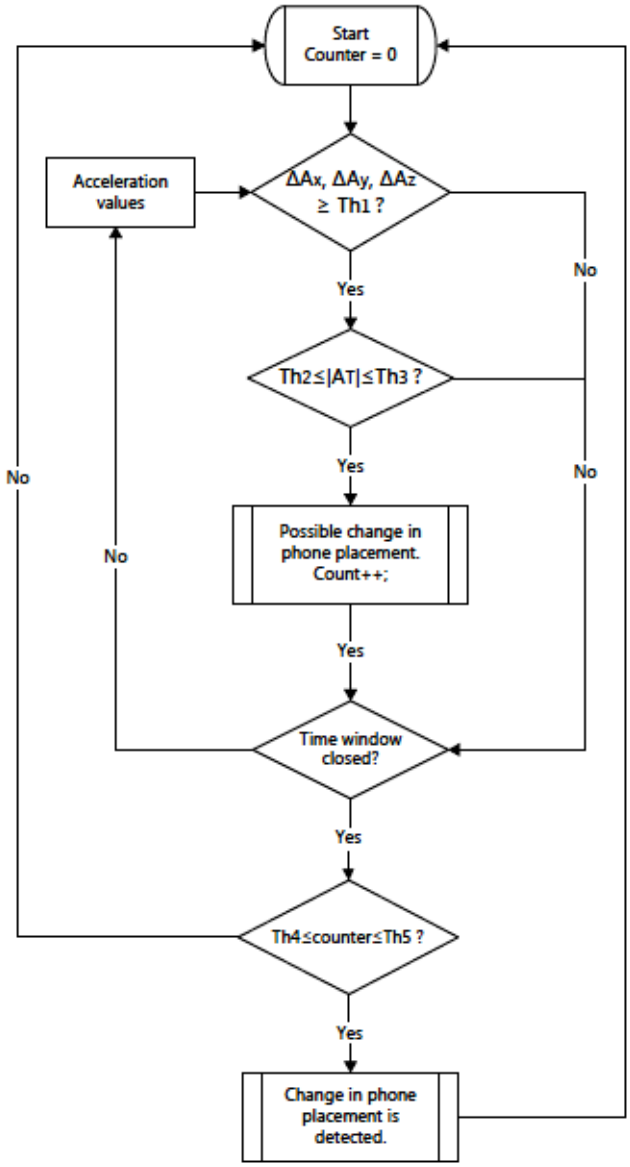


Figure 3.1: Flow diagram of the change in phone placement detection algorithm.

In the final step, if the counter status is $Th_4 \leq counter \leq Th_5$ after t sec window, then there is a real change in phone placement.

If $counter < Th_4$, it means the phone is in the same place. On the other hand, if $counter > Th_5$, then there is a chance that user is doing other activities like walking, running, going up or down the stairs.

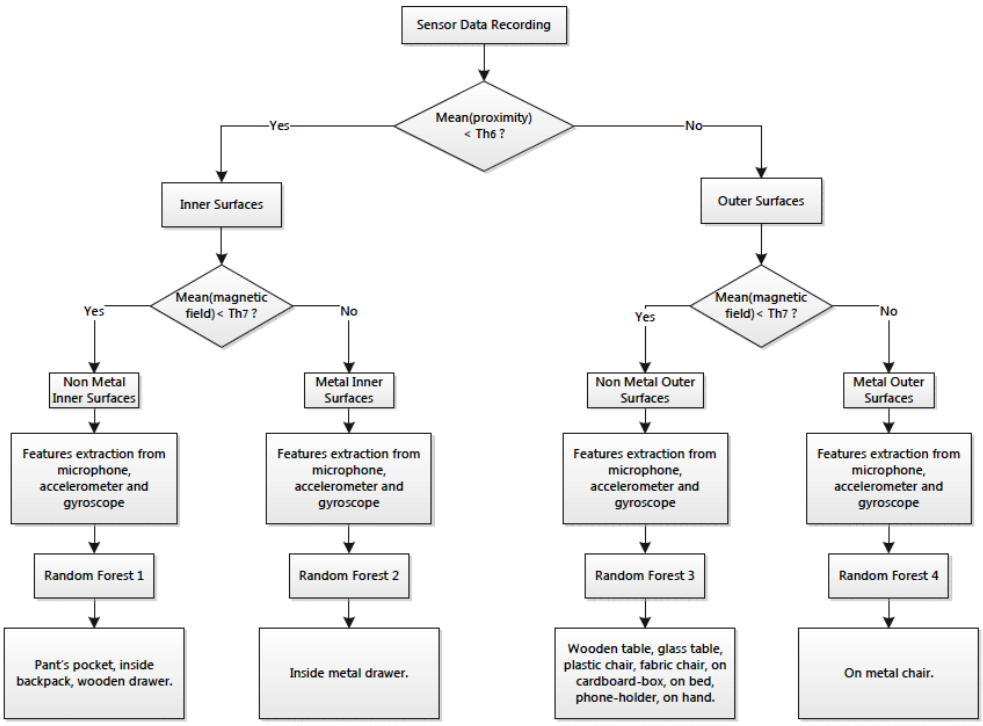
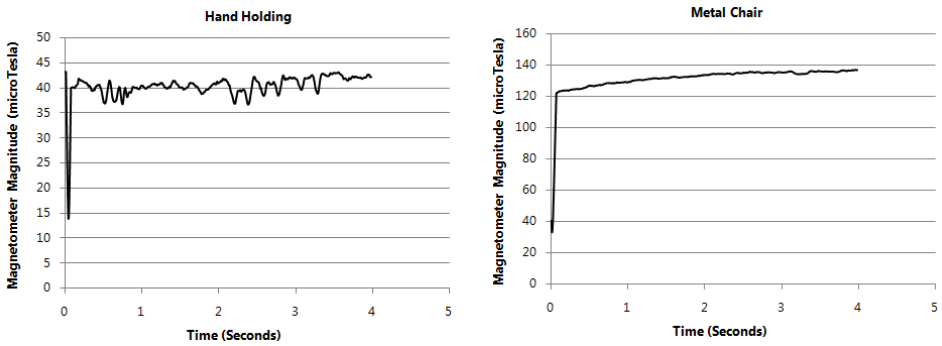


Figure 3.2: Flow diagram of phone's context sensing algorithm.

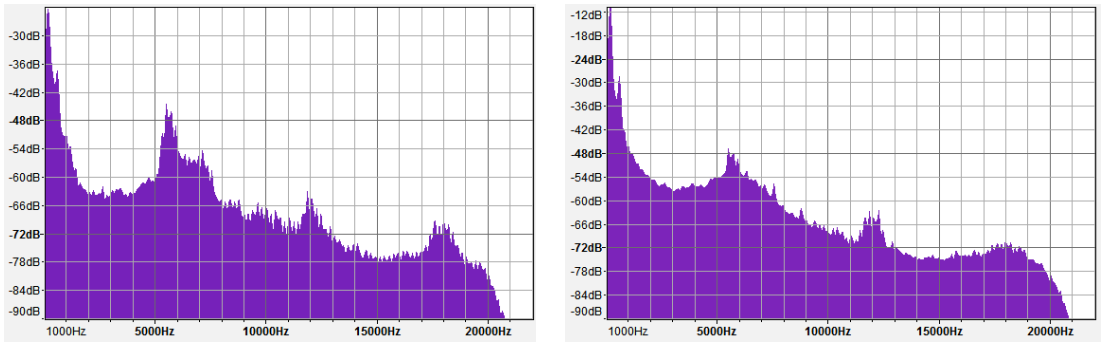


(a) (b)

Figure 3.3: Magnetometer readings for (a) hand-holding position and (b) metal chair position.

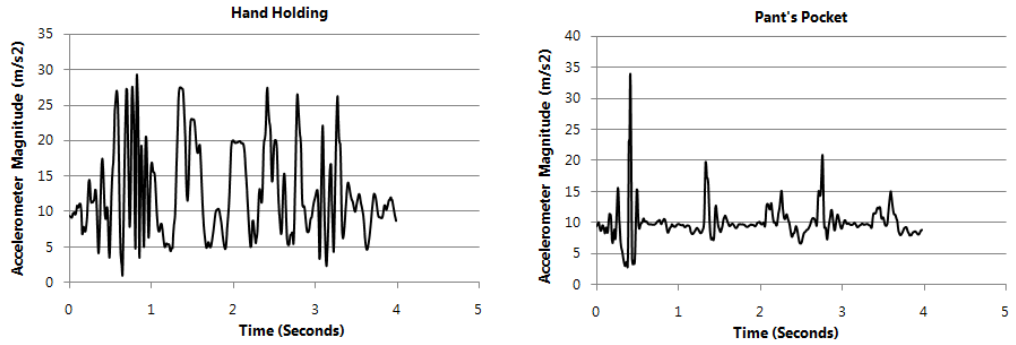
3.3.2 Phone's Context Sensing

If a change in phone placement is detected, then phone's context sensing module starts working. This module fundamentally comprises three parts: (1) surface categorization using proximity and magnetometer sensors (2) feature extraction from microphone, ac-



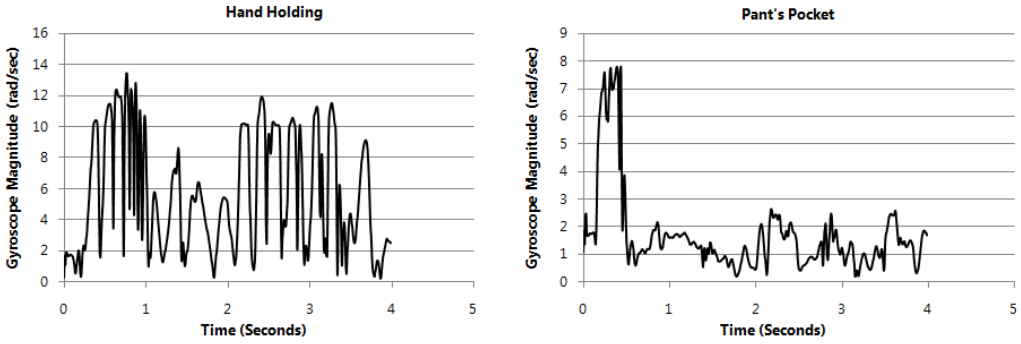
(a) (b)

Figure 3.4: Spectrum of vibration echoes for (a) hand holding and (b) pant's pocket position.



(a) (b)

Figure 3.5: Accelerometer readings for (a) hand holding and (b) pant's pocket position.



(a) (b)

Figure 3.6: Gyroscope reading for (a) hand holding and (b) pant's pocket position.

celerometer and gyroscope sensor data (3) Random Forest classifier to recognize phone placement. Figure 3.2 represents the flow diagram of phone's context sensing algorithm.

The details of each part of this algorithm are discussed below.

3.3.2.1 Surface Categorization

This surface categorization works on two levels. In the first level, a proximity sensor is employed and on the basis of proximity sensor data, phone placements are categorized into two broad groups: inner (i.e. pant's pocket, backpack, etc.) and outer (i.e. plastic chair, wooden table, etc.) surfaces. From the experiment, it is observed that proximity sensor of Samsung Galaxy S4 returns 0.00 c.m. as the average value of proximity distance for inner surfaces and in the case of outer surfaces, it is 8.00 c.m. Therefore, $Th_6 = 5$ c.m. (empirically) is considered as a threshold to distinguish between inner and outer surfaces.

In the second level, magnetometer sensor is used to categorize further each group into two subgroups: metal and non-metal surfaces. To be specific, non-metal inner surfaces are pant's pocket, backpack, and wooden drawer; the metal inner surface is metal drawer; non-metal outer surfaces are wooden table, soft-bed, glass table, plastic chair, cart-box, fabric chair, phone holder, and user's hand; metal outer surface is metal chair. In the experiment, we logged magnetometer sensor data at the sampling rate of 100 Hz. For non-metal surfaces, strength of magnetic field varies between 15 μT to 50 μT and in case of metal surfaces, it is approximately 65 μT to 200 μT. For example, Figure 3.3(a) & (b) represent magnetometer readings related to the positions of hand-holding and metal chair respectively. Here, we choose $Th_7 = 62$ μT as a threshold to distinguish metal surfaces from non-metal one.

3.3.2.2 Feature Extraction

During vibration phase, the device itself records the vibration echoes using a microphone at 44.1 KHz sampling frequency and motion data with the help of accelerometer and gyroscope sensors at 100 Hz sampling frequency. These vibration sound and motion data

vary depending on surfaces. For example, Figure 3.4(a) & (b) represent different sound signatures for hand-holding and pant's pocket positions respectively. Figure 3.5(a) & (b) depict the magnitude of raw accelerometer values for two smartphone positions, hand-holding and pant's pocket. Likewise, Figure 3.6(a) & (b) display two plots of gyroscope readings related to hand-holding and pant's pocket positions. To characterize different surfaces, we extract time and frequency domain features from recorded signals. Feature extraction step consists of three parts: (a) vibration sound features (b) accelerometer features and (c) gyroscope features.

Vibration Sound Features This sound fingerprint is processed in frames with a 20 msec sliding window and 50% overlap. Each window is smoothed with a Hamming filter, and the following features are extracted. [1 - 2]: zero crossing rate, short-time energy (time domain); [3 - 6]: spectral flux, spectral rolloff, spectral centroid, and Spectral entropy (frequency domain). For each 6D feature vector, the standard deviation is calculated over all windows.

Accelerometer Features The 3 axis accelerometer readings are divided into frames, and we find global vertical (the direction of gravity) and horizontal (perpendicular to the direction of gravity) components from each frame to eliminate smartphone's different orientation effects in the feature set. To do this, we use a simple normalization scheme as described in [76]. The algorithm works as follows.

Let the raw accelerometer readings in a frame be $a_i = [a_x(i), a_y(i), a_z(i)]$, $i = 1,...,n$, where $a_x(i)$, $a_y(i)$, and $a_z(i)$ are the accelerometer readings along x, y and z axis respectively and n is the number of accelerometer readings in a frame. Note that, a non-overlapping rectangular window is used in the framing process and window size is 320 msec. We obtain vertical acceleration vector p corresponding to gravity as $p = [m_x, m_y, m_z]$, where m_x, m_y and m_z are the average values in each axis, that is, $m_x = \frac{1}{n}\sum_{i=1}^{n} a_x(i)$. The dynamic component of a_i, caused by the user's motion rather than gravity, is rep-

resented as $d_i = [a_x(i) - m_x, a_y(i) - m_y, a_z(i) - m_z]$. Then using vector dot product, vertical component v_i is computed as $v_i = (\frac{d_i \bullet p}{p \bullet p})p$. The horizontal component, h_i, is calculated as $h_i = (d_i - v_i)$. Finally, we use $\| h_i \|$ and $\| v_i \|$ as horizontal and vertical components to extract following features. [1 - 8]: mean, std, min and max of vertical and horizontal acceleration respectively (time domain); [9 - 16]: min, max, kurtosis and skewness of vertical and horizontal acceleration respectively (frequency domain). For each 16D feature vector, the standard deviation is calculated over all windows.

Gyroscope Features We consider only time domain features from the magnitude of gyroscope readings, that is, [1 - 4]: mean, std, min, and max. Here, we use a non-overlapping rectangular window of size 320 msec for framing purpose. Finally, to get 4D feature vector from recorded gyroscope readings, the standard deviation is computed over all windows.

Ultimately, we have total 26 features, that is, our current feature vector is of 26 dimensions (6D + 16D + 4D).

3.3.2.3 Classifier

We use Random Forest (RF) classifier [77], provided by WEKA tool[1], to recognize phone placements. RF is an ensemble learning method for classification that operates by constructing a multitude of decision trees at training time and outputting the class that is determined by a majority vote of the trees. We preferred Random Forest because it is fast and efficient for training, and more importantly, it is computationally much lighter than other classifiers. Note that, computational time of RF is $O(T(MN \log(N))$, where T is the number of trees in the ensemble, M is the number of features and N is the number of samples in the dataset.

The configuration of the random forest classifier is as follows: (a) It is a classification model. (b) Number of trees used are 100 (bagging). (c) Random Forest (RF) uses

[1] http://www.cs.waikato.ac.nz/ml/weka/

standard variance reduction as split selection criterion. (d) In RF, each tree is 'fully grown' which means it continues until each terminal node has a specified number of data points. So the total number of nodes in each tree is not fixed. There is a max nodes parameter than can be set to limit the total number of nodes, but I think that is just to prevent trees from getting to large to fit in memory rather than being an tunable accuracy parameter. (e) Bagging has a single parameter which is the number of trees. (f) Fraction of random features used per node in sqrt(D). (g) Depth of trees present in the forest is not fixed. All trees are fully grown binary trees (unpruned). (h) Pruning is required in decision trees to avoid overfitting. In random forest, the data sample going to each individual tree has already gone through bagging (which is again responsible for dealing with overfitting). There is no need to go for pruning in this case. (i) Our training dataset is quite well balanced across all classes. So, there is no worry about imbalanced dataset while performing classification with random forest.

To detect phone's current context, we used four Random Forest classifiers. As per Figure 3.2, Random Forest$_1$ for non-metal inner surfaces, Random Forest$_2$ for metal inner surfaces, Random Forest$_3$ for non-metal outer surfaces, and Random Forest$_4$ for metal outer surfaces. The basic difference between these classifiers is that they are trained with different datasets. For example, Random Forest$_1$ is trained with a dataset which belongs to the non-metal inner surface group, whereas, the dataset of Random Forest$_2$ belongs to the metal inner surface group.

3.4 Experiments

3.4.1 Performance of Change in Phone Placement Detection Module

We implemented change in phone placement detection module using Samsung Galaxy S4 Android smartphone and we capture its accelerometer data at 100 Hz sampling frequency. Time window (t), data interval (δ) and threshold setting are an essential part

of our algorithm as shown in Figure 3.1. We analyzed accelerometer pattern from 10 participants in different phone placement changing scenarios and set $t = 2$ sec, $Th_1 = 2.5$ m/sec^2, $Th_2 = 11$ m/sec^2, $Th_3 = 16$ m/sec^2, $\delta = 250$ msec, $Th_4 = 1$ and $Th_5 = 8$ after conducting experiments. Ultimately, we achieved 71.24% accuracy to detect the change in phone placement.

3.4.2 Performance of Phone's Context Sensing Module

3.4.2.1 Data Collection

For the experiments, we use Samsung Galaxy S4 GT-I9500 Android smartphone which has inbuilt microphone, accelerometer, gyroscope, magnetometer and proximity sensors. To collect training and test dataset, we tried for three days in a lab environment with Galaxy S4 phone on 13 different surfaces. For each surface, we vibrated the phone for 4 seconds and recorded vibration sound using microphone at 44.1 KHz sampling rate, motion data using accelerometer and gyroscope at 100 Hz sampling frequency, magnetic field strength using magnetometer at 100 Hz sampling rate and proximity distance using proximity sensor at a sampling frequency of 10 Hz. We repeated this procedure 80 times for each surface with different phone orientations. In this way, we collected total 1040 samples (80×13) and divided it into four datasets, that is, dataset$_1$ for Random Forest$_1$, dataset$_2$ for Random Forest$_2$, dataset$_3$ for Random Forest$_3$ and dataset$_4$ for Random Forest$_4$. The dataset$_1$ contains 240 samples from pant's pocket, backpack, and wooden drawer; dataset$_2$ contains 80 samples from metal drawer; dataset$_3$ contains 640 samples from wooden table, soft-bed, glass table, plastic chair, cart-box, fabric chair, phone holder, and user's hand; dataset$_4$ contains 80 samples from metal chair.

3.4.2.2 Classification Results

We split each dataset into three subsets for conducting three-fold cross validation test. Table 3.2 represents the overall performance of SurfaceSense system. The average accu-

racy stands at 91.75%, with more than half of the phone placements achieving accuracies over 90%. From our result, it is observable that soft and hard surfaces can be distinguished with good accuracy, but it is quite difficult to recognize several similarly hard surfaces (e.g. wooden table and glass table). However, this is an excellent outcome with respect to the results of few previous works [50], [51], [37] and [52].

Table 3.2: Performance evaluation of 13 phone placement detection.

Placements	**Accuracy**	**Confused Surfaces**
Pant's pocket	98.82%	backpack (1.18%)
Backpack	97.21%	pant's pocket (2.79%)
Wooden drawer	100%	-
Metal drawer	100%	-
Wooden table	78.43%	glass table (16.61%), card-board box (4.96%)
Glass table	69.41%	wooden table (25.12%), plastic chair (5.47%)
Card-board box	78.88%	glass table (13.52%), plastic chair (3.9%), wooden table (3.7%)
Plastic chair	95.2%	glass table (3.2%), card-board box (1.6%)
Soft-bed	86.37%	fabric chair (13.63%)
Fabric chair	88.54%	soft-bed (11.46%)
Phone holder	100%	-
User's hand	100%	-
Metal Chair	100%	-

3.5 SurfaceSense Android App

We developed 'SurfaceSense' as an Android app in Samsung Galaxy S4. The main components, described in Section 3.3, were entirely implemented in Java SE 7 and successfully running on the Android smartphone. It took almost 22.35 seconds to detect a phone-placement correctly. We also measured the CPU, memory, and power consumption footprints of SurfaceSense app on Galaxy S4 with the help of PowerTutor[1] and OS

[1]https://play.google.com/store/apps/details?id=edu.umich.PowerTutor&hl=en

Monitor[1] applications, available at Google Play Store. The CPU usage is less than 4% during idle state and on average of 14% at the processing time. The memory consumption is ∼7.38 MB during silence and reaches ∼12.26 MB while running. The average power consumption for sensor reading is less than 18.76mW.

3.6 Mobile Interactions Based on SurfaceSense

We developed several example applications to show how this SurfaceSense framework can be used for mobile interactions. For examples:

Placement-based Notification: Using SurfaceSense would enable a person to set placement-specific notification preferences. Vibrate mode is preferable if the phone is inside a trouser pocket while ringing mode is most suitable if it is on the table.

Fine tuning of 'pedometer' app: If phone placement is inside the pant's pocket, then only pedometer app will count our footsteps. If the phone is in user's hand, then this app should stop counting because involuntary hand movement may mislead its accuracy.

Find my phone app: Sometimes we forget where we keep our phone after the last used. This app will help us to find our phone by sending the message like 'your phone is at the office on your desk' or 'your phone is at home in your backpack.' In this application, we integrated phone's GPS with SurfaceSense to infer more details context information.

3.7 Conclusion

We presented SurfaceSense, which is inexpensive, calibration free, and does not demand any external hardware. It only requires phone's built-in accelerometer, gyroscope, magnetometer, microphone and proximity sensors. This placement aware technique can be applied easily to develop various context-aware applications. Our proposed method provides almost 91.75% accuracy to recognize 13 different phone placements. One technical

[1]https://play.google.com/store/apps/details?id=com.eolwral.osmonitor&hl=en

issue is that we did our experiment with a plastic covered phone, but recognition rate may be affected by the presence of rubber cover or a leather cover. In future, we'll study our algorithm (a) with different mobile devices except Samsung Galaxy S4 and (b) in a noisy environment to test its feasibility at high scale.

Chapter 4

Magnetic Field Based Around Device Interaction for Text Entry in Smartphone

This chapter presents 'MagiText' that expands the text entry space beyond the physical boundaries of a device, to overcome the problems of touchscreen input. Our proposed approach uses the mobile phone's inbuilt magnetometer sensor to recognize 3D space handwritten character gestures. The key idea is to influence the magnetic sensor by writing character gestures (i.e. Graffiti or EdgeWrite) in front of the device using a properly shaped magnet taken in hand. The movement of this magnet changes the magnetic flux pattern around the device, and it is sensed and registered by the magnetometer sensor. Then, we classify this flux pattern using Support Vector Machine (i.e. SVM) classifier. The experimental result shows that participants can achieve better accuracy in character recognition with EdgeWrite gesture-set compared to Graffiti.

The rest of the chapter is structured as follows. Section 4.1 describes the motivation of our work. Section 4.2 describes the MagiText framework and Section 4.3 includes experimental results. Section 4.4 details the implementation of MagiText as an Android

App. Finally, Section 4.5 concludes this chapter.

4.1 Motivation

With the proliferation of electronic technology, small handheld devices (i.e. smartphones, tablets, music player, etc.) are now affordable and pervasive in many usages. These devices have become the mainstream communication and computing platform. In fact, they provide a rich text input space. But, none of these devices do have a physical keyboard, most text entry is done by using small soft keyboards displayed on the device's touchscreen. A standard keyboard contains almost 104 keys, and to fit all these keys in a small area, we usually reduce the number of displayed keys to a minimal amount, for example, the letters of the alphabet and one or more status modifier keys. Although this approach helps, it leads to keys of small sizes. For instance, the popular Swype[1] virtual keyboard provides each key size of 0.5 $\times$ 0.9 cm.

Further, the traditional way of entering text with the smartphone is Qwerty soft keyboard which demands a long visual search time and suffers from fat finger problem [78]. It is also hard to hit the keys while users in mobile scenarios and at the same time, letters are difficult to read, especially when eyesight-impaired. In recent past, researchers have developed several techniques to solve these problems. They proposed different modified versions of virtual keyboards such as BigKey [79], Fisheye keyboard [80], HoverZoom [46] etc. They tried with gesture-based approaches such as Vector keyboard [81], Unistrokes and Graffiti [82], ShapeWriter [83], H-4 Writer [84] etc. They also used speech [85] as an alternative, but it suffers from environmental noise and it demands server-based architecture for automatic speech recognition.

In this dissertation, we are particularly interested in around device interaction technique (ADI) for text input on smartphones because this approach expands the text entry space beyond the physical boundaries of a device. This ADI commonly deals with dif-

[1] http://www.swype.com/

ferent types of sensory inputs such as infrared distance sensor, camera, depth sensor, electric field, and magnetic field. Among all these sensory inputs, magnetic field based ADI is much simpler as (1) its hardware (i.e. magnetometer sensor) is already available in the current mobile devices; (2) it consumes very less power; (3) it does not suffer from illumination variation and occlusion problems like camera based ADI; (4) it can pass through many materials, for example, it enables in-pocket interaction. This chapter provides the details design and implementation of the magnetic field based around device interaction technique for text input with mobile devices.

4.2 MagiText Framework

The MagiText system architecture fundamentally comprises three parts: (1) input from magnetometer sensor (2) feature extraction from sensor data (3) modeling of multi-class Support Vector Machine (i.e. SVM) classifier. The working of the said three modules is discussed in the following sub sections.

4.2.1 Input from Magnetometer Sensor

MagiText supports the effective use of 3D space around the device for handwritten English alphabets recognition. The underlying principle of our MagiText approach is to influence the phone's embedded magnetic sensor by drawing character gesture around the device along 3D trajectories using a proper shaped (may be a ring, disk or rod type) magnet mounted on the finger. Here, we used Graffiti[1] (see Figure 4.1) and EdgeWrite [47] (see Figure 4.2) as character gesture and users have to press a button to indicate the starting and ending it. The temporal pattern of the magnetic flux's deformation is sensed by capturing the sensor values on x, y, and z coordinates. Note that, the range of this output changes from device to device. For instance, in Samsung Galaxy S4, the value range is $\pm 200\mu$T, whereas, it is $\pm 128\mu$T for iPhone 3GS. The

[1] http://en.wikipedia.org/wiki/Graffiti_(Palm_OS)

Figure 4.1: Graffiti Characters.

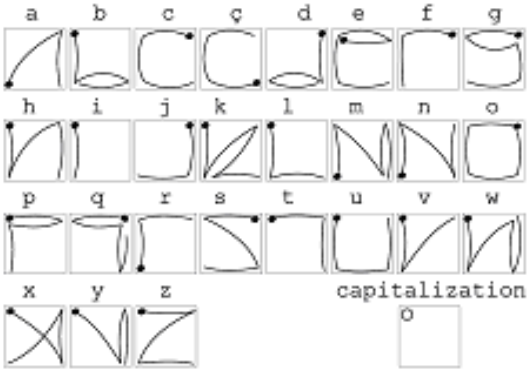


Figure 4.2: EdgeWrite Characters.

magnetic sensor can be affected by Earth's magnetic field and to eliminate this magnetic noise, we apply a time derivative operator on the output signals of the magnetic sensor.

4.2.2 Features Extraction

The next step in MagiText is feature extraction from recorded magnetometer signal. Here, we extract features over samples in an interval labeled with the starting and end of the gesture.

To capture the temporal pattern of the character gesture in a more detailed way, we divide the gesture interval into three equal length windows, extract a feature vector from each window and finally, concatenate three feature vectors to form a new feature vector to be used for the gesture classification step. Features used in this study are listed in Table 4.1. These features yield 32 elements feature vector for each window and altogether three windows form 96 elements feature vector for each character gesture.

Table 4.1: Selected features of magnetometer signal.

Type	Feature Name	Coef.
Time Domain	Mean and S.D. of magnetic field strength along x, y, and z	6
	Mean and S.D. of Euclidian norm of magnetic field strength along x, y, and z	2
	Piecewise correlation between magnetic field strength along xy, yz, and zx	3
	Zero Crossing Rate (ZCR) along x, y, and z	3
Frequency Domain	Mean and S.D. of magnetic field strength along x, y, and z	6
	Max. and Min. of magnetic field strength along x, y, and z	6
	Kurtosis and Skewness of magnetic field strength along x,y, and z	3

Notes: S.D.: Standard Deviation; Coef.: Coefficients

4.2.3 SVM Classifier Modeling

The extracted feature vector is used as input to multi-class Support Vector Machines (SVM) to classify different character gestures. There are two approaches to build multi-class SVM : (a) one-versus-one and (b) one-versus-all. In our approach, we have used one-versus-all strategy [86]. The non-linear kernel function namely radial basis function (RBF) is applied in our experiment and it is defined as

$$RBF : K(x, x') = \exp^{(-\frac{\|(x-x')\|_2^2}{2\sigma^2})} \tag{4.1}$$

x and x' represent feature vectors in input space. $\| (x-x') \|_2^2$ denotes squared Euclidean distance between the two feature vectors. σ is a kernel parameter and set $\sigma = 10$.

4.3 Experiments

This section consists of three subsections which are discussed below.

Figure 4.3: Experimental setup of MagiText.

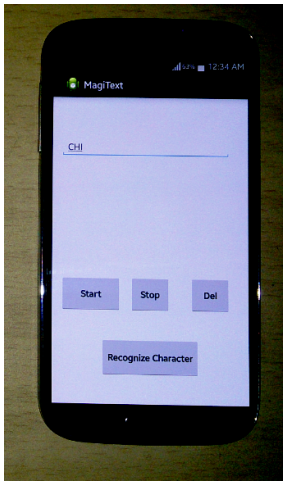


Figure 4.4: GUI of MagiText.

4.3.1 Apparatus

We used inbuilt magnetometer sensor of Samsung Galaxy S4 GT-I9500 Android smartphone and placed N54 grade, disk-shaped (10mm × 3mm in diameter and height respectively) neodymium magnet on the finger using a small strip of Velcro. This magnet provides a useful range of about 10 cm and its magnetic field strength is 0.7179 Tesla.

4.3.2 Data Collection

To collect training and testing dataset, we tried for four days in the lab environment. We invited 3 users (2 Male + 1 Female), aged between 22 and 26 years. All were regular smartphone users, right-handed, and spend on an average 4 hours per day. But, none had used Graffiti and EdgeWrite earlier. Each participant received a demonstration before experiments and practiced each alphabet at least 15 times. For quick learning, a printed copy of two unistroke character sets was visible during trials.

Each user holds the phone in one hand and moves the magnet mounted finger in front of the device (see Figure 4.3). They were asked to repeat each gesture 50 times. We developed an Android application to record the magnetic signals at 100 Hz sampling

frequency. In this way, we collected total 7800 samples (i.e. 50 times $\times$ 3 users $\times$ 26 alphabets $\times$ 2 types gesture set). There was variation in duration of recordings per character per user. This difference also depends on the input character gesture. We observed that average time to input an alphabet was 2.68 sec. Note that, we considered this variation in duration of recording in our training dataset. Then, features are extracted from those signals as described earlier. The extracted features are used for classification using SVM.

4.3.3 Classification Results

We perform two separate experiments - one for Graffiti and another for EdgeWrite. The Graffiti and EdgeWrite dataset contain 3900 samples each. We split each dataset into 10 subsets for carrying out a ten-fold cross validation test. The number of samples in training, validation, and test sets are in 6:2:2 ratio. To analyze the classifier performance, we used F_1 score [87]. Let, the amount of true positives is TP, the amount of false positives is FP, and the amount of false negatives is FN. Then, $Precision$ is defined as $P = \frac{TP}{TP+FP}$. $Recall$ is defined as $R = \frac{TP}{TP+FN}$ and F_1 is $\frac{2PR}{P+R}$. The F_1 score ranges from 0 to 1. Table 4.2 represents the overall performance of MagiText system. From our result, it is observed that MagiText system based on EdgeWrite character set can achieve 89.4% accuracy whereas Graffiti can distinguish characters with 81.2% recognition rate. This is because a user can easily map the EdgeWrite character's corners with the phone's four corner points at the time of drawing gestures. Hence, EdgeWrite gesture input is less ambiguous compared to Graffiti.

Table 4.2: Character Recognition Performance of MagiText System

Characters	Recognition Rate(F-Score)	
	Graffiti	EdgeWrite
A	0.764	0.924
B	0.719	0.841
C	0.733	0.866
D	0.845	0.929
E	0.825	0.931
F	0.862	0.908
G	0.759	0.972
H	0.792	0.861
I	0.828	0.848
J	0.814	0.839
K	0.836	0.954
L	0.802	0.850
M	0.786	0.849
N	0.774	0.852
O	0.882	0.938
P	0.806	0.912
Q	0.844	0.938
R	0.783	0.880
S	0.852	0.929
T	0.863	0.914
U	0.794	0.908
V	0.786	0.889
W	0.828	0.861
X	0.830	0.916
Y	0.848	0.894
Z	0.834	0.851
Average	**0.812**	**0.894**

4.4 Resource Profile of MagiText: CPU, Memory & Power Consumption Benchmark

We developed MagiText system as an Android application in Samsung Galaxy S4. The principal components, as described in 'MagiText Framework' section, were fully implemented in Java SE7 and successfully running on Android smartphone. To build this

prototype, we used libgdx library[1] to perform a fast Fourier transform (FFT) on incoming sensor signals in real time. We also used SMO WEKA[2] machine learning library for SVM classifier. Figure 4.4 shows the app's user interface (UI). By selecting 'Start' and 'Stop', a user enters character gestures and collects magnetic sensor readings and 'Del' button is used to delete recorded magnetic signal. Finally, by pressing 'Recognize Character' button, it extracts features from the signal and recognizes a particular character from input gesture. The duration of the overall process is ∼35.23 seconds.

We also measure the CPU, memory, and power consumption footprints of MagiText with the help of 'PowerTutor' and 'OS Monitor' apps, available at Google Play Store. The CPU usage is less than 4% during idle state and on average of 16% at the processing time. The memory consumption is ∼7.8MB during silence and reaches ∼20.54MB during running period. The average power consumption for sensor reading is less than 18.76mW.

4.5 Conclusion

We presented MagiText, based on drawing character gesture in the space around the device using a magnet. We also compared the character recognition accuracy on the basis of Graffiti and EdgeWrite gestures. This approach can be particularly suitable for mobile and tangible devices, and we can use it as an efficient text entry mechanism, instead of touch screen keyboard, for taking short messages or notes quickly. Magnetic field sensing based around device interaction is elegant because it uses magnetometer sensor that is already available on current generation mobile devices. Moreover, magnets are passive and have no power requirements. An obvious limitation is that the user needs to be instrumented with a ring-shaped magnet.

In this work, we did not evaluate the text entry speed (i.e. words per minute) because our system is taking almost 30.18 seconds to recognize a character. This is

[1]http://www.java2s.com/Code/Jar/g/Downloadgdxaudiojar.htm
[2]http://www.cs.waikato.ac.nz/ml/weka/

our early exploration of around device interaction based text input system. However, a lot of things need to be done to make this approach robust and practically usable. In future, we will focus on (1) studying sophisticated signal processing and machine learning algorithms (i.e. Hidden Markov Models) to improve MagiText performance in terms of character recognition accuracy and time required to identify a character gesture; (2) user independent character recognition; (3) automatic identification of gesture starts and ends; in this regard, performing a Viterbi decoding might be useful to find character boundaries implicitly; (4) a large-scale user study to understand the acceptability of MagiText system.

Chapter 5

Efficient Text Input on Smartwatches

In the present day context of wearable computing, smartwatches augment our mobile experience even further by providing information at our wrists. It offers the ability to read text messages, emails, notifications, etc. instantly, once it is synchronized with a smartphone. But, performing efficient text input task on the smartwatch is quite difficult due to its tiny touchscreen display. In this chapter, we present two techniques for text entry on smartwatches - 'ETAO keyboard' and '3D typing using hall effect sensors'. The ETAO keyboard is a full-fledged soft-keyboard where a user can input the most frequent English alphabets with a single tap and other keys (i.e. numbers and symbols) by a double tap. On the other hand, the Hall effect sensors based text entry technique effectively uses the 3D space around the smartwatch for entering alphanumeric characters. This technique does not consume any screen space, does not need any visual search to find a character and does not suffer from fat finger problem.

The rest of the chapter is structured as follows. Section 5.1 and Section 5.2 describe the details of ETAO keyboard and 3D typing using Hall sensors respectively. There are five subsections under each section and these sub-sections present motivation, the design

of proposed approach, implementation, user study result, and conclusion.

5.1 ETAO Keyboard: Text Input on Smartwatches

5.1.1 Motivation

Over the past few years, the world has seen a rapid growth in wearable computing and demand for wearable products. In recent times, smartwatches have gained a lot of public attention as one of the most popular wearable devices. It allows users to access several applications (messaging, email, calendar, maps, etc.) running on smartphones, without the need to use their phones. Although applications are instantly accessible on the watch, users face difficulties to reply immediately as there is no text entry method on the same device. While Qwerty soft keyboard has become the dominant text input modality for mobile devices but it's hard to fit on tiny wearable devices. Most present day smartwatches either don't offer a virtual keyboard as a text entry mechanism or provide methods like shorthand gestures which take lengthy user training sessions to get accustomed to it. Most modern smartwatches support the 'speech to text' mode, but there are some restrictions of voice typing - privacy issues, surrounding noise, various pronunciation styles and it demands constant Internet connectivity.

Most of the earlier research works [10]- [17] related to text entry on smartwatches tried to fit traditional Qwerty soft keyboard in an intelligent way and also used touch sensitive wristband for typing. The existing virtual keyboards, which provide good typing accuracy, are slow in nature and keyboards which support faster typing, are error-prone. Our aim in this work is to develop a keyboard which will try to establish a good trade-off between typing speed and error rate.

In this chapter, we present ETAO keyboard, a technique that supports faster and less erroneous text input on ultra-small interfaces of smartwatches. It supports all English alphabets, numbers and most symbols that we use on a daily basis. Using our proposed

(a) (b) (c) (d) (e)

Figure 5.1: ETAO keyboard layout: (a) home screen (b) intent layout of the second button of first row (c) intent layout of the third button of first row (d) digit's intent and (e) special symbol's intent.

keyboard user-interface (UI), a user can select most frequent characters (i.e. E, T, A, O, I, N, S and R) with a single tap and remaining characters, numbers and symbols with two taps. It supports faster typing with minimum keystrokes per character (KSPC). We use two swipe gestures for delete and space. Here, we consider 'tap' as the prime input method because it is really easy to perform when walking in a street. Moreover, this layout quickly eliminates 'fat-finger problem' by providing keys with bigger buttons.

5.1.2 Design

The area provided by a smartwatch is really small (1.65" screen diagonal) hence, in our proposed ETAO keyboard, we apply the concept of key layering where certain keys appear in one layer, and the rest appear in other layers. Except the two middle keys in the first row, the size of each key has been set at 40 dp. This size has been chosen after many trial and error tests so that the keys are not too small or too large to hamper the layout.

ETAO keyboard supports all English alphabets, numbers and most symbols that we use on a daily basis. The design layout consists of two layers of input modes where a user can access most frequent letters (i.e. E, T, A, O, I, N, S, and R) with a single tap and remaining characters with two taps. The first layer i.e. main screen is divided into two regions. The top most region has a text field where typed characters will appear. The bottom region has four 'grid keys' and eight individual buttons.

Figure 5.2: ETAO Keyboard prototype running on LG W100 smartwatch.

The leftmost (first) grid key is used for numeric entry. The second and the third (grid) buttons are used to enter the remaining English alphabets which are not mentioned separately. Each of these grid keys houses 9 buttons of 40 dp each. The second grid has the letters B,C,D, F,G,H and J,K,L arranged alphabetically (see Figure 5.1(b)). The third grid key comprises M,P,Q, U,V,W, and X,Y,Z also arranged in alphabetical sequence (see Figure 5.1(c)). The right most (fourth) grid button is used to enter special characters such as symbols and punctuation marks (see Figure 5.1(e)). There are four buttons that contain two symbols each. For example, the opening and closing parenthesis '(' and ')' are present on a single key. To enter the opening parenthesis, the user just needs to tap the button and in order to enter the closing parenthesis, the user has to long press the same key. Moreover, a special back button is provided in all the four grid keys. This is to help the user to get back to the main screen in cases of unintentional opening of a layer.

The remaining eight buttons on the main screen correspond to the characters 'E','T', 'A', 'O', and 'I', 'N', 'S' and 'R'. These eight characters are arranged on the basis of most frequently occurring English alphabets, starting with highest frequency character 'E' to the relatively less frequent 'O'. Note that, these eight letters cover almost 65.04% of all letter frequency occurrences in English [1].

Two swipe gestures are designed for space and delete key. To input a space the user

[1]http://www.math.cornell.edu/ mec/2003-2004/cryptography/subs/frequencies.html

has to swipe down by tapping either of the two grid buttons present in the middle and similarly, to delete a character the user has to swipe left by tapping on either of these grid buttons. Long pressing any key makes the character capitalized. When a key is pressed it provides a haptic feedback via 100 msec vibration to the user.

5.1.3 Implementation

We use the 'LG W100 Watch' and the 'Android Wear' platform for implementing our ETAO keyboard (see Figure 5.2). The G watch comes with Android Wear as the native OS. Hence, the development of the app had to be done on Android 4.4W API, which is compatible with Android Wear. This API fully supports the gestures that we incorporated in our application. The watch has the screen size of 30mm × 30mm. The home screen has all the buttons on 25mm × 25mm layout which includes all the three rows of the 'button region'. We used a keypad size of approximately 20mm × 20mm in a grid (the second layer of input) and placed nine keys with a return button in this grid. This translates to about 6.6mm × 6.6mm for each key and this area is big enough to avoid most fat-finger problems. To compare the key-size, we consider few other keyboards' key-size. For example, the size of the ZoomBoard [10] key is 2.9mm × 2.9mm when not zoomed and in the zoomed state is 5.8mm × 5.8mm.The area of each key of the SplitBoard [14] is 4.8mm × 6.5mm and is a little larger for space and backspace key.

5.1.4 User Study

5.1.4.1 Method

To evaluate the feasibility and practicality of ETAO keyboard, we performed some text entry tests and compared it with three existing keyboards like Qwerty, ZoomBoard, and SplitBoard. Ten post-graduate students (6 male + 4 female; mean age: 24) were recruited. They are all well experienced with smartphones, but not with smartwatches.

Before the beginning of the tests, the participants were shown the interface and

were informed of the gestures that were built into the keyboard. A demo session was conducted to educate them about the keyboard layout. For actual evaluation purposes, a total of 45 phrases were selected at random from the MacKenzie and Soukoreff [88] texts and were grouped into three sets of 15 each (i.e. Phrase_Set_1 - short, Phrase_Set_2 - medium and Phrase_Set_3 - long). The short phrase group were at most 23 characters in length, medium phrases had less than 32 characters, and long phrases had more than 32 characters. During the experiment, phrases were displayed to the users on a desktop screen. The participants were requested to input text with their dominant hand while they wore the smartwatch on their non-dominant hand. There were two scenarios: sitting and walking inside the lab. The participants were asked to perform three sessions in both the testing scenarios, each session included two trials and they had to write 15 phrases (we chose 5 phrases randomly from each of our three existing phrase-sets) in each trial. We conducted the sitting environment tests, first for all the participants and it spanned across three days. A gap of two hours was strictly maintained between each session. After the completion of the sitting environment tests, the walking tests were conducted. These tests also spanned three days, and the participants had to type the same phrases as they had typed in the sitting environment. Participants were instructed to correct any errors they made during the typing session, but a constraint was imposed upon them. The constraint being that they were allowed to correct a mistake, only if they observed it at the time of committing the error. So, if they typed along and realized later that they had made a mistake in a previous word or the beginning of the word they were typing, they weren't allowed to rectify the mistake. Note that, we also followed the same experimental setup protocol for the other keyboards.

5.1.4.2 Text Input Performance

In the experiment, we recorded the corrected WPM measure and not the raw WPM measure as it would have included incorrectly typed characters during the calculation.

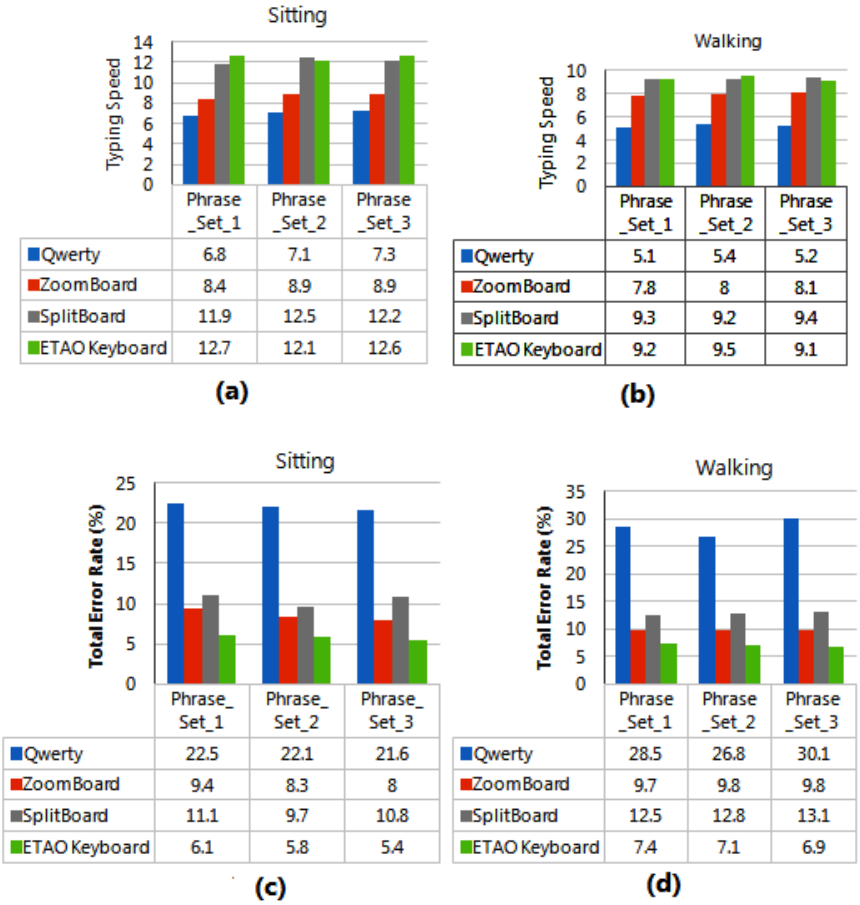


Figure 5.3: Typing speed in words per minute: (a) during sitting (b) during walking; Total Error Rate: (c) during sitting (d) during walking.

To analyse the WPM and error-rate of different keyboards, we used repeated measures ANOVA and a pairwise comparison. As the Qwerty keyboard showed significantly higher error rates, we removed it from our ANOVA measures and considered only three remaining keyboards (i.e. ZoomBoard [10], SplitBoard [14] and ETAO Keyboard). This removal was important as the inclusion of Qwerty would have given unnecessarily given higher values in ANOVA and would not have helped in the proper analysis of results.

There was a major effect of the keyboards on WPM in sitting experiment ($F(2,18) = 60$, $p < 0.05$) and during a walk ($F(2,18) = 47.227$, $p < 0.05$). Using ETAO keyboard, participants were able to enter the phrases with 12.46 WPM (SD = 0.71) in sitting

situation and 9.36 WPM (SD = 0.59) in walking scenario respectively. The Figure 5.3(a) & (b) represent the text entry speed of different keyboards during sitting and walking. The ETAO was faster than the Qwerty ($p < 0.001$) and the ZoomBoard ($p < 0.05$). However, there was no significant statistical difference between ETAO keyboard and SplitBoard when we compared their text entry rates ($p = 0.05$).

A similar effect was also seen on the Total Error Rates(TER) during the sitting experiment (($F(2,18) = 72.18$, $p < 0.05$) and while walking ($F(3,27) = 80$, $p < 0.05$). As can be seen from Figure 5.3(c) & (d), the Qwerty caused the most number of errors when compared with the remaining keyboards. This was observed in both the sitting and walking conditions. The ZoomBoard had an error rate lower than that of SplitBoard and Qwerty ($p < 0.05$). In our experiment, ETAO keyboard was the most accurate keyboard to enter text efficiently in both sitting and walking scenarios.

The average UER (Uncorrected Error Rates) were 0.74% for the Qwerty and was 0.61% for the SplitBoard. The ZoomBoard had a UER of 0.48% while the ETAO keyboard had the least UER at 0.41%.

The participants were asked to order the keyboards on the basis of 'learning-time' involved. Most participants favored the Qwerty citing that it had the most common key layout, and hence, it was easy to guess the location of a character, even though it was cumbersome to type text with a small key size. The next favorite keyboard was ZoomBoard as its interface was similar to Qwerty with zoom-in and zoom out features. The SplitBoard was their third preferred choice as it is a scrolling Qwerty keyboard and immediate learning is possible. Next they voted for ETAO keyboard. Participants suggested that it took some time getting used to the different layers involved, but after the initial learning effort, it was the easiest to type with. They also mentioned that after getting accustomed with our keyboard, they felt that ETAO keyboard achieved the best trade-off between error rate and typing speed.

5.1.5 Conclusion

We introduced ETAO keyboard, a text entry technique for smartwatches with tiny touchscreens. It allows a user to access eight most frequent English letters with a single tap while others including digits and special symbols by double tap. Here, we used two swipe gestures for 'delete' and inserting 'space'. This layout easily eliminates the 'fat-finger' problem by providing enough key-size and space between keys. The user requires few minutes of training to be accustomed to this keyboard. In this study, we didn't consider any dictionary-based features and focused only on the key entry efficiency of the keyboards. This is a definite shortcoming of this study. Nowadays, every modern touchscreen keyboard uses a language model. For more practically meaningful results, we would like to compare these keyboards after they are augmented with a dictionary-based text prediction feature.

5.2 Using Hall Effect Sensors for 3D space Text Entry on Smartwatches

5.2.1 Motivation

Smartphones became the most ubiquitous computing devices nowadays. Despite their high portability, it is not possible to provide almost instant access to digital services available in the smartphone to users as people typically carry these devices in pockets and bags. To address this, the concept of smartwatches had been proposed. In the present day context of wearable computing, smartwatches like Samsung Galaxy Gear S, LG G Watch, Motorola Moto 360, Apple Watch, etc. are one of the most commercially successful wearable devices. It allows users to access several applications (messaging, email, calendar, maps) running on smartphones directly from their wrists, without having to look at their phones. Although applications are instantly accessible on the watch, users face difficulties to reply immediately as there is normally no standard text entry

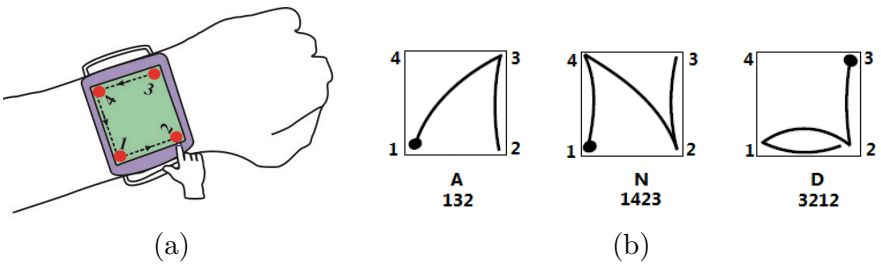
method on the same device. To give responses to text notifications on smartwatches, users have to use the voice communication built-in application like Google Now[1] on Android, Siri[2] on iOS. Text input using voice has certain limitations [89] like recognition of voice in noisy environments, eavesdropping on private information, etc.

Recently researchers have invested their efforts to fit virtual Qwerty keyboards either directly or with little modification on smartwatches [23-32]. However, on-screen keyboards require precious screen space, suffer from the fat finger and occlusion problems. It also demands visual search to find a character. So, our objective is can we design a text input technique which will not (a) require any screen space, (b) suffer from occlusion problem, and (c) need any visual search and (d) confine users within the limited touch area? To accomplish these criteria, in this chapter, we present hall effect sensors based text entry mechanism that effectively uses the 3D space around the smartwatch for entering alphanumeric characters.

5.2.2 Proposed Framework of Text Input Using Hall Effect Sensors

To avoid fat finger and occlusion problems during text entry on smartwatches, we develop hall effect sensors based text input mechanism that effectively uses the 3D space around the device for entering alphanumeric characters. For this purpose, four hall sensors are placed in four corners (marked as 1, 2, 3, and 4) of a watch and the user draws characters' gestures around the device using a magnet (may be ring or disk type) mounted on his finger (see Figure 5.4(a)). These hall sensors become active when a magnet comes into their sensing range. Our proposed technique adopts the EdgeWrite [47] mnemonic gesture set for alphanumeric input. In short, EdgeWrite is a unistroke method originally designed for stylus entry on PDAs by people with tremor. Here, we mapped each EdgeWrite letter to four corners. For example, using the author's labels for the corners, the corresponding corner-sequences of the letters 'A', 'N' and 'D' are '132', '1423' and

[1]https://www.google.com/landing/now/
[2]http://www.apple.com/in/ios/siri/



(a) (b)

Figure 5.4: (a) Positions of four hall effect sensors in four corners of a watch and user is trying to write 'C'. (b) Three EdgeWrite characters and their corner sequences. The dots mark the starting points.

Characters	Corner sequences	Characters	Corner sequences
A	132	N	1423
B	4121	O	34123
C	3412	P	1434
D	3212	Q	34323
E	13412	R	143
F	341	S	3421
G	34321	T	432
H	4132	U	4123
I	41	V	413
J	321	W	41323
K	3142	X	4231
L	412	Y	4232
M	14232	Z	4312
Numbers	**Corner sequences**	**Numbers**	**Corner sequences**
0	32143	5	34121
1	32	6	3121
2	43212	7	431
3	4321	8	34213
4	41232	9	3432

Figure 5.5: Alphanumeric characters and its corner sequences.

'3212' respectively, and it is shown in Figure 5.4(b). The Figure 5.5 represents the corner sequences of 26 alphabets and 10 digits. Note that, we modified the corner sequences of three characters ('E', 'K' and 'P') for user convenience. However, there are two more corner sequences, that is, '12' and '21' for spacebar and backspace key respectively. To enter any alphanumeric character, users move their magnet mounted finger over hall effect sensors following those corner sequences mentioned in Figure 5.5. If the drawn corner sequence matches with the previously stored corner sequence pattern, then the system recognizes the intended character.

This around device interaction based text entry method requires a solution to the segmentation problem since there is no 'stylus lift' event which is available in touchscreen

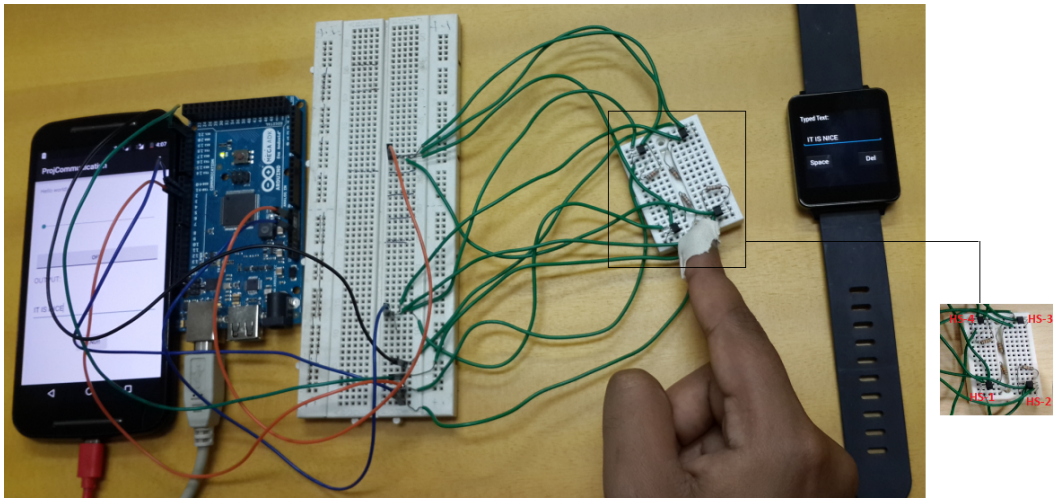


Figure 5.6: Prototype of our proposed text entry mechanism for smartwatches. Here, 'HS' (red color text in the rightmost image) represents hall effect sensor. Using this setup, user is trying to write 'IT IS NICE'.

interaction. Thus, we follow a Δ time-instant for segmenting two consecutive character gestures. If a user passes his finger over any two hall sensors within the Δ period, then our system understands that user is trying to follow the corner sequence pattern of a particular character or number. Otherwise, it recognizes the previous corner sequence and segmentation occurs, that is, the user is going to enter next alphabets or numbers. In our experiment, we empirically choose the value of Δ as 950 msec.

5.2.3 Implementation

To realize our proposed text input technique, we place four A3144 hall effect sensors in four corners of a mini breadboard (dimension: 47mm $\times$ 35mm $\times$ 8.5mm and it is almost equivalent to a smartwatch touchscreen display) and mount N54 grade, disk-shaped (10mm $\times$ 3mm in diameter and height respectively) neodymium magnet on the finger using a small strip of velcro. Hall sensors are connected to an Arduino micro-controller via an electrical circuit. On the other side, the Moto G (with Android OS version 5.1) smartphone is connected to the Arduino via a USB OTG cable and it is also paired with LG W100 smartwatch (with Android Wear version 4.4W) over Bluetooth. The complete

setup is shown in Figure 5.6. When a user brings his magnet mounted finger near to a hall sensor, then it becomes active and sends its value to the Arduino. Then Arduino recognizes the intended character/number by matching with the predefined patterns and transmits it to the phone's Android application. Finally, the entered character/number is transferred from the phone to the watch's application and appended to a text-field.

5.2.4 User Evaluation

This section presents the results of the experimental evaluation we conducted with participants. We first describe how we designed and conducted the experiments and then we report the results of our evaluation.

5.2.4.1 Method

To evaluate the feasibility of our proposed text input method, we performed some text typing tests and compared it with Qwerty layout available in Samsung Gear S. Five university students were recruited (3 male and 2 female), all aged between 20-28 (Mean = 24). The participants were primarily post-graduate students in our university's Information Technology department. None of them had any previous experience with smartwatches, but they are all well experienced with smartphones and accustomed to typing using phone's default Qwerty soft keyboard.

Before the beginning of the actual tests, a demo session was conducted to educate the participants about the hall sensors based text input mechanism. In the demo session, each participant was asked to type in their names, surnames, addresses and telephone numbers with the proposed technique. This was done to familiarize the participants with the system further, on a personal level. Following this practice session, each user spent almost 40 minutes for two sessions using the system to enter phrases, and finally answered a brief questionnaire and informal feedback. The second system was then tested in the same way. The evaluation was conducted in a calm lab environment.

For actual evaluation purposes, a total of 10 phrases were selected at random from the MacKenzie and Soukoreff [88] texts. This same phrase set was used by all participants in each system. During the test, phrases were displayed to the users on a desktop screen. Participants were able to rest whenever they wished, but were encouraged to rest between phrases rather than mid-phrase. During typing, they were allowed to correct any errors they made, but a constraint was imposed upon them. The constraint is that they were allowed to correct a mistake, only if they observed it at the time of committing the mistake. So, if they typed along and realized later that they had made an error in a previous word or the beginning of the word they were typing, they weren't permitted to rectify the mistake. One more typing constraint is that users were not allowed to use word-prediction, although it was available in Gear S smartwatch.

5.2.4.2 Results

In the experiment, text entry performance is measured in terms of WPM (words-per-minute) and TER (total error rate). Note that, we recorded the corrected WPM measure and not the raw WPM measure as it would have included incorrectly typed characters during the calculation. The WPM is calculated as $\left(\frac{\text{characters per minute}}{\text{5 characters per word}}\right)$ and TER as $\left(\frac{\text{INF + IF}}{\text{C + INF + IF}}\right)\times 100\%$; where INF is incorrect not fixed characters, IF is incorrect fixed characters and C is correct characters. The results for WPM and TER are shown in Figure 5.7(a) & (b). On average, participants entered the phrases with 5.78 WPM (SD=0.45) using the Qwerty keyboard and 3.9 WPM (SD=0.36) using the proposed text input method. A t-test shows that two technique had a significant effect on the WPM (p=0.02). The TER using the Qwerty keyboard was 22.12 (SD=3.43) and 6.4 (SD=2.62) for our proposed technique. This improvement in TER is significant (p=0.05). Therefore, our proposed technique provides acceptable typing speed with minimum error compare to Qwerty layout.

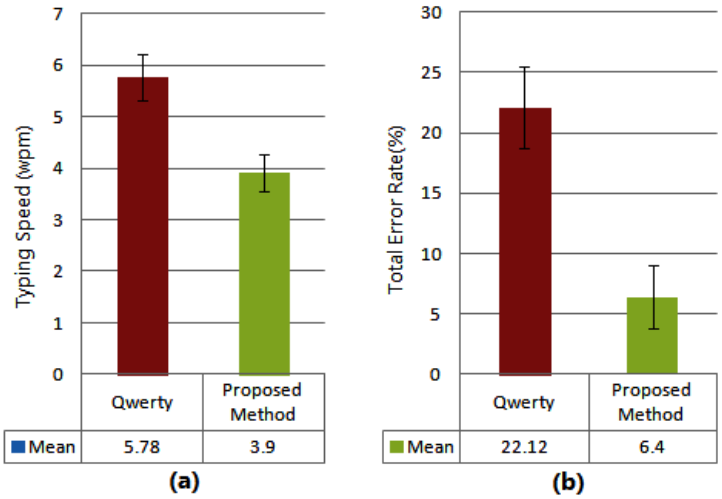


Figure 5.7: (a) The average WPM for the Qwerty and Proposed Method. (b) The average TER for the Qwerty and Proposed Method. In both figures the error bars show the standard deviation.

Table 5.1: Questionnaire results (mean, sd) for responses given on a Likert scale (1 = disagree strongly, 5 = agree strongly).

Statement	Qwerty	Proposed Text Input Method
Easy to use	3.2 (1.12)	**4.3** (**0.51**)
Fast to use	2.6 (0.34)	**4.4** (**0.33**)
Easy to learn	4.2 (0.12)	4.0 (0.24)
Improve with practice	3.3 (0.75)	**4.4** (**0.54**)
Felt in control	3.0 (0.44)	**3.8** (**0.38**)
Easy to undo mistake	4.1 (0.32)	3.9 (0.20)
Mental demand	2.5 (0.11)	4.4 (0.23)
Physical demand	2.4 (0.14)	3.9 (0.28)
Frustration	3.5 (0.18)	**2.2** (**0.08**)
Performance	3.1 (0.33)	**4.2** (**0.36**)

5.2.4.3 Questionnaire Results

After the end of each session, we asked participants to give their valuable feedback to a questionnaire comprised of ten statements on a 5-point Likert scale (1 = strongly disagree, 5 = strongly agree). Our proposed text input method was rated significantly higher than Qwerty keyboard. Majority (i.e. six out of ten) feedbacks were in favor of our developed system. Table 5.1 represents the list of statements, mean responses,

and significant differences. Moreover, users also reported few informal feedbacks. For example, one female user said that she got confused between character 'O' and number '0' most of the time. One male participant stated that he also did the same kind of mistake while writing 'K' and 'X'. Some users mentioned that they wanted to write punctuation symbols using this system.

5.2.5 Conclusion

In this chapter, we introduced the hall effect sensor based text input mechanism for smartwatches. This technique does not require any touchscreen space and visual search to find a character but demands little cognitive load. User study reported that proposed method can well balance between typing speed and error rate. This technique easily overcomes the 'fat finger' problem. Here, we did a small-scale user study in controlled environment, but in future (a) we will build a fully integrated system inside a watch and will investigate other usability aspects of our proposed technique (b) we are also planning to use proximity sensors instead of hall effect sensors to avoid the major concern of carrying an extra magnet on finger.

Chapter 6

Side Pressure-Based Input for Smartwatch Interaction

Smartwatches have gained a lot of public interest as one of the most popular wearable devices in recent times, but their diminutive touch screens mar the user experiences. The small screen of watch suffers from visual occlusion and the fat finger problem. To address these issues, we present 'PressTact' that extends interaction space beyond the watch surface to the sides of the device. It augments smartwatches with four pressure sensors - two sensors on the left side of a watch and another two on the right side. It enables users to input different levels of pressure that can be used for bi-directional navigation (zooming, scrolling, rotation) on smartwatches. In this chapter, we explore the pressure event based input vocabulary set. Our preliminary user study shows that participants can input different pressure levels (light press, medium press, and strong press) in discrete and continuous mode with an acceptable accuracy. Finally, we develop several example applications to illustrate the potential of the proposed technique.

The rest of the chapter is structured as follows. Section 6.1 describes the motivation of our work. Section 6.2 presents PressTact prototype and Section 6.3 defines pressure event vocabulary for interactions. Section 6.4 details user study of PressTact input

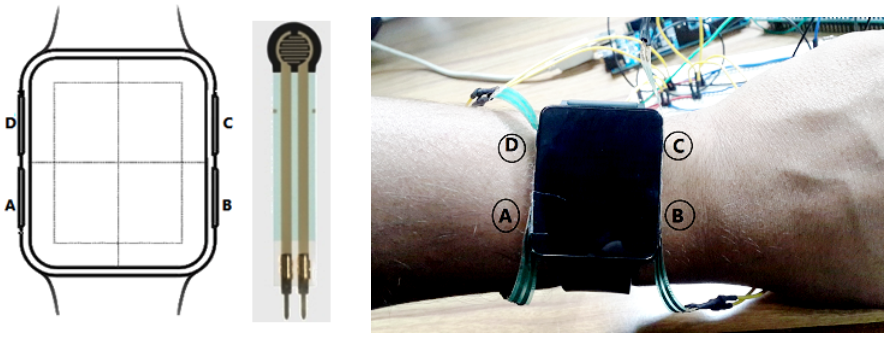
vocabulary. We demonstrate several example applications in Section 6.5 and finally, Section 6.6 concludes this chapter.

6.1 Motivation

Smartwatches have become increasingly popular particularly at the consumer level as an emerging computational form factor due to the unprecedented success of miniaturization technology. The primary input methods for the commercially available smartwatches are touchscreen and the physical buttons located on its side. However, its small touchscreen limits available interactive surface and lacks tactile feedback. Simply making smartwatches larger to provide more space for interaction is not a feasible option as this would make them more obtrusive.

In this context, we present PressTact that extends interaction space beyond the watch surface to the sides of the device. It augments smartwatches with four pressure sensors - two sensors on the left side of a watch and another two on the right side. It supports users to input different levels of pressure in discrete and continuous mode that can be mapped to different actions in a variety of applications such as zoom-in and zoom-out a picture, rotating an image, scrolling a list at variable speed, select and edit text. Our approach is related to the work done by Spelmezan et al. [90] where they installed two continuous pressure sensors on the one side of a smartphone to detect squeeze based inputs. In our case, we are particularly interested in exploring this side pressure sensors based interaction for smartwatches as this kind of wearable device has a fixed position on the wrist and it is less likely to be misplaced. Here, we report initial results from a study on how users can comfortably input different levels of single sided (pointing type) pressure and two-sided (grasping type) pressure in wristwatch context. We also consider the chorded keyboard[1] to take few design decisions in our experiment. Further, the pressure sensor has several advantages - it requires very less power to operate, it

[1] https://en.wikipedia.org/wiki/Chorded_keyboard



(a) (b)

Figure 6.1: (a) Interlink Electronics 400FSR pressure sensor positions around a smartwatch. The positions are marked as 'A', 'B', 'C' and 'D'. (b) The experimental PressTact prototype with LG W100 smartwatch.

provides inexpensive input interface, users can rapidly switch between different pressure modes and it is thin enough for wearable devices. Because of its thin size, pressure sensors can be easily integrated into the smartwatches without significantly changing device's form-factor.

6.2 The PressTact Prototype

The PressTact prototype, shown in Figure 6.1, lets users apply pressure onto the bottom-left sensor (A), the bottom-right sensor (B), the top-right sensor (C) and the top-left sensor (D) individually or in a combination of any two sensors simultaneously. Our prototype has four primary components: LG W100 smartwatch running on Android Wear, Moto G Android smartphone, four force-sensing resistors (Interlink Electronics FSR 400), and Arduino Mega ADK. The FSRs are attached to the body of a smartwatch in the configuration shown in Figure 6.1. Each FSR has a round sensing area of 7.62 mm diameter and works like a variable resistor whose resistance changes when a force or pressure is applied. The FSRs don't have a linear resistance vs. force characteristic. In order to linearize pressure input, an op-amp based current-to-voltage converter circuit was used as recommended by [91]. The pressure sensors are connected to the Arduino

Table 6.1: PressTact input vocabulary consisting of 30 pressure-events

Pressure Levels	Combination of Force-Sensing Resistors									
	A	B	C	D	AB	CD	AC	BD	AD	BC
Light Press	#1	#4	#7	#10	#13	#16	#19	#22	#25	#28
Medium Press	#2	#5	#8	#11	#14	#17	#20	#23	#26	#29
Strong Press	#3	#6	#9	#12	#15	#18	#21	#24	#27	#30

micro-controller via an electrical circuit. The Arduino samples pressure sensor data at 50 Hz and 10 bit resolution and sends it to Moto G phone using a HC-05 serial port bluetooth module. The phone processes the sensor data, runs an algorithm for recognizing different pressure events, and transfers the detected pressure input to a paired smartwatch.

6.3 Pressure Event Vocabulary for Interaction

Our objective is to propose different pressure events that could be combined to support a richer interaction and at the same time, they should be unambiguous to recognize. We thus consider that users can apply pressure on each sensor individually or in a combination of any two sensors simultaneously. It results in total ten types of FSR combinations - Press (A), Press (B), Press (C), Press (D), Press (A, B), Press (C, D), Press (A, C), Press (B, D), Press (A, D) and Press (B, C). Further, users can actuate FSRs at different levels - light press, medium press, and strong press. To recognize three discrete pressure levels, we take the average of 500 msec sensor data (F_{avg}) each time and check the following conditions:

- Light press if $0.5 \leq F_{avg} < 3$
- Medium press if $3 \leq F_{avg} < 5.5$
- Strong press if $5.5 \leq F_{avg} < 10$

Here, we measure input force in Newton, and a user can apply approximate 10N force

at the maximum. We use 0.5N thresholding to avoid unintended pressure input. Ultimately our input vocabulary consists of 30 (10 combinations of FSRs $\times$ 3 pressure levels) different pressure events which are represented in Table 6.1.

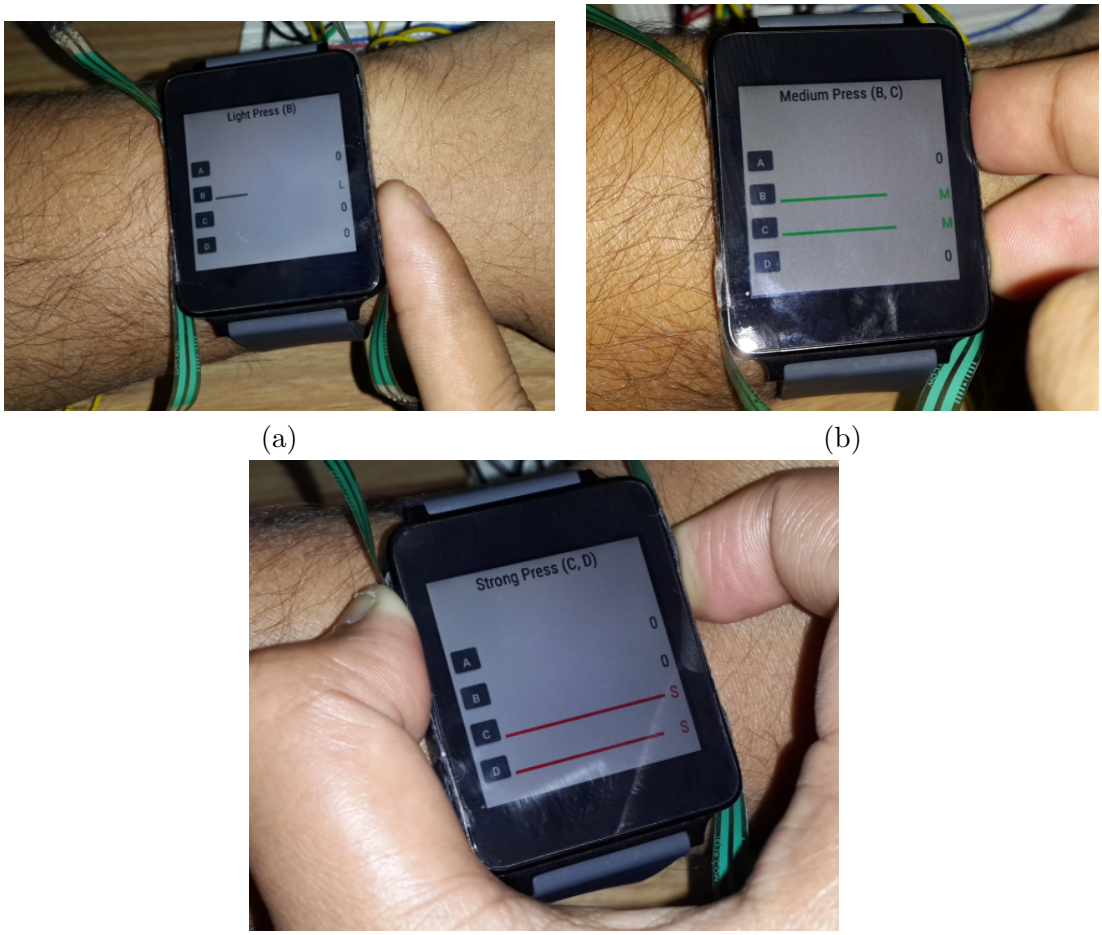
6.4 User Study of PressTact Input Vocabulary

In order to evaluate the user's ability to trigger each of the thirty pressure events, we performed a pilot study in which the participants were asked to selectively input different levels of pressure on demand.

6.4.1 Method

We developed one Android application where users have to input target pressure according to the instruction, and they can visualize the corresponding sensor's pressure level through a progress-bar and a text-box situated at the right most side as shown in Figure 6.2. Further, our system also provides vibration feedback. Users will feel light, medium, and strong intensity of vibration based on the different levels of input pressure. We designed targeting test in two modes: discrete and continuous. For the discrete mode test, the participants were asked to achieve the target pressure at the first press and then release the sensor(s). For the continuous mode test, after attaining the target pressure at the first press, they have to maintain that target pressure for three seconds and then release the sensor(s). Note that, users will feel 500 msec of vibration for the discrete mode and 3 sec of vibration for the continuous mode, as soon as they achieved the target pressure. When a user successfully completes a trial, a new target task is randomly generated in the application interface. During the experiment, we logged the test type, trial number, correctness of the trials, and completion time for each trial.

Six unpaid volunteers (2 females, aged 24-32, average 28.6 years) from our Institution took part in this study. All participants were right-handed, and none of them had any prior experience with the smartwatches. They wore our PressTact prototype on their



(a) (b)

(c)

Figure 6.2: User study application interface: (a) light pressure on sensor B (b) simultaneous medium pressure on sensors B and C (c) strong pressure on both C and D at the same time.

left wrist and performed different pressure events with the right hand. As per our instruction, they used their index finger to apply pressure to the individual sensor (i.e. A, B, C, and D). They took the help of their index and middle fingers to press two sensors located at the same side (i.e. BC, AD) and used both the thumb and the index finger to press sensors on the opposite sides (i.e. AB, CD, AC, and BD). This user study was performed in a lab environment, and users were in seated positions. Before the beginning of the experiment, a demo session was conducted to make them familiar with the software interface and the interaction mechanism. They also practiced before

Table 6.2: Users' success rate(%) for performing different levels of pressure in discrete and continuous conditions.

FSRs	Light Press		Medium Press		Strong Press	
	Discrete	Continuous	Discrete	Continuous	Discrete	Continuous
A	86.42	81.29	85.94	78.63	77.12	75.36
B	91.88	88.29	90.18	87.38	86.44	83.56
C	91.54	89.61	92.04	89.59	88.23	85.33
D	83.74	82.88	82.58	82.11	76.51	75.81
AB	98.46	98.54	97.89	96.53	98.12	96.75
CD	97.83	98.06	98.24	95.49	96.82	95.22
AC	85.93	82.27	81.41	76.61	78.13	72.85
BD	83.81	82.66	82.93	82.06	77.08	71.01
BC	88.84	88.03	85.63	84.93	80.31	77.72
AD	78.61	74.34	75.08	72.66	73.81	72.96

starting the actual test. Each user performed 30 pressure-events $\times$ 10 repetitions $\times$ 2 test modes = 600 trials and took approximately 54 minutes (SD = 4.6) to complete the test. Lastly, we interviewed participants for informal feedback.

6.4.2 Results and Discussion

Table 6.2 represents users' average success rate to input different levels of pressure (light press, medium press and strong press) in discrete and continuous conditions.

From this table, we observe that the pressure sensors A, B, C, and D provide 80.79%, 87.96%, 89.39% and 80.61% accuracy respectively. Users achieve significantly better performance from sensors B and C compare to sensors A and D while they are applying pressure on each FSR. The reason is that users have comfortable index finger position when they put pressure on B and C. If they want to input pressure on A and D, then they have to rotate index finger by 180° and it is quite difficult to maintain different pressure levels in this finger position.

While we consider any two FSRs jointly, then AB and CD combinations provide the best success rate, that is, overall above 96%. In both cases, the position of sensors is entirely opposite to each other (i.e. left and right side); as a result users can easily

maintain different pressure levels with their thumb and index fingers.

The combination BC gives the second best performance ($\sim$ 84.25%) because users are able to put pressure on B and C simultaneously just by placing their middle finger and index finger on the respective sensors. While users apply pressure from the right side, the placement of the watch body shifts toward left. Although this position shifting occurs in small scale, it has significant contribution in errors during the experiment.

The next best performance comes from the AC and BD combinations, and it is almost 79.51%. Here users face difficulty to input different pressures as the sensors are diagonally opposite to each other. Finally, we get the most erroneous performance in our study from AD combination and it is $\sim$74.57% on an average. The reason behind this poor performance is that users can't maintain balanced pressures on both the sensors from the left side using their index and middle fingers.

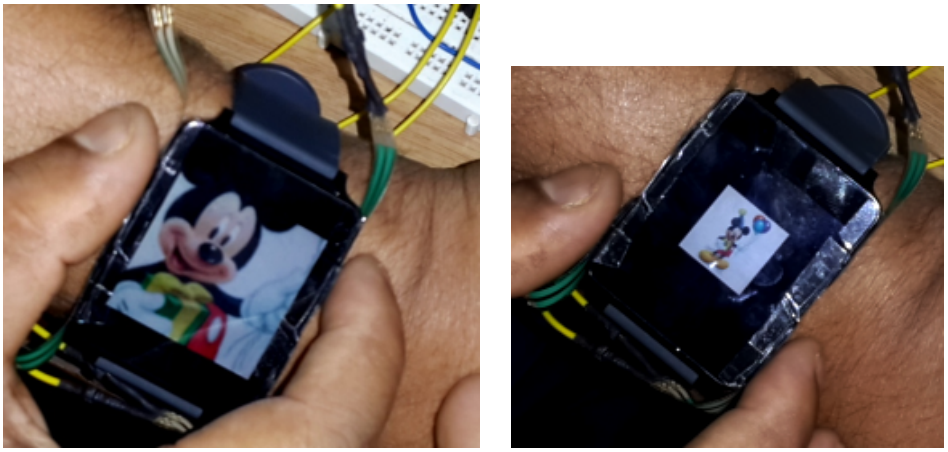
In our experiment, the average completion time for discrete trial was 1.42 sec (SD = 0.18) and it was 1.75 sec (SD = 0.26) for continuous trial.

In feedback session, most of the users mentioned that side pressure sensor based smartwatch input is easy to learn, easy to press, and it is a promising input modality for future smartwatches. They were able to control whole pressure event vocabulary in discrete and continuous mode with an average success rate of 85.16%. In fact, most of the participants felt more natural and pleasant with the discrete mode of pressure input.

6.5 Application Example

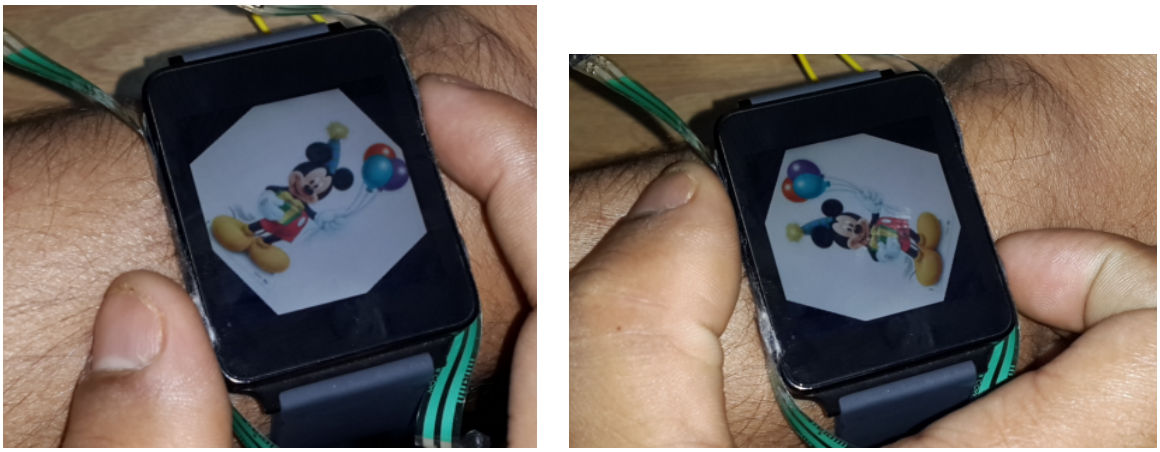
To show the feasibility of our proposed pressure event based input vocabulary, we developed two applications - photo gallery app and number entry app.

In the photo gallery app, users can zoom-in and zoom out an image by applying pressure on CD and AB, respectively (see Figure 6.3). They can control zoom-in/out rate by applying different levels of pressure. For example, light pressure corresponds to slower zoom-in/out, while strong pressure provides faster zoom-in/out. For continuous



(a) (b)

Figure 6.3: Photo gallery app: (a) zoom-in an image by pressing CD simultaneously (b) zoom-out an image by pressing AB simultaneously.



(a) (b)

Figure 6.4: Photo gallery app: (a) clockwise image rotation by pressing AC simultaneously (b) anti-clockwise image rotation by pressing BD simultaneously.

zoom-in/out, they have to apply a certain level of pressure continuously. Similarly, users have to press AC and BD for rotating an image in a clockwise and anti-clockwise direction respectively, and it is shown in Figure 6.4.

In number entry app, users can control the caret inside the text box quickly and precisely using the pressure event vocabulary set. To move the cursor one digit left, press C lightly, and to move one digit right, just light press D. The cursor movement

Figure 6.5: Number entry app: press C to move caret toward left.

example is shown in Figure 6.5. After fixing the caret at a particular position, they can perform 'delete' operation using AB combination.

6.6 Conclusion

In this chapter, we investigated the use of side pressure sensors on smartwatch device for occlusion-free interactions. We presented a working prototype of PressTact and defined a rich vocabulary of pressure event that can be mapped to many different actions in a variety of applications. Through a preliminary user study, we showed that the idea is feasible to use. For the future work, we plan to conduct more extensive user studies in a real life setting. We will also compare this analog pressure input technique with the buttons and dials of the existing watches.

Chapter 7

Conclusion and Future Research

The main contribution of this thesis is the development of a number of novel interaction techniques and user interface concepts for mobile devices like smartphones and wearable devices like smartwatches. We observe that the development of mobile and wearable user interfaces is closely coupled with the evolution of their sensory capabilities. Our goal is to make a better use of the available sensing capabilities of these devices and provide suggestions on the types of sensor technologies that could be added to the future devices in order to enrich their input expressiveness. In this dissertation, we have explored our research in three promising areas: context-awareness, text input interfaces and input beyond touch. This chapter summarizes the significant contributions of our work and future scope of extending the research.

7.1 Contributions

The major contributions of our research work can be summarized as follows.

Chapter 3 presents phone's context-aware mobile interaction. Here, we propose 'SurfaceSense', an approach to identify different phone's placements (i.e. context) using its built-in sensors such as an accelerometer, gyroscope, magnetometer, microphone, and a proximity sensor. The contributions of our approach are: (1) don't need any external

hardware like [48] [36] [49] (2) faster and energy efficient compare to [50]- [53] (3) able to identify 13 different phone's placements with 91.75% accuracy (4) showed how different applications can be configured dynamically to enhance mobile interaction on the basis of phone's current placement.

In Chapter 4, we propose 'MagiText' which provides the text entry space beyond the physical boundaries of a device. It overcomes the problems of virtual keyboard based typing, for example, long visual search time and fat-finger issue. This approach uses phone's magnetometer sensor to recognize 3D space handwritten character gestures. The key idea is to influence the magnetic sensor by writing character gestures in front of the device using a properly shaped magnet taken in hand. The movement of this magnet changes the magnetic flux pattern, and it is sensed and registered by the magnetometer sensor. Then, we classify this flux pattern using machine learning classifier and identify intended character. The contributions of 'MagiText' are: (1) don't require any external sensory inputs such as infrared distance sensor [3] [4], RGB camera [5] [6], depth camera [7], electric field sensing [8] and magnetometer [55] [56] (2) earlier work [55], [56], [57]- [61] didn't explore the perspective of text entry mechanism using phone's buit-in magnetomter sensor, that is really a challenging task.

In Chapter 5, we develop two text entry approaches: 'ETAO keyboard' and '3D typing using Hall effect sensors'. In ETAO keyboard, a user can select most frequent characters with a single tap and remaining characters, numbers and symbols with two taps. It provides a good trade-off between typing speed and error rate with respect to [10]- [17]. With 3D typing, users can write text just by drawing character gesture above the watch face. It is basically a touchless text input approach for smartwatches.

Chapter 6 presents 'PressTact' which augments smartwatches with four Force Sensing Resistors (FSR) - two sensors on the left side of a watch and another two on the right side. It expands the smartwatch's interaction space beyond its touch screen input and overcomes visual occlusion problem. We design a pressure event vocabulary set that

can be used for bi-directional navigation (zooming, scrolling, rotation) on watches. Our preliminary user study confirms the feasibility of the pressure sensors based smartwatch interaction technique and also shows that participants can input different pressure levels (light press, medium press, strong press) with an acceptable accuracy. The main advantage of our technique compared to previous work [64] [18] [67] [20] [69] [22] - [73] is that 'PressTact' requires minimal hardware instrumentation without effecting the smartwatch form-factor and it has immediate feasibility to fit into square and round faced smartwatches.

7.2 Future Research Directions

In this dissertation, different interaction techniques suitable for smartphones and smartwatches have been presented from both design and user study perspective. Further, we can further explore these interactions in a broader perspective. In the case of phone's context sensing based mobile interaction, user's context can be combined with phone's context to develop a number of novel applications. In ADI based text input on a smartphone, we only showed our preliminary results of character gesture recognition. In future, we may try to detect starting and ending point of input character gesture automatically. On the other hand, the hall effect sensor based smartwatch text entry technique can be redesigned with IR proximity sensors. Then we don't need any external magnets. Finally, we can add postural state of the watch with the proposed pressure sensors based interaction to enlarge the interaction space.

Publications

Conference

- Rajkumar Darbar, Prasanta Kr. Sen, Debasis Samanta, "PressTact: Side Pressure-Based Input for Smartwatch Interaction", In 34^{th} ACM Conference on Human Factors in Computing Systems (CHI 2016), 7 - 12 May, 2016, San Jose, CA, USA.

- Rajkumar Darbar, Debasis Samanta, "SurfaceSense: Smartphone Can Recognize Where It Is Kept", In 7^{th} ACM International conference on Human Computer Interaction (IndiaHCI 2015), 17 - 19 Dec, 2015, IIT Guwahati, India.

- Rajkumar Darbar, Punyashlok Dash, Debasis Samanta, "ETAO Keyboard: Text Input Technique on Smartwatches", In 7^{th} IEEE International conference on Intelligent Human Computer Interaction (IHCI 2015), 14 - 16 Dec, 2015, IIIT Allahabad, India.

- Rajkumar Darbar, Prasanta Kr. Sen, Punyashlok Dash, Debasis Samanta, "Using Hall Effect Sensors for 3D space Text Entry on Smartwatches", In 7^{th} IEEE International conference on Intelligent Human Computer Interaction (IHCI 2015), 14 - 16 Dec, 2015, IIIT Allahabad, India.

- Rajkumar Darbar, Debasis Samanta, "MagiText: Around Device Magnetic Interaction for 3D Space Text Entry in Smartphone", In 3^{rd} IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT 2015), 10 - 11 July 2015, IIIT-Bangalore, India.

References

[1] S. Agrawal, I. Constandache, S. Gaonkar, R. R. Choudhury, K. Caves, and F. DeRuyter, "Using mobile phones to write in air." *In the Proc. of ACM International Conference on Mobile Systems, Applications, and Services (MobiSys '11)*, pp. 15–28., 2011.

[2] T. Deselaers, D. Keysers, J. Hosang, and H. A. Rowley, "Gyropen: Gyroscopes for pen-input with mobile phones." *In the Proc. of IEEE Transactions on Human-Machine Systems*, pp. 1–9, 2014.

[3] A. Butler, S. Izadi, and S. Hodges, "Sidesight: multi-touch interaction around small devices." *In the Proc. of ACM Symposium on User Interface Software & Technology (UIST '08)*, pp. 201– 204, 2008.

[4] S. Kratz and M. Rohs, "Hoverflow: Expanding the design space of around-device interaction." *In the Proc. of ACM International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '09)*, 2009.

[5] J. Song, G. Soros, F. Pece, S. R. Fanello, S. Izadi, C. Keskin, and O. Hilliges, "Inair gestures around unmodified mobile devices." *In the Proc. of ACM Symposium on User Interface Software & Technology (UIST '14)*, pp. 319 – 329, 2014.

[6] T. Niikura, Y. Hirobe, A. Cassinelli, Y. Watanabe, T. Komuro, and M. Ishikawa, "In-air typing interface for mobile devices with vibration feedback." *In the Proc. of ACM International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '10).*

[7] S. Kratz, M. Rohs, D. Guse, J. Muller, G. Bailly, and M. Nischt, "Palmspace: Continuous around-device gestures vs. multitouch for 3d rotation tasks on mobile devices." *In the Proc. of ACM International Conference on Advanced Visual Interfaces (AVI '12)*, pp. 181 – 188.

[8] M. L. Goc, S. Taylor, S. Izadi, and C. Keskin, "A lowcost transparent electric field sensor for 3d interaction on mobile devices." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '14)*, pp. 3167–3170.

[9] H. Ketabdar, M. Roshandel, and K. A. Yuksel, "Magiwrite: Towards touchless digit entry using 3d space around mobile devices." *In the Proc. of ACM International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '10)*, pp. 443 – 446.

[10] S. Oney, C. Harrison, A. Ogan, and J. Wiese, "Zoomboard: A diminutive qwerty soft keyboard using iterative zooming for ultra-small devices." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '13)*, pp. 2799 – 2802.

[11] X. Chen, T. Grossman, and G. Fitzmaurice, "Swipeboard: A text entry technique for ultra-small interfaces that supports novice to expert transitions." *In the Proc. of ACM Symposium on User Interface Software & Technology (UIST '14)*, pp. 615 – 620.

[12] H. Cho, M. Kim, and K. Seo, "A text entry technique for wrist-worn watches with tiny touchscreens." *In the Proc. of ACM Symposium on User Interface Software & Technology (UIST '14)*, pp. 79–80.

[13] M. D. Dunlop, A. Komninos, and N. Durga, "Towards high quality text entry on smartwatches." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '14)*, pp. 2365 – 2370.

[14] J. Hong, S. Heo, P. Isokoski, and G. Lee, "Splitboard: A simple split soft keyboard for wristwatch-sized touch screens." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '15)*, pp. 1233 – 1236.

[15] F. Poirier and M. Belatar, "Uniwatch - some approaches derived from uniglyph to allow text input on tiny devices such as connected watches." *In Proc. of HCI International 2015*, pp. 554 – 562.

[16] J. M. Cha, E. Choi, and J. Lim, "Virtual sliding qwerty: A new text entry method for smartwatches using tap-n-drag." *In Proc. of Applied Ergonomics*, vol. 51, pp. 263 – 272, 2015.

[17] M. Funk, A. Sahami, N. Henze, and A. Schmidt, "Using a touch-sensitive wristband for text entry on smart watches." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '14)*, pp. 2305 – 2310.

[18] H. Xia, T. Grossman, and G. Fitzmaurice, "Nanostylus: Enhancing input on ultra-small displays with a finger-mounted stylus." *In the Proc. of ACM Symposium on User Interface Software & Technology (UIST '15)*, pp. 447–456.

[19] R. Xiao, G. Laput, and C. Harrison, "Expanding the input expressivity of smartwatches with mechanical pan, twist, tilt and click." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '14)*, pp. 193–196.

[20] S. T. Perrault, E. Lecolinet, J. Eagan, and Y. Guiard, "Watchit: simple gestures and eyes-free interaction for wristwatches and bracelets." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '13)*, pp. 1451–1460.

[21] C. Harrison and S. E. Hudson, "Abracadabra: wireless, high-precision, and unpowered finger input for very small mobile devices." *In the Proc. of ACM Symposium on User Interface Software & Technology (UIST '09)*, pp. 121–124.

[22] G. Laput, R. Xiao, X. A. Chen, S. E. Hudson, and C. Harrison, "Skin buttons: cheap, small, low-powered and clickable fixed-icon laser projectors." *In the Proc. of ACM Symposium on User Interface Software & Technology (UIST '14)*, pp. 389–394.

[23] M. Ogata and M. Imai, "Skinwatch: Skin gesture interaction for smart watch." *In the Proc. of ACM International Conference on Augmented Human (AH '15)*, pp. 21–24.

[24] W.-H. Chen, "Blowatch: Blowable and hands-free interaction for smartwatches." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '15)*, pp. 103–108.

[25] F. Kerber, P. Lessel, and A. Kruger, "Same-side hand interactions with arm-placed devices using emg." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '15)*, pp. 1367–1372.

[26] M. Azizyan, I. Constandache, and R. R. Choudhury, "Surroundsense: mobile phone localization via ambience fingerprinting." *In the Proc. of ACM International Conference on Mobile Computing and Networking (MobiCom '09)*, pp. 261–272.

[27] H. Lu, W. Pan, N. D. Lane, T. Choudhury, and A. T. Campbell, "Soundsense: scalable sound sensing for people-centric applications on mobile phones." *In Proc. of the 7th International Conference on Mobile Systems, Applications, and Services (MobiSys '09)*, pp. 165–178, 2009.

[28] S. Hemminki, P. Nurmi, and S. Tarkoma, "Accelerometer-based transportation mode detection on smartphones." *In Proc. of the 11th ACM Conference on Embedded Networked Sensor Systems (SenSys '13)*, 2013.

[29] J. Yang, "Toward physical activity diary: Motion recognition using simple acceleration features with mobile phones." *In Proc. of the 1st ACM International Workshop on Interactive Multimedia for Consumer Electronics (IMCE '09)*, pp. 1–10, 2009.

[30] M. Rossi, S. Feese, O. Amft, N. Braune, S. Martis, and G. Troster, "Ambientsense: A real-time ambient sound recognition system for smartphones." *In Proc. of the International Workshop on the Impact of Human Mobility in Pervasive Systems and Applications (PerMoby '13)*, pp. 230–235, 2013.

[31] F. Bert, M. Giacometti, M. R. Gualano, and R. Siliquini, "Smartphones and health promotion: A review of the evidence," *Journal of Medical Systems*, vol. 38:9995, pp. 1–11, 2014.

[32] M. Weiser, "The computer for the 21st century." *Scientific American, 265(3)*, pp. 66 – 75, September 1991.

[33] A. Farmer, "Wearables sector grows as smartwatches increase in popularity," *In Consumer, Digital, Media & Technology, Editor's picks, Front Page, Technology, Technology & Telecoms, UK*, 2016.

[34] A. Meola, "The smartwatch market just received an encouraging sign," *In Business Insider Intelligence*, 2016.

[35] G. E. Moore, "Cramming more components onto integrated circuits." *Electronics*, vol. 38(8), April 19, 1965.

[36] J. Wiese, T. S. Saponas, and A. B. Brush, "Phoneprioception: enabling mobile phones to infer where they are kept." *In Proc. of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*, pp. 2157–2166, 2013.

[37] K. Kunze and P. Lukowicz., "Symbolic object localization through active sampling of acceleration and sound signatures." *In Proc. of the 9th International Conference on Ubiquitous Computing (UbiComp '07)*, pp. 163–180, 2007.

[38] R. Wimmer and S. Boring, "Handsense - discriminating different ways of grasping and holding a tangible user interface," *In the Proc. of ACM International Conference on Tangible and Embedded Interaction (TEI '09)*, pp. 359 – 362, 2009.

[39] M. Goel, J. O. Wobbrock, and S. N. Patel, "Gripsense: Using built-in sensors to detect hand posture and pressure on commodity mobile phones," *In the Proc. of ACM Symposium on User Interface Software & Technology (UIST '12)*, pp. 545 – 554.

[40] E. Freeman, S. Brewster, and V. Lantz, "Tactile feedback for above-device gesture interfaces: Adding touch to touchless interactions," *In the Proc. of ACM International Conference on Multimodal Interaction (ICMI' 14)*, pp. 419 – 426, 2014.

[41] ——, "Towards usable and acceptable above-device interactions," *In the Proc. of ACM International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '14)*, pp. 459 – 464, 2014.

[42] R. Murmuria, J. Medsger, A. Stavrou, and J. M. Voas, "Mobile application and device power usage measurements," *In the Proc. of IEEE International Conference on Software Security and Reliability (SERE '12)*, pp. 147 – 156, 2012.

[43] L. Ardito, G. Procaccianti, M. Torchiano, and G. Migliore, "Profiling power consumption on mobile devices," *In the Proc. of IEEE International Conference on Smart Grids, Green Communications and IT Energy-aware Technologies (ENERGY '13)*, pp. 101 – 106, 2013.

[44] N. Savio and J. Braiterman, "Design sketch: The context of mobile interaction," *In the Proc. of ACM International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '07)*, 2007.

[45] A. K. Dey and J. Hakkila, "Context-awareness and mobile devices," *In IGI Global Mobile Computing: Concepts, Methodologies, Tools, and Applications*, 2009.

[46] F. Pollmann, D. Wenig, and R. Malaka, "Hoverzoom: making on-screen keyboards more accessible." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '14)*, pp. 1261–1266.

[47] J. Wobbrock and B. Myers, "Trackball text entry for people with motor impairments." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '06)*, pp. 479–488.

[48] C. Harrison and S. Hudson, "Lightweight material detection for placement-aware mobile computing." *In Proc. of the 21st Annual ACM Symposium on User Interface Software and Technology (UIST ' 08)*, pp. 279–282, 2008.

[49] F. Wahl and O. Amft., "Personalised phone placement recognition in daily life using rfid tagging." *In Proc. of the 1st IEEE Symposium on Activity and Context Modeling and Recognition*, pp. 19–26, 2014.

[50] J. Cho, I. Hwang, and S. Oh., "Vibration-based surface recognition for smartphones." *In Proc. of IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA '12)*, pp. 459–464, 2012.

[51] S. Hwang and K. Y. Wohn, "Vibrotactor: low-cost placement-aware technique using vibration echoes on mobile devices." *In Proc. of the International Conference on Intelligent User Interfaces (IUI '13)*, pp. 73–74, 2013.

[52] J. Yang, E. Munguia-Tapia, and S. Gibbs, "Efficient in-pocket detection with mobile phones." *In Proc. of the ACM Conference on Pervasive and Ubiquitous Computing (UbiComp '13)*, pp. 31–34, 2013.

[53] I. Diaconita, A. Reinhardt, F. Englert, D. Christin, and R. Steinmetz, "Do you hear what i hear? using acoustic probing to detect smartphone locations." *In IEEE Symposium on Activity and Context Modeling and Recognition*, pp. 1–9, 2014.

[54] D. Kim, J. Lee, H. Lim, J. Seo, and B. Kang, "Efficient dynamic time warping for 3d handwriting recognition using gyroscope equipped smartphones." *In Expert Systems with Applications*, vol. 41, pp. 5180 – 5189, 2014.

[55] H. Ketabdar, K. A. Yuksel, and M. Roshandel, "Magitact: Interaction with mobile devices based on compass (magnetic) sensor." *In the Proc. of ACM International Conference on Intelligent User Interfaces (IUI '10)*, pp. 413 – 414.

[56] S. Hwang, A. Bianchi, M. Ahn, and K. yun Wohn, "Magpen: Magnetically driven pen interaction on and around conventional smartphones." *In the Proc. of ACM International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '13)*, pp. 412 – 415.

[57] A. Kadomura and I. Siio, "Magnail: user interaction with smart device through magnet attached to fingernail," *In the Proc. of ACM International Symposium on Wearable Computers (ISWC '15)*, pp. 309–312.

[58] S. Hwang, M. Ahn, and K. yun Wohn, "Maggetz: Customizable passive tangible controllers on and around conventional mobile devices," *In the Proc. of ACM Symposium on User Interface Software & Technology (UIST '13)*, pp. 411–416.

[59] A. Bianchi and I. Oakley, "Designing tangible magnetic appcessories," *In the Proc. of ACM International Conference on Tangible, Embedded, and Embodied Interaction (TEI '13)*, pp. 255–258.

[60] S. Hwang and K. yun Wohn, "Magcubes: Magnetically driven tangible widgets for children," *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '15)*, p. 163.

[61] S. Hwang, M. Ahn, and K. Wohn, "Magnetic marionette: Magnetically driven elastic controller on mobile device," *In the Proc. of ACM International Conference on Intelligent User Interfaces (IUI '13)*, pp. 75–76.

[62] S. Montaparti, P. Dona, N. Durga, and R. D. Meo, "Openadaptxt: an open source enabling technology for high quality text entry." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '12).*

[63] M. Belatar and F. Poirier, "Uniglyph: only one keystroke per character on a 4-button minimal keypad for key-based text entry." *In Proc. of HCI International 2007*, pp. 479 – 483.

[64] P. Baudisch and G. Chu, "Back-of-device interaction allows creating very small touch devices." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '09)*, pp. 1923–1932.

[65] D.-Y. Huang, M.-C. Tsai, Y.-C. Tung, M.-L. Tsai, Y.-T. Yeh, L. Chan, Y.-P. Hung, and M. Y. Chen, "Touchsense: expanding touchscreen input vocabulary using different areas of users' finger pads." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '14)*, pp. 167–168.

[66] I. Oakley, D. Lee, M. R. Islam, and A. Esteves, "Beats: Tapping gestures for smart watches." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '15)*, pp. 1237–1246.

[67] I. Oakley and D. Lee, "Interaction on the edge: offset sensing for small devices." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '14)*, pp. 169–178.

[68] J. Pasquero, S. J. Stobbe, and N. Stonehouse, "A haptic wristwatch for eyes-free interactions." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '11)*, pp. 3257–3266.

[69] Y. Ahn, S. Hwang, H. Yoon, J. Gim, and J. hee Ryu, "Bandsense: Pressure-sensitive multi-touch interaction on a wristband." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '15)*, pp. 251–254.

[70] J. Kim, J. He, K. Lyons, and T. Starner, "The gesture watch: A wireless contact-free gesture based wrist interface." *In the Proc. of IEEE International Symposium on Wearable Computers (ISWC '07)*.

[71] J. Han, S. Ahn, and G. Lee, "Transture: Continuing a touch gesture on a small screen into the air." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '15)*, pp. 1295–1300.

[72] J. Knibbe, D. M. Plasencia, C. Bainbridge, C.-K. Chan, J. Wu, T. Cable, H. Munir, and D. Coyle, "Extending interaction for smart watches: enabling bimanual around device control." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '14)*, pp. 1891–1896.

[73] D. Akkil, J. Kangas, J. Rantala, P. Isokoski, O. Spakov, and R. Raisamo, "Glance awareness and gaze interaction in smartwatches." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '15)*, pp. 1271–1276.

[74] K. Hinckley and E. Horvitz, "Toward more sensitive mobile phones." *In the Proc. of ACM Symposium on User Interface Software & Technology (UIST '01)*, pp. 191–192.

[75] J. Ho and S. Stephen, "Using context-aware computing to reduce the perceived burden of interruptions from mobile devices." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '05)*, pp. 909–918.

[76] D. Mizell, "Using gravity to estimate accelerometer orientation." *In Proc. of IEEE International Symposium on Wearable Computers (ISWC '03)*, pp. 252–253.

[77] L. Breiman, "Random forests." *In Machine Learning*, vol. 45(1), pp. 5–32, 2001.

[78] P. O. Kristensson, "Five challenges for intelligent text entry methods." *AI Magazine*, vol. 30(4), pp. 85–94., 2009.

[79] K. A. Faraj, M. Mojahid, and N. Vigouroux, "Bigkey: A virtual keyboard for mobile devices." *In Proc. of HCI International 2009*, pp. 3–10.

[80] M. Raynal and P. Truillet, "Fisheye keyboard: Whole keyboard displayed on pda." *In Proc. of HCI International 2007*, pp. 452–459.

[81] M. Klima and V. Slovacek, "Vector keyboard for touch screen devices." *In Proc. of HCI International 2009*, pp. 250–256.

[82] S. Castellucci and I. S. MacKenzie, "Graffiti vs. unistrokes: An empirical comparison." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '08)*, pp. 305–308.

[83] P. Kristensson and S. Zhai, "Shark2: A large vocabulary shorthand writing system for pen-based computers." *In the Proc. of ACM Symposium on User Interface Software & Technology (UIST '04)*, pp. 43–52.

[84] S. J. Castellucci and I. S. MacKenzie, "Gestural text entry using huffman codes." *In Proc. International Conference on Multimedia and Human-Computer Interaction (MHCI 2013)*, pp. 119.1–119.8.

[85] K. Vertanen and P. O. Kristensson, "Parakeet: a continuous speech recognition system for mobile touch-screen devices." *In the Proc. of ACM International Conference on Intelligent User Interfaces (IUI '09)*, pp. 237–246.

[86] S. Canu, Y. Grandvalet, V. Guigue, and A. Rakotomamonjy, "Svm and kernel methods matlab toolbox." 2005.

[87] D. Powers, "Evaluation: From precision, recall and f-measure to roc., informedness, markedness and correlation." *Journal of Machine Learning Technologies*, vol. 2(1), pp. 37–63., 2011.

[88] I. S. MacKenzie and R. W. Soukoreff, "Phrase sets for evaluating text entry techniques." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '03)*, pp. 754–755.

[89] N. Sawhney and C. Schmandt, "Nomadic radio: Speech and audio interaction for contextual messaging in nomadic environment." *ACM Trans. Comput.-Hum. Interact. 7(3)*, pp. 353–383, 2000.

[90] D. Spelmezan, C. Appert, O. Chapuis, and E. Pietriga, "Side pressure for bidirectional navigation on small devices." *In the Proc. of ACM International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI '13)*, pp. 11–20.

[91] C. Stewart, M. Rohs, S. Kratz, and G. Essl, "Characteristics of pressure-based input for mobile devices." *In the Proc. of ACM International Conference on Human Factors in Computing Systems (CHI '10)*, pp. 801–810.