UNIVERSITY OF OSLO
Department of Informatics

Finger Movements Based on Biometric Authentication for Touch Devices

Master thesis

Xiaoli Zhang

Network and System Administration

Oslo University College

May 24, 2011

Finger Movements Based on Biometric Authentication for Touch Devices

Xiaoli Zhang

Network and System Administration
Oslo University College

May 24, 2011

Abstract

The primary goal of this thesis is to collect and compare the touch parameters of finger movements on touch devices and build personal profile with good-recognition parameters to indicate the touch characteristics of individual users using the touch devices. In order to study the possibility of implementation of touch-style identification for touch devices, this work mainly focuses on finding and testing the possible touch parameters which could be used to compose a profile to verify the users.

A full test with an developed anroid application on tablet was performed by 20 subjects to collect touch information, including location of finger points, finger pressure force and speed of finger movements. Statistical analysis was applied on each dataset of the users. The finding has shown that each user can be identified by the discriminative information of finger movements on the touch screen. The results show huge difference in mean, standard deviation and skewness for the dataset of each user giving a reason to hope the implementation of finger movements based on biometric authentication for touch devices. Hopefully, the result of this project will be valuable for further research of implementation of biometric authentication on touch devices based on the finger movements.

Acknowledgement

First and foremost, I would like to express my greatest thanks to my supervisor Frode Eika Sandnes. His inspiration helps me to find this intresting research topic. Thank you for always bringing genius and valuable suggestions into this project, showing great encouraging in my progress, teaching English writing with great patient and all kinds of mental and technical support during the entire project. I feel very proud and lucky to be his student and under his supervising. It will be the most unforgettable experience in my life.

I also very appreciate the help from Kyrre M. Begnum, Harek Haugerud, Ismail Hussain and all the other teachers, thank you so much for all your help and care during my master study. I am so proud to be a student in Oslo University College and so happy to study with all my classmates in the last two years.

Special thanks to the 20 people participated in the full test. Thank you for finishing the testings, patience and letting me collect personal touch information for analysis. This project can not be completed without your help. Unfortunately I cannot acknowledge all of you by name, but I will not forget. Thank you all.

Last but not least, I want to give special thanks to my dearest family, my loved mother, father and grandmother. Thank you for understanding and supporting me to pursue my dream, and letting the only child stay far away from home. I am also grateful to have my dear boyfriend supporting me during these months. Thank you for inspiring and encouraging me all the time. Especially, thanks for participating test in this project, discussing with me and fixing my poor writing English. I really appreciate what you have done, that is more than what I can express.

Once again, thank you all.

Oslo, May 2011

Xiaoli Zhang

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Security was an important issue when the computer and network technologies were introduced to the world. Since the explosive evolution of the Internet in the 1980s, life of human beings have totally changed. Every day a large amount of information and scientific data is produced, transferred through the Internet between different computer systems and shared among people. As the information and data grow, protection and confidentiality are becoming increasingly important. The problems related to computer and network security have been the focus of attention. Areas such as intrusion detection, firewalls, identification and authentication for access control have become popular areas of research.

Authentication is an identification process to validate the user with a legitimate account and set the privileges for the user. The traditional method of authentication relies on a username and password, and it is still the most common authentication method today. One user is given a unique username and password. The username is the user's identity and the password is the evidence the user uses to prove he is the right person with the identity. Username/password authentication is a simple way for access control, but weak passwords have been a common problem for a long time. Weak passwords are easily guessed or cracked using special algorithms [3]. It is also difficult to keep the password secret. If someone wishes to gain access to a user's account, it is easy to look over the shoulder of the person when the user enters username and password. This is called shoulder-surffering [4]. To strengthen the security, stronger passwords are required with more characters, complex combination of digits and letters, and uncommon words. Some computer scientists declare some suggestion rules to created strong passwords which was known as password policies [5]. But these long and complicated passwords are very difficult for users to remember especially when a user has many different accounts. Some users have the habit of writing down their passwords on a piece of paper and keep it in plain view of others. Thus, strengthening authentication with convenient and efficient methods becomes one of the main

research problems in the research area of authentication.

To resolve these authentication problems, more access control methods are introduced such as pin codes, smart cards and biometric authentications [6]. With biometric technology, authentication can rely on the physical or behavioral characteristics which are unique to a person. And biometric authentication has a huge advantage over username/password authentication as it is much harder to fake. Because the physical characters are different from one person to another. Through the benefit of using physical and behavioral characteristics, users don't need to remember the username and password. They are the credentials for themselves to prove their identities. This advantage of convenience and efficiency makes biometric authentication so popular and widely used in today's society.

Touch devices bring a new way of interaction with machines and have totally changed the traditional relationship between human beings and computer equipmentF. The technologies for the virtual reality makes computer users be able to feel virtual objects with their sense of touch [7]. With this technology, people can feel the virtual objects in the computer system as items in the real world and control the movements using fingers. Thus, touch has become a very centralized control sense implemented in many devices. There are already many touch devices on the market such as computer, mobile phones and touch tablets. And these touch devices have a large impact on human beings. Even though most sophisticated touch devices are expensive, people are getting used to these touch devices in their daily life. Information protection and confidentiality is still a problem for these devices. For example, people easily forget or lose their touch mobiles in public places and often there is personal information and valuable data inside.

Touch devices can help the process of system administration and increasingly people are using touch devices to monitor their remote systems. With the installation of monitoring applications on small touch devices, system administrators can monitor and manage the systems and servers remotely with their administrator account. For example, administrators can sit in a coffee shop chatting with friends, at the same time login to the remote system and do their work. But the touch device is small and easily overlooked or somebody may steal it because it is expensive. In this situation, with the data in that touch device someone may login to the system as administrators and stop the web service or even shut down the whole system which can cause severe damages to a company. Thus, how to protect the data in touch devices and strengthen the touch authentication has increasingly attracted attentions.

In order to improve the efficiency and accuracy of touch authentication, this work mainly focuses on finding and testing the touch parameters of individual users and build the personal profiles for the touch devices. The main challenge for this project is how to get the right touch parameters with high accuracy certification and build the reasonable touch information profile for

different users.

1.2 Problem Statement

The problem statements in this project are as follows:

- Does every user have special touching characteristics that can be used as a biometric authentication to verify the user's identity?
- What touching parameters can be used for strengthening authentication?
- How different users show the touch characteristics with the touching parameters?
- How to use these results to build a good personal profile of touching style?

Android will be used to create an application which can be installed on touching tablet device to get the personal touching data. Enough touching samples will be collected from individual users. Then based on the results of these experiments, analyzing and comparing the different touching parameters will be done in order to build touching authentication profiles for individuals with a good recognition.

1.3 Thesis Outline

This thesis is structured as follows: chapter 1 introduces the motivation and problem statements of this thesis. The background and related technologies are introduced in chapter 2. Chapter 3 explains the design and approach which are used to collect the personal touching information. Chapter 4 presents the data and results achieved in different touch parameters. The discussion in chapter 5 analyzes the results and summarizes the making of decisions through the whole project. The conclusion and future work is presented in chapter 6.

Chapter 2

Background

With the rapid development of computer and Internet technology, the confidentiality, integrity and availability of computer devices have changed dramatically. The huge growth of information and data make protection and confidentiality increasingly important. Areas such as computer and network security, authentication, verification and cryptography are given increasing attention from researches. Some new technologies such as biometrics with its high convenience and efficiency become a popular research area.

2.1 Computer and Network Security

The study of security in computer and networks is a rapid growing area of interest because of the fast increasing number of computer users and data transferred between systems. Computer and network security is a critical issue. Security is not only protecting the system which holds personal or organizational data but also building the infrastructure of networks, routers, domain name servers, and switches together; giving the magnitude of securing cyberspace to these different levels and make them work well together [8].

Dieter Gollmann defines computer security as:"prevention and detection of unauthorized actions by users of computer systems." and "measures we can take to deal with intentional actions by parties behaving in some unwelcome fashion" [9]. There are three important issues for the classical definition of securtiy: confidentiality, integrity and availability. Confidentiality means prevention of unauthorized disclosure of information. Integrity makes sure everything is how it is supposed to be. Availability is the property of being accessible and usable upon demand by an authorized entity. Based on these critical objects, a security infrastructure can be built.

All the manifestations of security are organized by the following three components [8]:

- requirements: something users expect security to do (security goals)
- policy: the steps to reach the security goals (the meaning of security)

- mechanisms: the tools, procedures and other ways to ensure the implementation of security steps (ways to enforce policy)

Requirements are the security goals which control the user actions and system states which are allowed or disallowed. The requirements of the security are very different among individual users based on their different needs. For example, a public organization probably needs to put more focus on availability for data sharing when they build their security system, while a personal computer may place more attention on confidentiality and integrity for protecting data storage when implement the security infrastructure on the computer. A security policy defines the meaning of security, with describing the special system statements and what actions are allowed or disallowed. Policies are used to make pairs of system statements and user activities. Users are only allowed to perform the right actions which are allowed under the current system statement. Security mechanisms are the methods used to ensure and enforce the policies and protect systems from statements which are disallowed according to the security policies. These three components constitute the manifestations and features of computer and network security.

The implementation of computer and network security is a continual process. As lots of research has been done to secure systems and networks, new vulnerabilities are being discovered and new softwares are being developed [10]. The most common measures used to secure computer and network are Intrusion Detection System(IDS) and firewalls.

2.2 Intrusion Detection and Firewalls

2.2.1 Intrusion Detection System

An intrusion is defined as "the act of thrusting in, or of entering into a place or state without invitation, right, or welcome" [11]. In other words, an intrusion is usually called an attack which is not allowed by the system itself. Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of possible incidents, which are violations or imminent threats of violation for computer security policies, acceptable use policies, or standard security practices [12]. Intrusion detection system (IDS) monitors and collects data from a target system that needs to be protected, processes and correlates the gathered information, and initiates responses when evidence of an intrusion is detected [13]. IDSs cannot stop the attacks, but to gather and analyze the data from the a number of sources and tell administrators if their system has been attacked successfully. IDSs actually watch the statements of the system and cause an alarm once any attacks take place. There are three main categories of IDS classified by their functionalities: Host-based intrusion detection system (HIDS), Network intrusion detection system (NIDS) and Distributed Intrusion Detection System (DIDS).

Figure2.1 shows a basic architecture of HIDS in a network system. HIDS is built separately on web server, domain name server (DNS) and different internal hosts. The rule set of each HIDS may be different based on the functions of the computer systems. These HIDS can help system administrators monitor the activities of each host and analyze the statements of the computer system to inform about potential intrusion incidence in the network system.

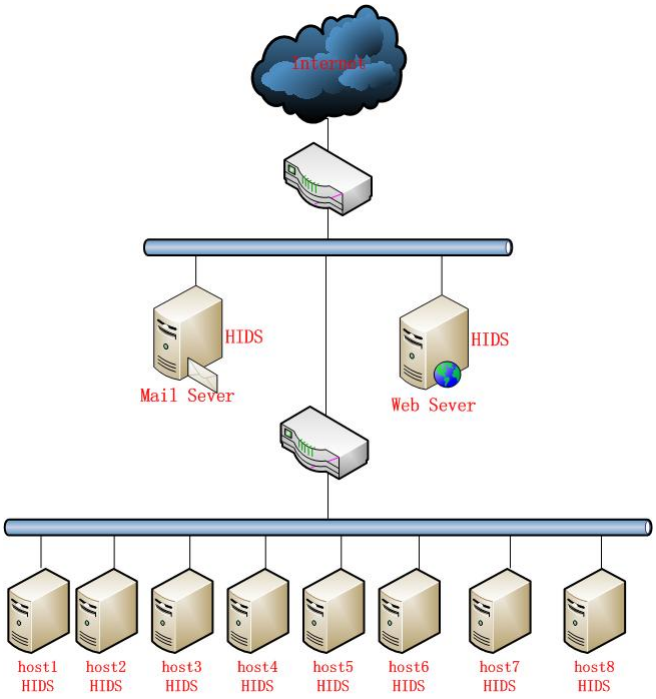


Figure 2.1: The host-based intrusion detection system network.

As shown in Figure 2.2, a network uses two NIDS. The NIDS units are placed in the basic architecture of the network layers and can monitor the network traffic for all the devices. All the incoming and outgoing traffic packets of this network will be detected and analyzed by one NIDS to check there are any malicious codes inside. The internal host systems inside the network are protected by an additional NIDS to mitigate the exposure of the internal host systems. This type of multiple NIDS within a network is a defense-in-depth security architecture [14].

Figure 2.3 shows a DIDS system comprised of four sensors and a centralized management station. The sensors NIDS1 and NIDS2 are protecting the public web and mail severs, while the sensors NIDS3 and NIDS4 are used to protect the internal host systems. These four sensor NIDS agents are connected and controlled by a NIDS management station.

2.2.2 Firewall

A firewall [10] is implemented as a series of packet-filtering rules defined by options on the iptables [15] command line. With looking outwardly the sys-

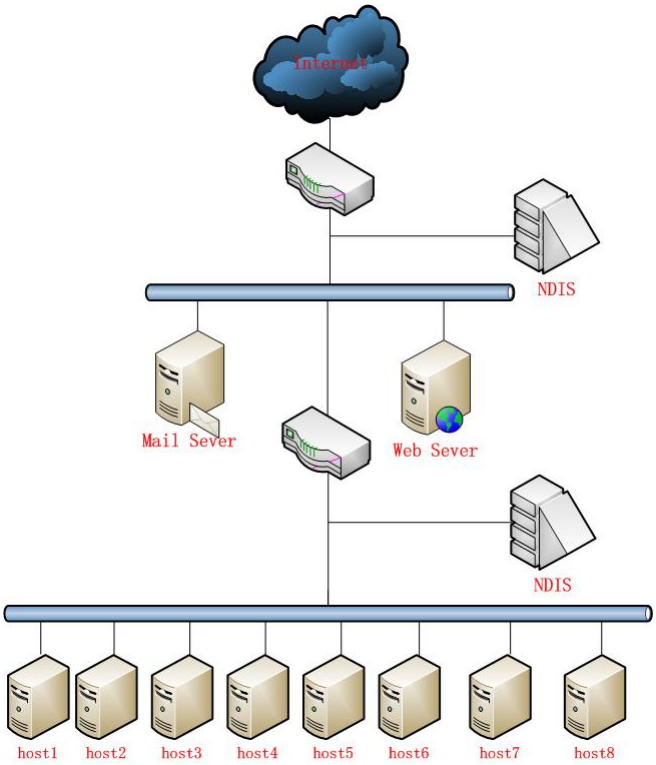


Figure 2.2: The network intrusion detection system network.

tem to detect the intrusions, firewalls can discover the attacks when malignant packages pass through it. This is a huge advantage compared to IDS.

The term firewall has various meanings depending on its implementation and purpose. Steve Suehring and Robert Ziegler describe firewall in their book as "the Internet-connected machine. This is where your primary security policies for Internet access will be implemented" [10]. The first firewall was developed in 1988 by the Digital Equipment Corporation (DEC), which was known as packet filter firewalls [16]. From the first generation of packet filter firewall, the history of firewalls goes through the application layer firewalls, stateful firewalls and some firewall commercial products. Now firewalls are very wildly used to enforce the security policies people defined and protect personal and group computer systems.

Figure 2.4 shows the basic architecture of a computer system with five layers:

Based on the system architecture, firewalls can be classified into four types depending on where the communication is taking place, where the communication is intercepted and where the state is traced.

- proxy device
- network layer firewalls

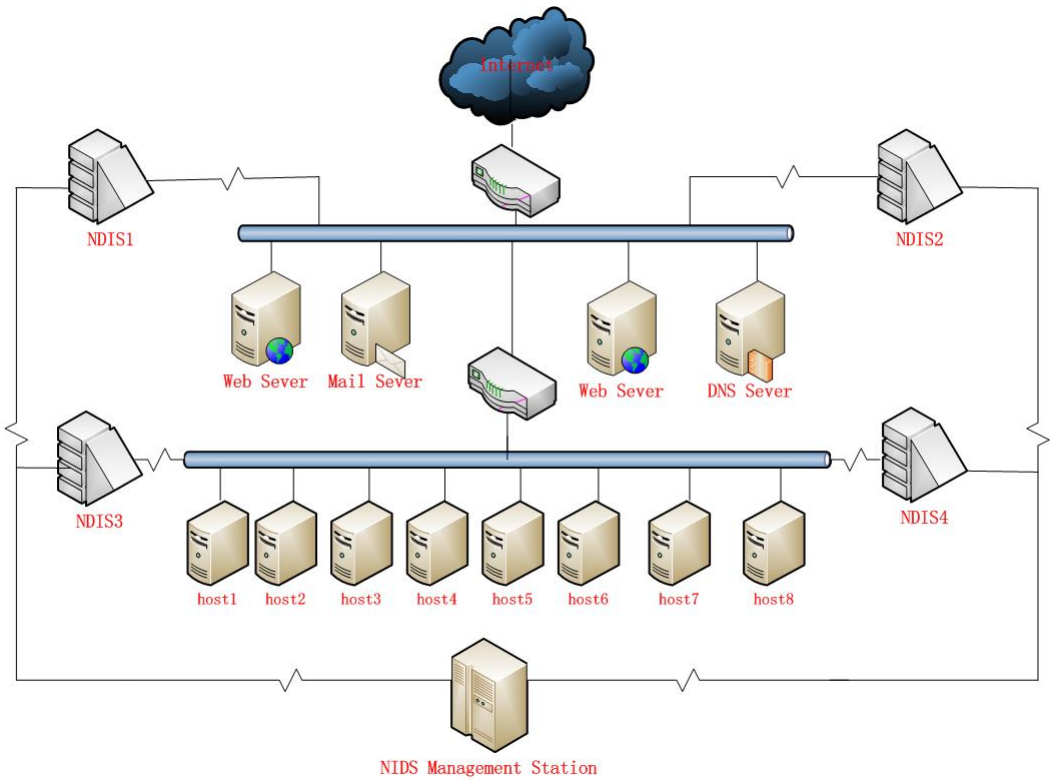


Figure 2.3: The distibuted intrusion detection system network.

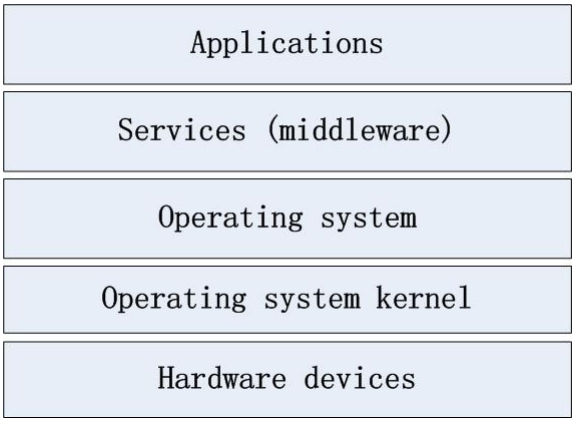


Figure 2.4: The computer system layers.

- application-layer firewalls
- network address translation (NAT)

Summarily, firewalls scan the network traffic flows which go through it, then filter out some of the attacks and stop them to be executed on the target computer. Firewalls can turn off the port which does not be used. And it also can control the incoming and outgoing traffics to prevent system away from the attacks.

2.3 Identification and Authentication

Most of the currently available intrusion detection systems and firewalls do not provide any authentication functionalities to identify the users who access the computer system. But with the growth of information and scientific data in today's society, data protection and confidentiality becomes increasingly important. One efficient way is validating users with legitimate accounts and giving the data access control to the users. The process of verifying a user's identity is typically referred to user identification and authentication. In computer security, indentification and authentication are the issues about who the users claiming to be and how to prove their identities. Once users are given legitimate accounts, they get their identities and the credentials to prove the identities in the right system. Every time when users want to access to the system, they need to go through the authentication process to prove their identities and let the system give the right data access to them. Often this is achieved through a system log in process. Some systems require several repeated authentications to reduce the chance of an attacker using a machine where another user is logged in [17].

After the process of validating users, the access to the system resources and the privileges to run activities on the system are given to the users based on their identifications. Normally, in many systems there is no more process to redo the authentication after the user is verified. Some few applications [18] implemented a method called re-authentication to guarantee that the current user is the right one authenticated before. This is a methodology to continually monitor the users' behavior and verify their identities after the log in process.

Typically, the authentication process is before the start of each session. The combination of username and password is the traditional method most often used for verifying users. The username is user's identity and the password is the evidence which the user used to prove he is the right person with the identity. Figure 2.5 shows a basic process of username/password authentication:

Username/password authentication is a simple way for access control, but this approach has often proven inadequate in preventing unauthorized access to computer resources when used as the sole means of authentication [17]. Another common problem existed for a long time is weak passwords. But strong

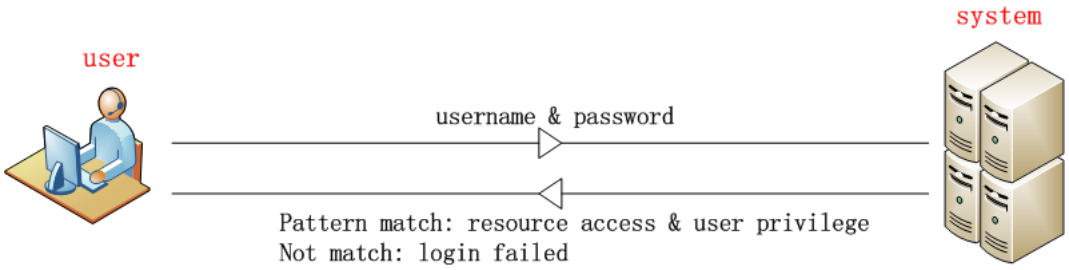


Figure 2.5: The process username/password authentication.

passwords with many characters, complex combination of numbers and letters, and uncommon words are difficult to remember. Suffering from lots of password attacks such as shoulder-suffering [4], dictionary attacks [19] and password cracking [20], the username/password authentication method faces increasingly threats on the Internet.

To resolve these problems, more access control algorithms based on the user authentication options are introduced. Dieter Gollman describes the user authentication options in his book as follows [9]:

- (a) Something you know (passwords).
- (b) Something you hold (token).
- (c) Who you are (your body).
- (d) What you do (your actions).
- (e) Where you are (space, time, context).

(a) is the username/password authentication which is most often used. (b) is a quite common authentication method used by banks. (a) and (b) are very commonly used in today's society. (c) and (d) are some new technologies related to biometrics which identify users by the physical or behavioral characteristics. This type of technique with good convenience and efficiency in authentication has recently become a hot research.

2.4 Biometric Authentication

With the rapid development of the biometric technology, biometric authentication has become a popular research area. Biometrics is a type of authentication method used for identifying a person based on a physiological or behavioral characteristics such as fingerprints, facial characteristics, hand signatures, voice, gait and keystroke dynamics [21]. Biometric authentication is much easier for users than remembering passwords. As shown in the figure 2.6, a sample of the biometric trait is first captured, processed and stored in a database.

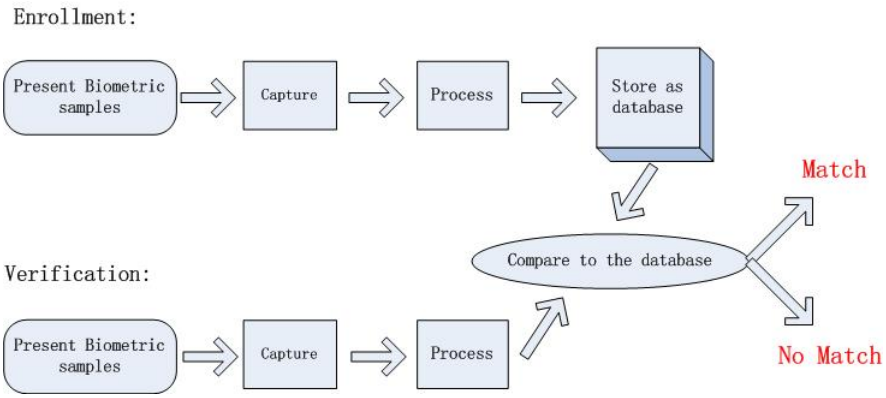


Figure 2.6: A sample of biometric trait.

Then the newly captured biometric sample can be compared with the entries in the database to verify the identity of a person.

Biometrics has a long history. The first recorded account of biometrics dates back to the 14th century involving fingerprinting in China. A European explorer Joao de Barros wrote that the Chinese merchants were using ink to stamp children's palm prints and footprints on paper so as to differentiate the young children from one another [22]. This method is still being used today. As now the biometric technology has moved from a single method of fingerprinting to several effective methods. There are a number of biometric authentication methods studied and implemented in the real life.

These biometric authentication technologies are studied and implemented to help people to strengthen the security of authentication systems. As shown in Figure 2.7, several of biometric authentication methods are produced and used today, and fingerprint is the most common biometrics with occupying percentage of 48.8 on the market. The data in figure 2.7 is taken from [23].

The identity authentication system based on the unimodal biometric method may not be acceptable to a particular user group or in a particular situation. To overcome the problems existing in the conventional unimodal methods, multimodal biometric identification technology [24] is developed and becoming a popular research area in biometric authentication. This type of biometric technique uses two or more individual modalities to improve the identification accuracy [25].

2.4.1 Fingerprint Identification

Fingerprint identification is the most important and widely used biometric technology, with the recent major advances in fingerprint technology [26]. Every person has a unique fingerprints. Based on this truce, fingerprint indentification, which is also known as dacyloscopy [27], is a type of biometric authentication method to collect the ridges and furrows on the surface of a fingertip from one person and compare it with the fingerprints database to detect this

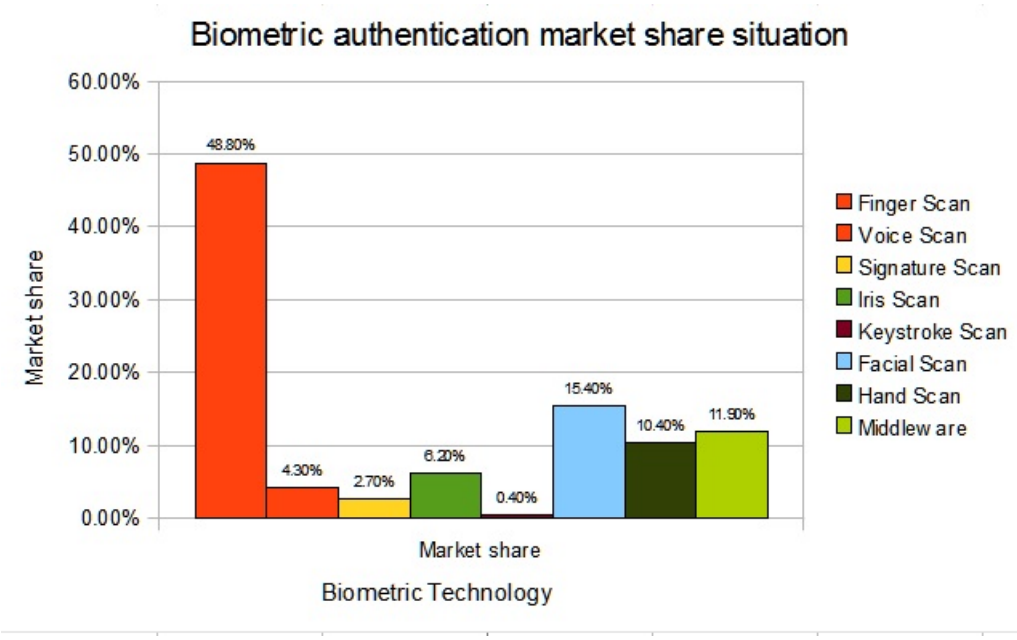


Figure 2.7: Biometric authentication market share situation.

person's identity.

The two most prominent ridge characteristics are used for verifying individuals, known as minutiae, are ridge ending and ridge bifurcation [1]; a ridge ending is the point where a ridge ends abruptly and a ridge bifurcation is the point where a ridge forks or diverges into branch ridges. Figure 2.8 shows one example of ridge ending and ridge bifurcation on a fingerprint image:

These ridge characteristics are never exactly the same for two individual persons. This is the foundation of figerprint indentification. By comparing the ridge characters, fingerprint authentication systems can easily verify the user's identity. This technology can be very convenient and hard to counterfeit. The equipments to implement fingerprint identification are not too expensive. All these reasons support fingerprints becoming the most wildly used and important biometric authentication technology today.

The history of fingerprint identification starts from 14th century in China. In 1980, the first fingerprint authentication system, which was developed by Richard Edward Henry of Scotland Yard, essentially reverting to the same methods used by the Chinese for years was used by the police. And now fingerprint identifiction is used in a lot of areas and plays an important role in modern authentication systems.

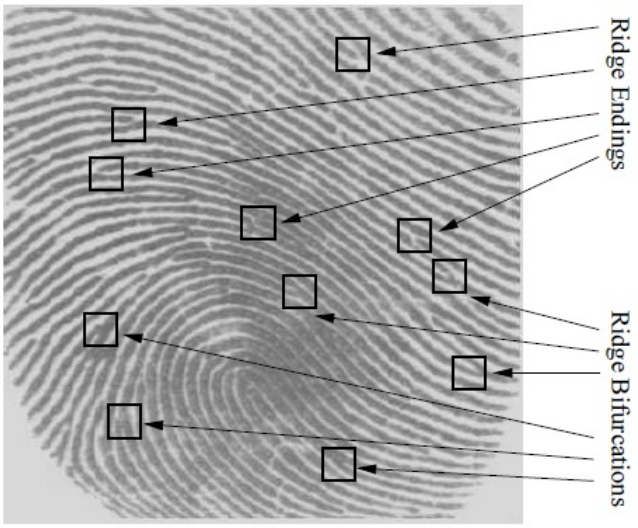


Figure 2.8: A sample of ridge ending and ridge bifurcation on a fingerprint from [1].

2.4.2 Face and Ear Recognition

With the development of biometric technology, more and more physical characteristics from human beings can be used as identification factors such as face and ear. Facial recognition is one type of biometric identification which can detect individual user's facial features and use these as patterns to verify the user's identity [28]. The technology of facial recognition has recently drawn a substantial amount of attention with the conventional cameras which is so often used to capture the images of people's face. The human ear is a new feature in biometrics which can be easily captured from a distance without a fully cooperative subject, although sometimes it is obstructed by hair [29]. With these advantages, ear recognition has become a very attractive research area of biometric authenticaion.

Biometric face recognition, a very popular biometric authentication method, works by using computer systems to analyze the facial structures from people's photos which were taken before, and verifying the identities of the subjects based on their facial characteristics. An essential facial image database needs to be built before the recognition process by using cameras to take photos from individuals. Recognition process compares the facial characters including the distances between key characteristics such as eyes, nose and mouth, angles of key features such as the jaw and forehead, and lengths of various portions of the face. Some studies [30, 31] shows that the way of taking a photo and the combination of different facial characters will affect the subject's facial template used for identification. There are several advantages to using biometric face recognition: a non-intrusive verification process which is similar to having a photo taken, a fast and reasonable biometric recognition by

comparing facial images with templates in the facial database, the only visual biometric identification confirmed by facial photos. Although everyone has a unique face, but it will change by time in the real-life environments. Some the facial features will change when people are getting old. So the facial template database needs to be updated frequently.

Biometric ear recognition is a new technology in biometrics to verify the identities of people based on their ear characteristics. Compared with the human face, the human ear is a relatively stable structure and not changing much with the age growing and facial expressions. A medical study [32] shows that the growth of the human ear is proportional after the first four months of birth and the changes are not noticeable in the rest of human's life. And the ear features can be captured from a distance without the announcing people, so the implementation of ear recognition benefits a lot in the area of automatic identifying people. With these advantages, ear recognition technology has recently attracted attention in the research community. A voice is announced that the ear recognition may outperform face biometrics in the near future [33].

2.4.3 Voice and Gait Recognition

Just like fingerprint, voice and gait are unique to individuals which can be used as the features of biometric authentication. Voice recognition, also known as voiceprint identification, is the biometric technology used to convert human voice to digital signals and analyze the characteristics to identify the users. With the existing telephones, the voice recognition allows users to authenticate remotely, which brings convenience for authentication process and reduces the cost of implementation. Another advantage of voice is that the storage size of the voiceprint is small compared with fingerprint and facial images. Unlike other biometrics, gait recognition relies on video. Gait recognition is defined to be the recognition of some salient property such as style of walk, based on the coordinated, cyclic motions that result in human locomotion [34]. Figure 2.9 shows the basic steps of the voice and gait recognition processes.

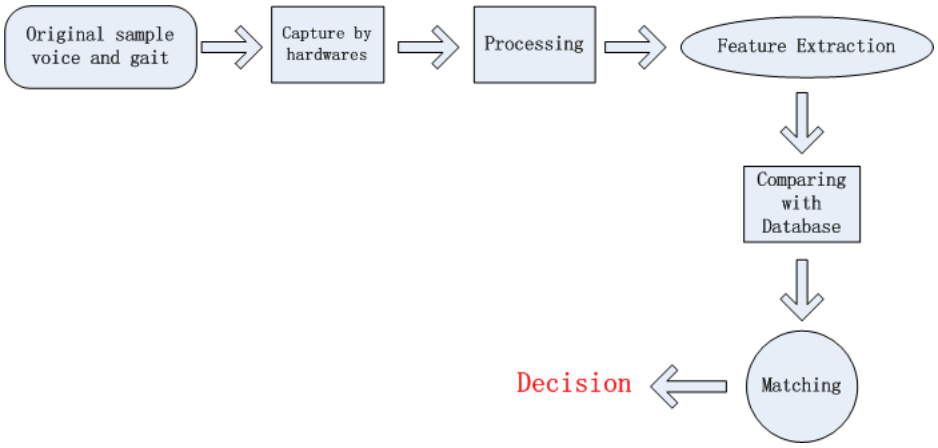


Figure 2.9: Steps of voice and gait recognition methods.

Samples are collected by telephones and cameras. For voice recognition, users are asked to speak a certain set of words or phrases, or to speak for a certain length of time. In gait recognition users need to walk for a certain distance. Then after the computer process, a digital representation of the voice and the gait is created and compared with database to verify the users' identities. These two biometric methods are easy to use and easily accepted by users and fit for many requirements of the future authentication such as convenient, remote and automatically. Although voice and gait recognition are not the most secure of the biometric technologies compared with fingerprints and facial recognition, they become more powerful when used in conjunction with other authentication systems.

2.4.4 Keystroke Dynamics Authentication

Keystroke dynamics, is a method for identifying users based on their typing patterns, and a popular research area in biometric authentication studies because of its transparent in authentication based on the existing hardware. There are numbers of keystroke dynamics methods [35] were proposed since the advent of computers.

The first keystroke dynamics authentication method developed during the World War II, known as the "first of the sender" [36] to identify senders transmitting a message by the rhythm, pace and syncopation of the signal. Then a lot of research work has been done to improve keystroke dynamics as a method of identification. The study and achievement of Gaines [37] in 1980 is a big thing to attract people's attention to keystroke dynamics with a complete novelty method of biometrics. Then Joyce and Gupta established a new algorithm to calculate the difference between a reference vector and an instant vector in 1990 [38]. Gokcay used two neural network algorithms, back propagation and self organizing feature maps to implement keystroke rhythm analysis in 1991 [39]. In 2001 Monrose and Rubin found a new way to harden passwords [40] by using polynomials and vectors for the keystroke patterns. Now keystroke dynamics identification is not well developed or implemented. But with the huge growth of using computers, the research work of this area has increased. Some creative methods are appended to the Keystroke Dynamics Authentication such as Alert Levels [41] to improve the accuracy of the identification.

There are two main factors usually measured to verify identities with keystroke dynamics authentication [2]:

- Dwell time: how long a key is pressed.
- Flight time: how long it takes to move from one key to another.

The keystroke dynamics identification systems can collect the time each key is pressed down and the cycle time between one key-down and the next.

These time information is different from each other based on the person's typing style. As shown in figure 2.10, data of dwell time and fight time is collected and analyzed when a user is typing words, then keystroke dynamics authentication system uses these typing information to verify the identity of a user.

Figure 2.10: Keystroke dynamics identification measurement from [2].

2.4.5 Mouse Movements Identification

Another very new and popular research area of biometrics is mouse movements identificaion [42, 43, 44]. This type of biometrics technology verifies a user's identity based on the features of mouse movements when the user is using the mouse. Traditional authentication techniques can be applied on re-authentication system by asking the users to re-type the passwords or tokens. However, repeated authentication is nasty to the users, inconvenient, and often unreliable. Mouse movements identificaion provides a new way to implement the re-authentication automatically by building a model of a users' behavior directly from their mouse movements.

A mouse movements identification system can consist of three components: mouse data collection, feature extraction and pattern classification [45]. The information of mouse movements from the individual users is first captured, then these data is analyzed to extract features to show the style the user prefer, at last these features are classified to show the characteristics of mouse movements from the user. Some mouse movements identification systems put more attentions on the curves when a user is moving the mouse [46].

The main advantages of the mouse movements idenfication are as follows:

- Re-authentication system: Normally the authentication system is processed before a session. Once the session is started, it is impossible to go back to the authentication step to find out if the user is the one with right identity. Mouse movements identificaion solved this problem by monitoring the mouse movements from the user.

- Low costs: unlike other biometric authentication methods which needs very expensive equipments to collect physical characteristics, mouse movements identification systems don not need any special hardware devices but collect data just from mouse. The cost of this type of biometrics is very low.
- Low invasiveness [45]: with the straight purpose of detect the type of mouse usage, mouse movements identification is processed during one person using the computer. Once the session started, it will not be stopped. And this identification systems collect data from the mouse which is the device difficult to attack.

2.5 Touch Screens

A touch screen is a special type of electronic visual screen which is sensitive to detect the touch location and pressure by fingers. The significant milestone in touch-screen technology is the first "Touch Sensor" developed by Doctor Sam Hurst in 1971, which is the predecessor of modern touch screens. Elographics developed the five-wire resistive technology in 1977, the most popular touch-screen technology in use today [47]. With fast development of modern technologies, touch screens are wildly used on many devices such as mobile phones, tablets, computers and hospital equipments. The touch devices are very friendly for users since touch screens can give the reflections immediately to the users when they are picking up choices on the screen with their fingers. This has totally changed the traditional way of communication between users and computer devices.

As people are getting used to different touch equipments, data protection for these touch devices becomes problem. One solution for this problem is to combine touch screens and biometrics together to build touch biometric authentication systems on the touch devices. When users put their fingers on the touch screens to control the devices, the related touching information from users is collected, analyzed and compared with owner's samples which are stored to verify the identity of users. In this way, touch devices are able to "recognize" their owners when their screens are touched by users.

This chapter introduces the relative background. The following chapter will describe the detail approaches of implementation of this project.

Chapter 3

Methodology

This chapter covers the design and implementation of the experimental environment including: collection and classification process for personal touch information, development android application, building the testing environment, analysis of touch parameters and properties, and the experimental testing procedure.

In the data collecting process, log files are used to record the touch information from the users. Using data from multiple users and multiple touches with only one finger is essential to get comparable results in the experimental data. Data graphs are created for touch parameters, which reflect the patterns of each user, and in turn differentiate and recognize them. During the analysis process, statistical methods are used to calculate distributions. Then the patterns of touch parameters and users are stored into one database for future authentication. An overview of the data gathering and classification process is illustrated in Figure 3.1.

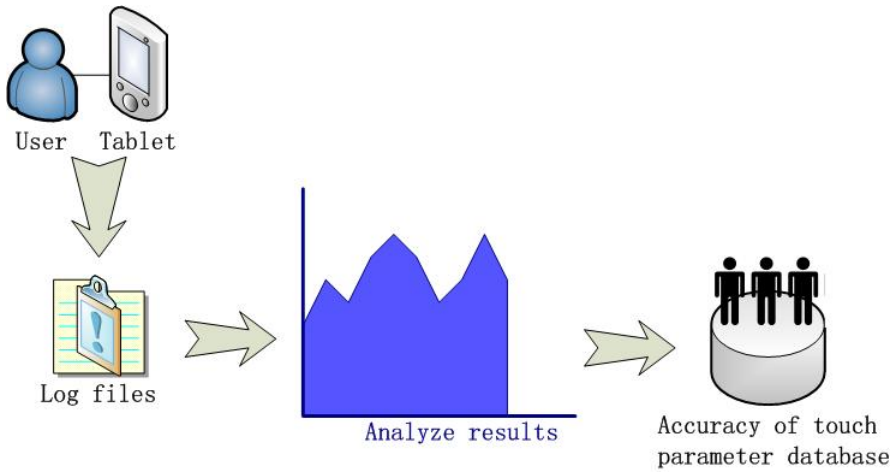


Figure 3.1: Overview for data gathering and classification process.

In this study, touch information such as finger location, finger pressure, pressing time and moving distance from the touch device when the users use

it has collected. The data was saved in a log file. Data analysis tools are used to deal with the results, calculating directly through using mathematical methods and graphical visualization to see whether the user's finger movements provide an accurate model for the identification of a user.

3.1 Objectives

Based on the terms and concepts introduced in the background chapter, the objectives described here are corresponding to the problem statements which were discussed in the section 1.2.

According to the problem statements, to verify whether every user has special touching characteristics, personal touch information needs to be collected and analyzed. A data collecting and classification method was designed and implemented to collect the touch parameters and properties from individual users. An Android application is developed as the tool to be installed into a touch device and collect touch information such as finger location, finger pressure and finger movements from individuals. Then this data collecting process is tested through a quantitative method. Multiple users perform the same touch testing experiment under the same testing conditions. Observation and analysis from the data of full test can show the different touching characteristics from individual users. Statistical methods are applied to measure and analyze observations in order to improve the accuracy of the results. Based on the data of the experimental testings, the various comparison of several touch parameters among different users will be discussed and evaluated, with respect to achieve a good accuracy of individual identification.

The following sections cover the implement of the experimental environment, study of touch parameters and properties, developing android application to collect touch information, applying statistical methods to analyze touch behaviors from individual users, and introducing the technologies used in this project.

3.2 Environment

Before the experimental environment was set up, an android application was developed to collect touch information from users. The programming structure was built using the Android Software Development Kit (SDK) with the JAVA Eclipse development platform. Advantages of using open resources, such as libraries and built projects, made android the selected option for developing the data collecting application. JAVA Eclipse platform was chosen in this project because of the familiar proficiency with the programming environment. Eclipse can simulate a virtual touch device for testing before implementing the application into the real touch device. Another reason is that JAVA is a very popular language and widely used in the world. The real testing equip-

ment is the Samsung Galaxy Tab which is a 7-inch touch tablet supporting Android 2.2 operating system.

3.2.1 Eclipse Android Development

In 2007, Google announced Android as an open source software platform and operating system based on the Linux kernel. It was firstly used on mobile devices and then expanded to the laptop computer and many other areas suc television and mp3 player. The investigation of 2010 shows that with only two years of the formal launch, Android has surpassed Nokia Symbian OS system and become the most popular smart-phone platform in the world.

Android comprises as operating system, middleware, user interface and application software. It uses the software stack architecture and can be divided into three parts: The low layer, developed in C, is based on the Linux kernel and it only provides basic functions. The middle layer, developed in C++, includes the function library and virtual machine. The top layer, developed by the companies themselves with JAVA, is a variety of applications.

Eclipse is a popular cross-platform Integrated Development Environment (IDE). This project used Eclipse android SDK to develop android application to collect touch characteristics from individual users. After the programming environment was set up, a virtual mobile phone named Android Virtual Device (AVD) was used for running Android projects. Figure 3.2 shows the interface of this AVD in the computer system.

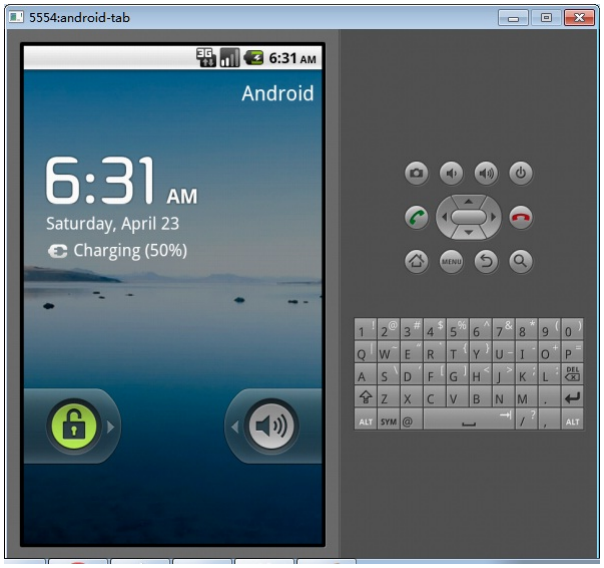


Figure 3.2: The Android Virtual Device interface in computer system.

3.2.2 Equipment

All the touch information from individual users were collected through a Samsung Galaxy Tab. Once the users touch the screen with their fingers, the information of the touching characteristics are collected and stored in a log file. Their finger events such as press-down and sliding movements were handled and some touch parameters such as finger position, finger pressure, push-down time and move distance were recorded for later analysis.

The Samsung Galaxy Tab is an Android-based mobile phone and compact tablet computer. This project uses this tablet as the testing equipment to install Android application and collect touch parameters from users. Figure 3.3 shows the conformation of the Samsung Galaxy Tab. The hardware information of the Samsung Galaxy Tab are listed in table 3.1.

Figure 3.3: The Samsung Galaxy Tab

Operating System	Android 2.2
CPU	1 GHz ARM Cortex A8 ”Hummingbird”; 1.2 GHz
Storage	Flash memory
Memory	512 MB
Capacity	16 GB models and microSD slot
Display	1024 600 px (aspect ratio 10:6), 7.0 in (18 cm) diagonal
Input	Multi-touch screen

Table 3.1: The Samsung Galaxy Tab hardware information

3.3 Touch Parameters and Properties

When the users touch the tablet with their fingers, a number of touch parameters and properties are collected and stored in the log files. In this experimental project, the activities of ACTION_DOWN, ACTION_MOVE and ACTION_UP

are the most important finger events used to catch touch parameters. All these three finger actions are the events of a class named MotionEvent which is the object of class encapsulation of the touch screen event, encapsulating all the event information such as the location of touch, touch type, and the touch time. The figure 3.4 shows one example of touch event illustrating the finger actions actually happened during the time period of the touch event.

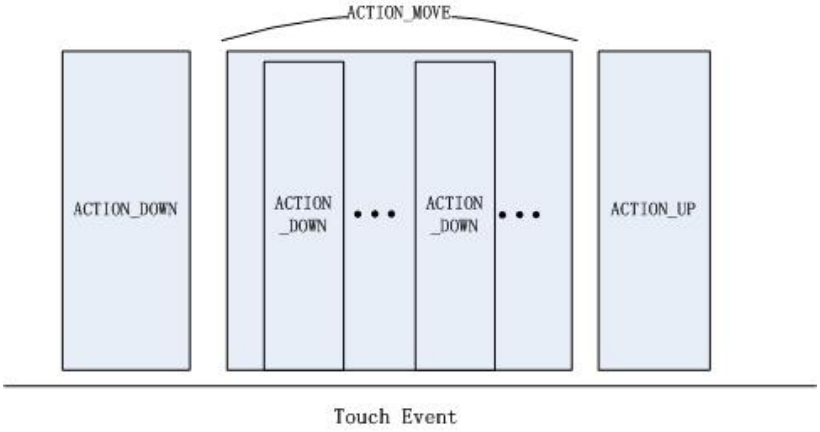


Figure 3.4: Graphical representation of one touch event.

The ACTION_DOWN is defined as the event of pressing the touch screen, not moving. This type of touch event occurs once the user press the screen and is the first touch event to be handled. The ACTION_MOVE is defined as the action of moving the point of load after pressing the touch screen. More than one ACTION_DOWN events can happen during the process of the ACIONT_MOVE event. Because only the ACIONT_DOWN event may be handled by more than one View, others must be handled only once. The ACTION_UP is the event referring to release the touch screen.

With the methods of onTouchEvent in Android, a set of touch parameters can be caught and collected from the screen during the different types of touch event. The following are the touch parameters and properties used in this project:

- press_type: the type of one touch event is defined as three situations in this project: press_down, press_move and press_up. The press_down means pressing the screen, the press_move is one finger sliding on the screen and the press_up stands for releasing the finger from the screen.
- point_x: the value of x-axis of each touch point location on the screen.
- point_y: the value of y-axis of each touch point location on the screen.
- pressure: the finger pressure on the screen from each touch point.
- size: the size of each touch point.

- downtime: the start time of pushing down the screen.
- eventtime: the end time of each touch event.

The value of point_x and point_y are the finger's relative position of the container in the android application. In this project, the application-view window is a little smaller than the screen of Samsung Galaxy Tab. To reduce the deviation, this relative position is treated as the finger position on the screen.The pressure generally ranges from 0 (no pressure at all) to 1 (normal pressure), however the physical meaning of this item is explained fuzzy in the SDK documentation [48]. The pressure parameter in this project is considered as the average value of pressing on each touching size area.

With these parameters, more features are extended with mathematics calculations. With the finger locations of touch points, the displacement of each touch point can be obtained. Moreover, using the displacement and the time length of touch event, the speed of the finger sliding on the screen is calculated. The force of the finger used to press the screen comes with the parameter pressure and size. And the whole touch time of each touch event is calculated by the parameter downtime and eventtime. To extract the parameters, a procedure was created by a script. This procedure reads the log files and do the calculations with some data to get new features. The following are the extracted parameters:

- press_counter: the counts of the pressing down event on the screen during the whole reading process.
- point_dx: the value of displacement in x-axis on the screen between two adjacent touch points.
- point_dy: the value of displacement in y-axis on the screen between two adjacent touch points.
- finger_force: the value of finger force on the screen from each touch point.
- touch_event_time: the time period calculated by the parameter of downtime and eventtime. It is the time stamp from pushing down the screen till release the finger as the end of a touch event.
- speed_x: the speed in x-axis of one finger moving event.
- speed_y: the speed in y-axis of one finger moving event.

3.4 The Android Application

As part of this project, an android application was developed to collect touch information from individual users through a tablet. An android application named PDF-Viewer was selected option with the advantage of the existing resource of the project architecture and the main source code. Special functions

are written into this Android application to create log files and record touch information.

An android project is usually written with a combination of four components: activity, intent receiver, service and content provider. Activity is used to show the response activity to the view events. When the application needs to execute a response to external events, a intent receiver will be used. A service is a number of static codes which can be running without user interface. Content provider is a special class used for data recovery to implement data share with several different application projects. All these components and the functions are announced in an XML file named Android Manifest.xml.

In this project, a PDF reader was chosen as a prototype of the final android application used for full test. The design of the layout and the main project structure was downloaded from the webpage of android open resources. Basic functions of getting the path of file and showing the PDF pages to the user were implemented by the original codes. Based on these functions, graphic changes and functional methods were written into the source code to make the application fit for gathering touch parameters in this project. After completing this android application, username was requested to create a log file in the first window of user interface, this PDF viewer opened the same PDF file for each testing, and background functions were running to collect touch information from users during the whole reading process. Figure 3.5 shows the main graphic windows of the final android application used for full test when it is running in a AVD.

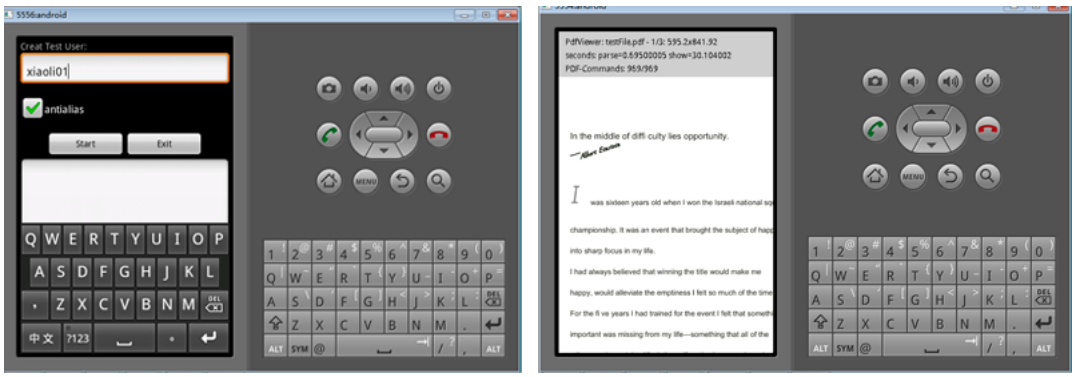


Figure 3.5: The implementation of Android PDF veiwer application.

The user needs to use the finger to slide the pages to read and can use manual button to move to next or previous pages, and zoom in or zoom out. During this reading process, the information of the touching characteristics are collected from users and recorded into the log files which were created in the first process of this application. The log files are very important for future analysis.

During the development process, some difficulties were solved. The first problem was how to create the log files manually each time to store the touch

information of different users. The final solution was adding a login process to ask the users to type the username every time before the reading process, creating a log file for this user based on the username and recording all the touch parameters and properties of this user into the corresponding log file. Second problem was how to write data into the log files in android platform. This was solved by creating a file-output stream every time when a touch event was occurred, and writing all the touch information as string into the log file through this output stream. The third problem is choosing the document for users to read around 10 minutes. Since the testing subjects were chosen in different gender, age and professional area, it was difficult to choose a document which can make all users find interesting during the reading in 10 minutes. The final decision was a comic book. A short 24-page comic story was chosen as the reading material.

3.5 Data Collection

The full testing was participated by 20 different users. All of the touch information were collected from the same Samsung Galaxy Tab. Every user was asked to hold the tab in the horizontal direction which made the comic page is easy to read. All the subjects were provided the same comic paper to read. And every time when the users touched the screen, only one finger could be used, 2 or more finger touching points make subsequent analysis and classification more difficult.

In this project, the focus remains on finding the possibility of user identification from the finger movements based on touch biometric and selecting the touch parameters which result in a high recognition accuracy. It is easier to analyze one finger movement to achieve the objectives of this project. Table 3.2 shows 20 different users participated the full test. Since every person has different reading speed and habits, the time users used to read this 24-page comic story is different and the size of the log files are distinct. In order ot get a wide range of testing samples, people in different gender, age, origin and the proficiency of using touch devices are chosen to do the testing to collect personal touch information. Table 3.2 shows detailed information about the subjects.

As shown in the table 3.2, 10 of the subjects were male and the other half were female. 13 of the users were between 20 and 30 years old, 6 users' age were between 30 and 40, and only 1 users were over 40 years old. The main age of the user group were between 20 and 30, who were supposed to be the main user group of the touch devices on the market. Most of the testing subjects are students studying in engineering and biology. These individual users were from Europe, Asia, Africa, North America and South America.

The testing users were divided into 4 different proficiency levels of using touch devices, which was level 0 to level 3. Level 0 means the users never used any touch devices, level 1 means that the users haven't own any touch

Gender	male	10
	female	10
Age	40+	1
	30+	6
	20+	13
Origin	Europe	9
	Asia	7
	Africa	2
	North America	1
	South America	1
Proficiency of using touch devices	Level 0	2
	Level 1	3
	Level 2	4
	Level 3	11

Table 3.2: Detailed information of the subjects.

devices but used some public machines with touch screens a few times, level 2 points to the users who have touch devices but use them with a low frequency such as once a week or several times per month, and level 3 which is the most proficient level means the users have at least one touch device and use it every day. 2 users were registered as level 0. 3 users were at level 1. At level 2, 4 users were found and 11 users were thought fit for the highest level with good proficiency of using touch devices.

3.6 Select Parameters from Characteristics

After completing the data collection, the touch characteristics information of individual users were stored inF log files. With the original touch parameters and extracted parameters, many data about the users' touch styles was available for analysis. The following step is to select some parameters from touch characteristics for later analyzing. This project focuses on the following parameters to verify how different between people when they are using fingers on the touch screen. The touch parameters are discussed in section 3.3.

- (a) press_counter
- (b) point_x
- (c) point_y
- (d) finger_force
- (e) speed_x
- (f) speed_y

(a) is the parameter about the counts of the pressing down event on the screen during the reading process, which can refect how much the users like to press the screen. (b) and (c) can determine the location of each touch point on the screen to show which part of the screen users like to touch. (d) stands for the value of finger force on the screen from each touch point, which is supposed to be very different between individuals. The moving speed of the finger on the touch screen is defined by (e) and (f) to show how fast different users sliding on the screen.

3.7 Technology

After completing the experimental testing, the statistical analysis methods are used to calculate and analyze the collected touch information data from individual users. With mathematical analysis, the mean and standard deviation of each data are calculated and compared to show the differences between the datasets, which indicates the differentiations of the characteristics of the users using touch devices. Tools such as Excel, Perl scripts and R scripts are used to deal with the datasets and draw the graphs based on the touch data from each subject.

3.7.1 Statistical Analysis

In statistics, mean is the arithmetic mean which is the average value of a dataset. If the data is (x_1,x_2,x_3,...,x_n), the arithmetic mean μ is calculated in formula 3.1. This data can show the average value of each touch parameter such as the average finger pressure force and the evenness finger-move speed.

$$\mu = \frac{x_1 + x_2 + x_3 + \ldots + x_n}{n} (n > 0) \tag{3.1}$$

The variance described in formula 3.2, is one of the descriptors of a probability distribution, describing how far the numbers lie from the mean. It is used as a measure of comparing the variance of the datasets spread out from each other.

$$s^2 = \frac{(x_1 - \mu)^2 + (x_2 - \mu)^2 + (x_3 - \mu)^2 + \ldots + (x_n - \mu)^2}{n} (n > 0) \tag{3.2}$$

Standard deviation calculated in formula 3.3, shows how much variation between the mean or expected value and the dataset. A low standard deviation represents that the data points tend to be very close to the mean value, whereas high standard deviation indicates the data are spread out over a large range of values. This data value is very useful to represent the changes of finger force and finger-move speed from each user.

$$\delta = \sqrt{\frac{(x_1 - \mu)^2 + (x_2 - \mu)^2 + (x_3 - \mu)^2 + \ldots + (x_n - \mu)^2}{n}} (n > 0) \tag{3.3}$$

As shown in formula 3.4, the coefficient of variation is defined by the mean and standard deviation. Obviously, this data is used to indicate the relationship of mean and standard deviation when these two value features represent no positive correlation such as the dataset of location of finger points on the touch screen.

$$Coefficient\_of\_Variation = \frac{\delta}{\mu} \tag{3.4}$$

The skewness in statistics is a indicator to measure the extent of asymmetry of the probability distribution of a dataset. The skewness value could be positive and negative or even undefined. The negative skew means that the data area on the left side of the probability density function is longer than the right side and most of the values located to the right of the mean. Otherwise, a positive skew indicates that the data on the right side is longer than the left side and most of the values lie to the left of the mean. Wit a sample size n, the definition of skewness (γ) is described in formula 3.5. The skewness is used to measure the variance of spread of x-location of finger points in the full test.

$$\gamma = \frac{\frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^3}{(frac1n\sum_{i=1}^{n}(x_i - \mu)^2)^3/2} \tag{3.5}$$

3.7.2 Data Analysis Tools

In this project, a large amount of data is created and need to be analyzed for further comparisons. The Microsoft Office Excel is chosen as the main data analysis tool to deal with the values in each of the log file by applying several mathematical functions. A perl script is used to read all the log files and filter out the different type of touch points, press-down and press-move of finger points. Most of the graphs are drawn by excel with calculations of statistics, and several figures are created by a R script.

In probability theory, a probability density function (pdf) is a possibility function which describes the relative likelihood for a random variable value to occur at a given point. Formula 3.5 shows the definition of a pdf. The probability density distribution is the spatial distribution of the pdf. This is used to analyze the parameter of finger pressure force to show the distribution of the difference of finger force each user pressed on the touch screen.

$$F_X(a) = \int_{-\infty}^{a} f_X(x)\,dx.$$

(-$\infty < a < +\infty$)(3.6)

A histogram is a graphical representation, which indicates a visual impression of the probability distribution of a dataset. In this project, histograms are used to create figures of finger-move speed to show an easy visual impression of the distribution in the parameter of speed of finger movements. A histogram consists of frequencies and erected over discrete intervals (bins),

with an area equal to the frequency of the observations in the interval. The total area of a histogram equals 1.

Chapter 4

Results

This chapter covers the results of the experimental testing for the selected touch parameters. It details the distribution and comparison of the touch information with each user based on the observations, and gives insight on variation of the touch style of the users using the touch screen.

4.1 Data Collection Results

As discussed in chapter 3 section 3.5, 20 users participated in the full test. With different age, culture, reading habits and proficiency of using touch devices, these testing subjects used different time horizon to finish the reading material. Figure 4.1 shows the information about the reading time of each user in the full test. The data in this graph is sorted in descending order. The average time for these users to finish the same comic story is 15.35 minutes.

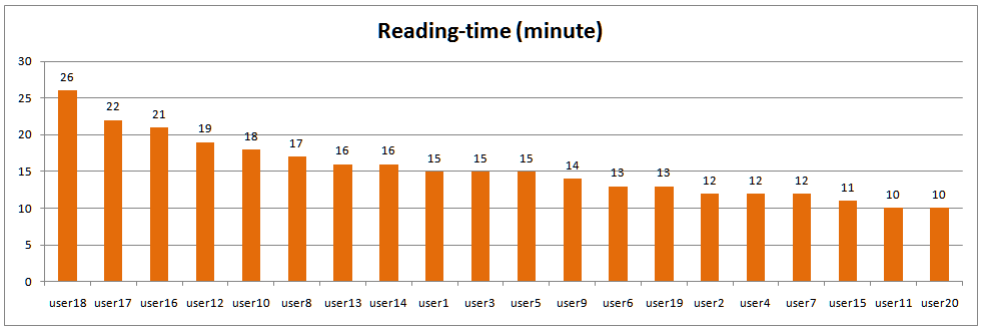


Figure 4.1: The reading time of each user in the full testing.

During the full test, 20 log files were created which stored the detail values of touch parameters from 20 different users. These original touch parameters including type of pressing, values of x-axis and y-axis of points on the screen, finger pressure, finger touch size, and the time of each touch event started, time of each touch event finished, represent the discriminate characteristics for each subject using the touch screen. With these primary data and calculations, the extracted parameters were obtained such as the counting of press-down

points on the screen, the displacement of finger movements in each sliding event, the finger pressure force of each touch point on the screen, and the speed of every finger-move point. The following are three touch information records of three finger events from one user.

```
press_type;point_x;point_y;pressure;point_size; \\
finger_force;downtime;eventtime;move_dx;move_dy; \\
speed_x;speed_y
press_move;389.33334;168.0;0.15686275;0.022222225; \\
0.0034858393;131902520;131903333;-2.0;-0.66667175; \\
-0.0024600246;-8.2001445E-4
press_up;389.33334;168.0;0.15686275;0.022222225; \\
0.0034858393;0;131903378;131903378
press_down;282.0;159.33334;0.15686275;0.04444445; \\
0.0069716787;131903542;131903542
```

In the following sections, 4 of the subjects in the full test are chosen to have a deep inside view on the detail results of each user. The overall comparison of the touch parameters for all the 20 user will be proceed later in the discussion chapter. One assumption of this project is that every person has a special touch style of using touch devices. Different origins, gender, age proficiency of using touch devices and different culture should make the personal touch style show more obvious varieties. The 4 users are respectively from Europe, Asia, Africa and South America. The personal information of these 4 subjects are shown in the table 4.1.

User	Origin	Gender	Age	Proficiency	Reading-time	Log File
User 6	Asia	male	25	Level 3	13 min	1334 K
User 14	Europe	female	31	Level 2	16 min	1906 K
User 17	South America	female	22	Level 3	22 min	1244 K
User 18	Africa	male	28	Level 1	26 min	1102 K

Table 4.1: Detailed information of the selected 4 users in full test.

4.2 Touch Parameter Results

All the assumptions and studying of the test results are based on the data values at hand. With the data collection results, analysis will go deep into the several parameters including the location of finger points on the screen, finger pressure force of each touch points and the speed of finger movements. These parameters are assumed to be the main factors of one user's touch style and could be used to analyze users' touch characteristics and compare the difference among individual users.

Press-down and press-move are two touch events corresponding to different finger activities discussed in chapter 3 section 3.3. Counting of these two

styles of touch points from the users and dividing them with the reading-time horizon these subjects used in the full test, the count number of each type of touch points in the unit time is obtained, which could show the preferences of the different users to the type of finger activities during the same time horizon. For example, table 4.2 shows the average count of the press-down points and press-move points in every minute from the selected 4 users. User 14 has fewest press-down points while the count of press-move points is the largest. This user appears to slide a lot but press very few times on the screen during the reading process. User 18 holds a large value of press-down points and a smallest value of the press-move points. This shows user 18 prefer to press more and slide few on the screen compared to the other 3 users. A value is created to indicate the character of the subjects using the touch screen, by using the number of press-move points divide the count of press-down points.

User	Press-down Point (d)	Press-move Point (m)	m/d
User 6	29	694	23.93
User 14	14	841	60.07
User 17	41	350	8.54
User 18	34	257	7.56

Table 4.2: Average Count of press-down and sliding-move points in every minute for the selected 4 users.

As shown in figure 4.2, the average count of the press-move points every minute is set to x and the number of press-down points is set to y. With the values of x and y, the location of one point can be determined in the graph. In figure 4.2, 20 points are created based on the data of 20 users in the full test to illustrate the different habits of these users using touch screen. The points, which are far away from each other, means these two subjects have very different preferences of finger touch style such as user 6 and user 19. Similarly, shorter distance between two users means their touch habits are closer to each other such as user 10 and user 16.

The following sections will analyze and compare the parameters including location of touch points on the screen, finger pressure force and the speed of finger movements based on the collected touch information from the selected 4 uers. The information about touch parameters were represented in chapter 3 section 3.3.

4.2.1 Finger Points Location

The parameters point_x and point_y stand for the x-axis and y-axis values of the location of each touch point on the screen. Once the users touch the screen, one finger point is created and the value of location on the touch screen from this point is recorded and stored into the log file. With these two parameters, the graph of finger points' locations during the full test from the 4 users can be created to show which part of the screen every user tend to touch.

Figure 4.3 shows all the touch points on the screen from user 6 during the reading process. The graph reveals that, user 6 probably only used one hand

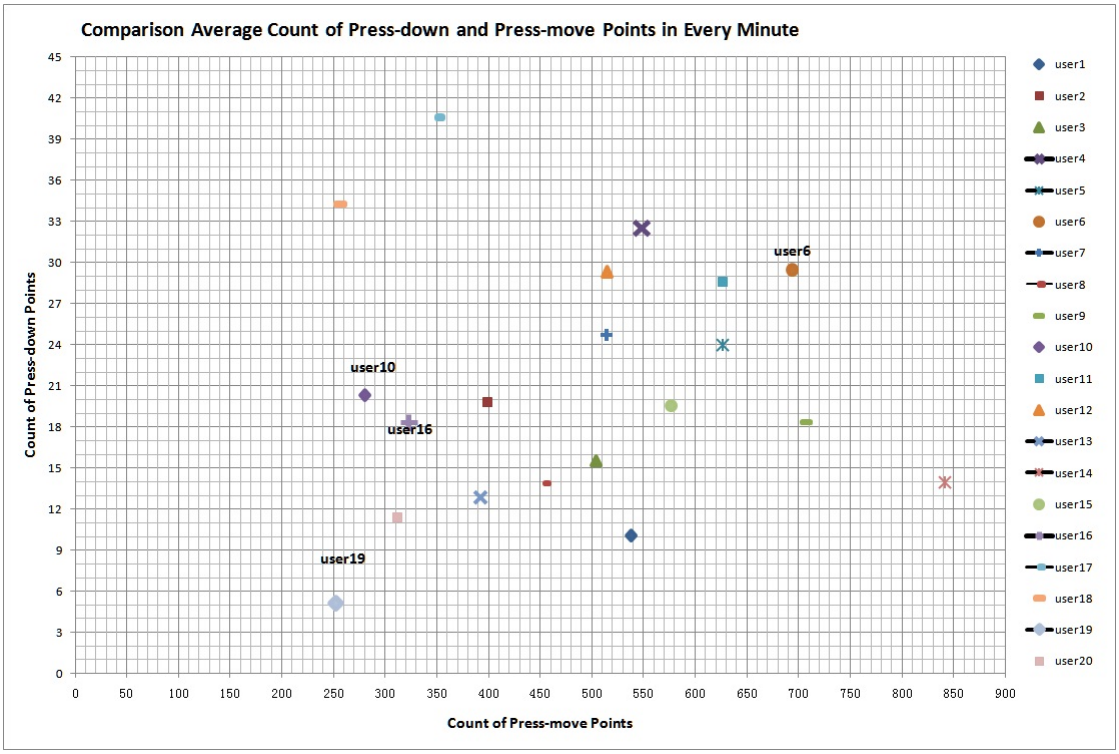


Figure 4.2: Comparison average count of press-down and press-move touch points in every minute from 20 users.

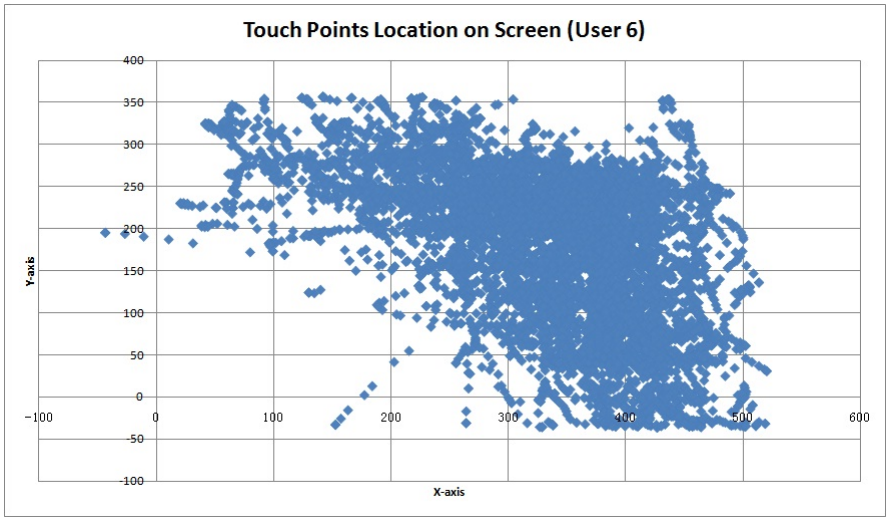


Figure 4.3: The finger points location of user6.

to touch the screen and the finger points are very scattered which covers a lot of area of the screen in addition to the lower left part. Most of the points are located in the area of x-axis values from 250 to 450, y-axis values from 100 to 300.

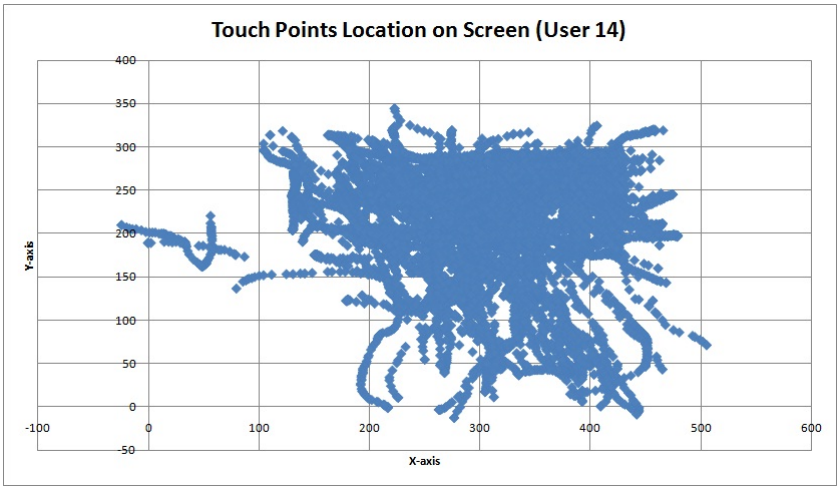


Figure 4.4: The finger points location of user14.

As shown in figure 4.4, user 14 has more continuous points, which means this user slided a lot on the screen so that more finger press-move points were created. Most of the touch points are located in the center of the screen and the area is between 200 to 400 in the x-axis and 150 to 300 in the y-axis.

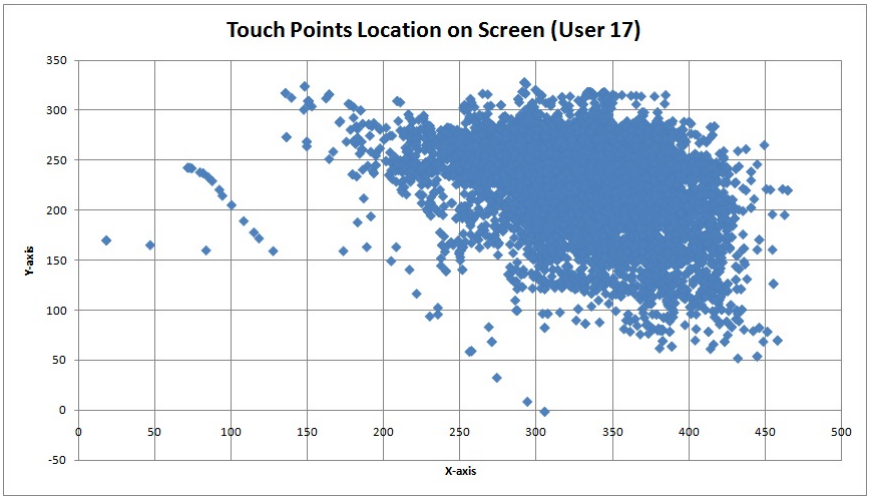


Figure 4.5: The finger points location of user17.

The information about location of touch points from user 17 is presented in figure 4.5. As shown in the graph, user 17 has more individual points compared to user 14 and most of the points are concentrated in the area of upper right on the screen. This user prefer to use the screen area of x-axis range from 250 to 400, y-axis range from 150 to 300.

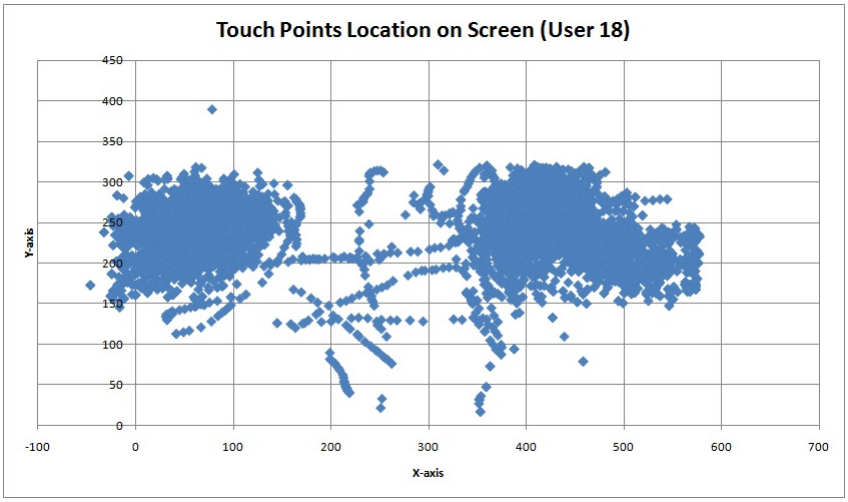


Figure 4.6: The finger points location of user18.

There are two obvious concentrated areas of touch points in figure 4.6, which indicates that the user 18 used both hands to touch the screen during the reading process. The two touch areas are located in the middle of the screen on both left and right side. User 18 has a lot of sliding-move touch points which are centralized in these two areas on the touch screen.

From the locations of the touch points in the above figures, these 4 users prefer to use different part of the touch screen. User 6 touched the largest area of the screen comparing to the other three users. Summarily, the touch points from user 6 are imploded in the lower right area of the screen. User 14 obviously has more sliding-move points than the other three users and most of these points are located in the center of the screen. The touch points from user 17 are very concentrated in the upper right area of the screen, which appears this user has the smallest finger-activity area on the tablet in these 4 selected users. User 18 uses fingers from two hands during the full test so that there are two different centralized areas of touch points located individually on the left and right side of the screen. User 6 and user 17 have a lot of single press-down points, which represents these two users prefer to press down the screen during the reading process. While user 14 and user 18 like to slid on the screen with fingers so that they created more sliding-move points. User 6, user 14 and user 17 probably only used fingers from one hand since the touch points from them were not significantly distributed in different regions of the touch screen.

As shown in figure 4.3, figure 4.4, figure 4.5 and figure 4.6, there are large difference for the touch points in x-positions. To make a better comparison, the focus will put on the x-location of the touch points and the y-coordinates are ignored because of the small difference for each user in this axise. For each of these 4 users, the mean, standard deviation and skew values of x-location are calculated to show the features of the dataset of finger locations. The mean

value is the centroid in x-coordinates of touch point locations. Standard deviation shows the variance of the each point's x-location with the mean value. The skewness measures the asymmetry of the probability distribution of a real-valued random variable. These values features were discussed in chapter 3 section 3.7. Table 4.3 represents the mean, standard deviation and skewness values of the 4 users' x-location of touch points.

	user6	user14	user17	user18
mean	336.452	307.897	334.752	252.390
standard deviation	93.928	78.130	48.943	186.623
skewness	-0.877	-0.419	-0.712	0.028

Table 4.3: Comparison of mean, standard deviation and skewness of x-location of touch points.

As shown in the table 4.3, user 18 has the smallest mean and the largest standard deviation, which means the centroid point of the touch point area from this user has a smaller x-axise value than the other three users and the location of the points are very different. This is because user 18 used two hands to touch the screen during the reading process, which is presented in figure 4.6. Figure 4.7 is a scatter plot of the mean and standard deviation x-position of touch points from these 4 users in the full test. The point of user 18 is on the top left of the graph which is far away from the other three points. This indicates user 18 has a very special touch character in touch location of the screen. There is large variation among user 6, user 14 and user 17, which can be used to distinguish the users.

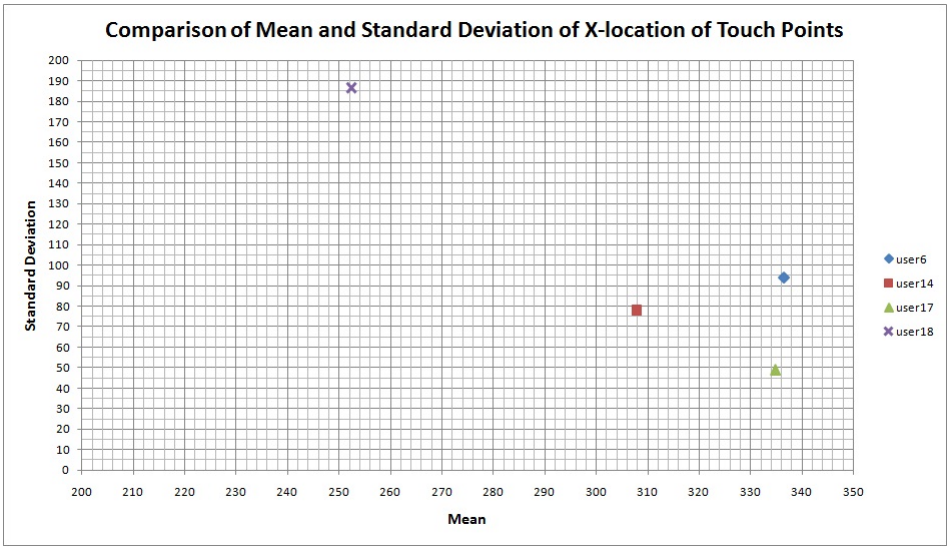


Figure 4.7: Comparison of mean and standard deviation value of x-location of touch points from 4 users.

The skewness of x-location of touch points from these 4 users are shown

in figure 4.8. Only user 18 has the positive skewness which means most of the points from user 18 are lying to the left side of the mean. The other three subjects with the negative skewness value indicate that the touch-point area on the right side of the centroid is longer than the right side and the bulk of the points lie to the right of the x-location mean. Looking at the figure 4.8, user 18 has a very small spread area of touch points and user 6 make the largest spread touch area on the screen. The mean, standard deviation and skewness with huge difference to different subjects give a possible way to differentiate individual users by the touch point locations.

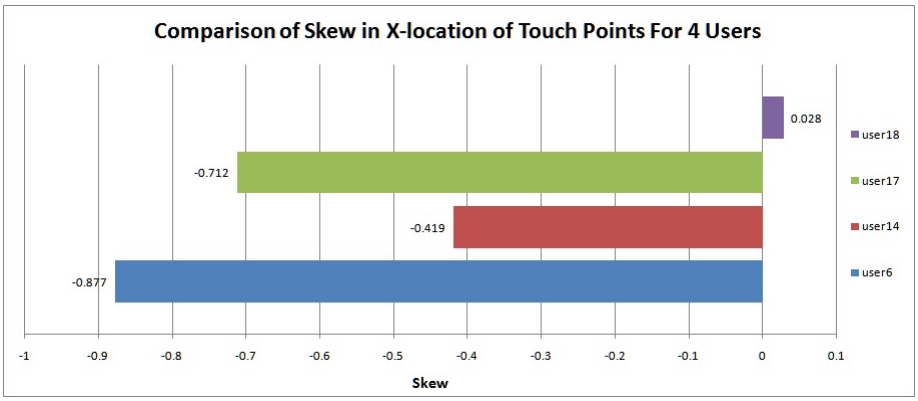


Figure 4.8: Comparison of skewness value of x-location of touch points from 4 users.

4.2.2 Finger Pressure Force

The parameter of finger pressure mentioned in chapter 3 section 3.3 generally ranges from 0 to 1. Value 0 means no pressure at all and value 1 means normal pressure on the screen. The differentiation of the pressure from different users are too small to compare. And some of the touch points hold the same value of finger pressure, which is unexpected. By composing the pressure and the size of each touch point, the finger pressure force can be obtained. To make the data easily comparable, the values of pressure are expanded 1000 times to change the unit of this parameter from 1K Pascal (KPa) to Pascal (Pa).

$$1KPa = 1000Pa \tag{4.1}$$

$$Force = Pressure * Size \tag{4.2}$$

Figure 4.9 shows the finger pressure force for all the touch points from user 18 during the reading process. Most finger force of the touch points is under value 15. The maximum number is about 28 and the minimum value is about 3. The average force of the touch points from this user is 12.12. And the standard deviation of the finger pressure force from this user is 4.6. Based on

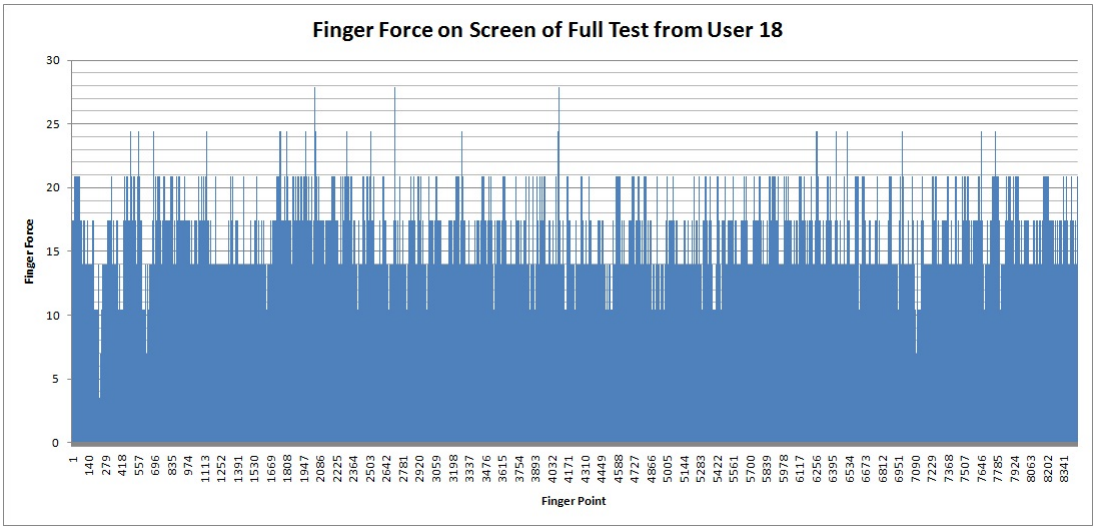


Figure 4.9: The values of finger pressure force for each touch point from user 18.

this original data, the probability density distribution of finger force for touch points from user 18 is shown in figure 4.10.

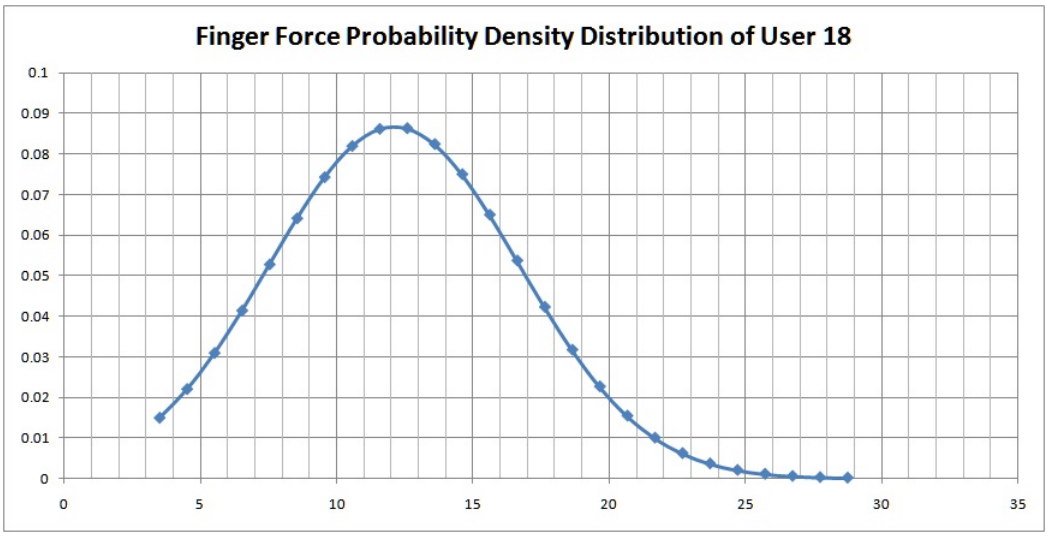


Figure 4.10: The probability density distribution of finger pressure force for touch points from user 18.

As shown in figure 4.10, the probability density distribution line starts from 3.5 and ends in 29. The percentage of the finger pressure force from touch points ranges between 0 to 0.09. The highest point in the graph achieves around 8.8

Figure 4.11 shows the comparison of probability density distibution of finger pressure force from the 4 selected users. The maximum finger force from user 6 is about 14, the minimum finger force is about 3.5, and the mean value is

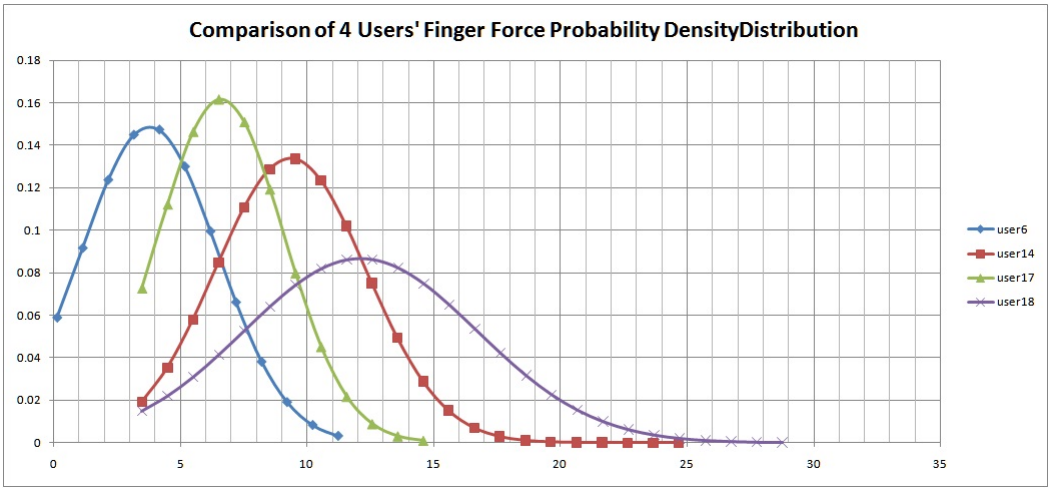


Figure 4.11: The probability density distribution of finger pressure force for touch points from user 6, user 14, user 17 and user 18.

about 7. The maxmum, minimum, and mean values for user 14 are about 24.5, 3.5, and 9.5. User 17 holds 14, 3.5, and 6.5 for the maxmum, minimum, and average values of finger force for touch points. Looking at figure 4.11, each line is different from each other. The values of mean and standard deviation for each user are distinct based on the different probability density distributions. These lines show the 4 users have unique characteristics of the finger pressure force on touch screen during the reading process.

User	Maximum	Minimum	Mean	Standard Deviation	Coefficient of Variation
User 6	13.943	3.486	7.139	2.680	0.375
User 14	24.400	3.486	9.360	2.983	0.319
User 17	13.943	3.486	6.610	2.469	0.374
User 18	27.887	3.486	12.12	4.600	0.380

Table 4.4: Data analysis of finger pressure force for touch points from the 4 selected users.

Table 4.4 shows the detail data features of the finger force for touch points from user 6, user 14, user 17 and user 18. User 18 has the largest finger force value in these 4 users. User 6 and user 17 share the same maximum value. All the users have the same minimum value but the mean and standard deviation values are differnt from each other. The average finger force on the screen from user 17 is the smallest one compared to the other subjects, which is almost half of the value of user 18. User 17 appears to press very gently in the full test. And user 18 provides the larggest mean value which indicates this user press heavily on the screen. The coefficient of variation is determined by mean and standard deviation, which is used to compare the degree of difference between two data sets. This indicator can show how different in the factor of finger

force on the screen when the users are using touch devices.

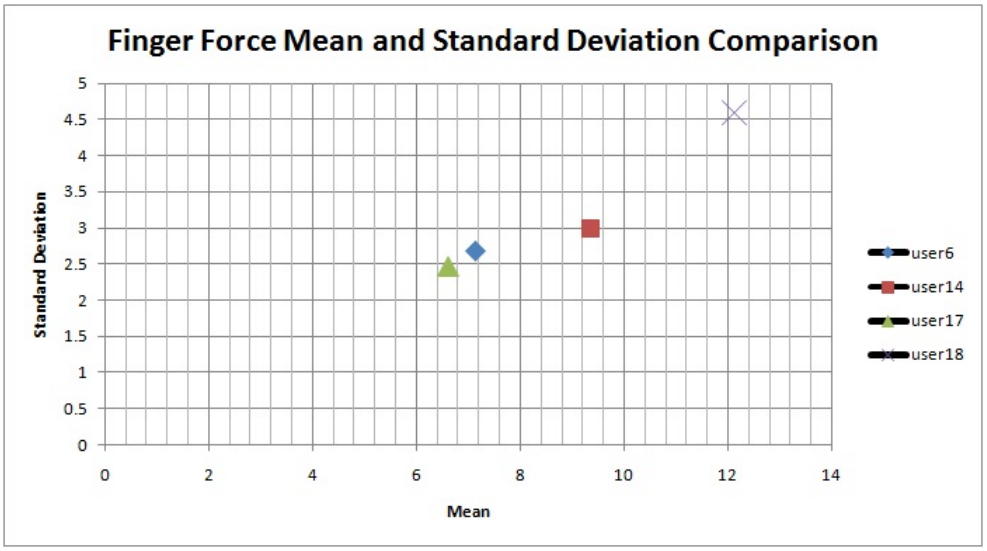


Figure 4.12: Comparison of the coefficient of variation of finger force on touch screen in user 6, user 14, user 17 and user 18.

As shown in figure 4.12, the points of user 6 and user 17 are very close to each other. These two users used the similar finger pressure force when they touched the screen. The coefficients of variation for these two users are 0.375 and 0.374 which are also very value-close. The point of user 18 in the top right of the graph is far away from the other three points. This figure shows that user 18 used bigger finger force during the full test than other subjects. Summarily, large mean value seems corresponding to a large standard deviation for the points in the figure 4.12. This is a feature worth noticing. More focus will be applied on this later for further discussion.

Figure 4.11, table 4.4 and figure 4.12 represent a considerable value difference in the data features of finger pressure force on the screen among the selected 4 users. This gives a support to hope differentiations in the parameter of finger pressure force on touch screen to distinguish the different users.

4.2.3 Finger-drag Speed

The finger event of dragging on the screen creates the press-move touch points during the reading process. Each finger movement on the screen is composed by a number of press-move points. The data of displacement in two axises (dx, dy) and duration (t) is available in the log file for each point. The displacement value of every touch point (X) relative to the coordinate origin of the screen is calculated through arithmetic sum of squares. The speed of each point (V) is defined by the distance traveled per unit time, which directly indicate how fast the users slided on the touch screen.

The displacement value of one press-move point (X) in the dataset can be formulated as:

$$X = \sqrt{(dx)^2 + (dy)^2} \tag{4.3}$$

The overall speed value of each press-move point (V) on the screen can be described as:

$$V = \frac{X}{t} \tag{4.4}$$

User 6

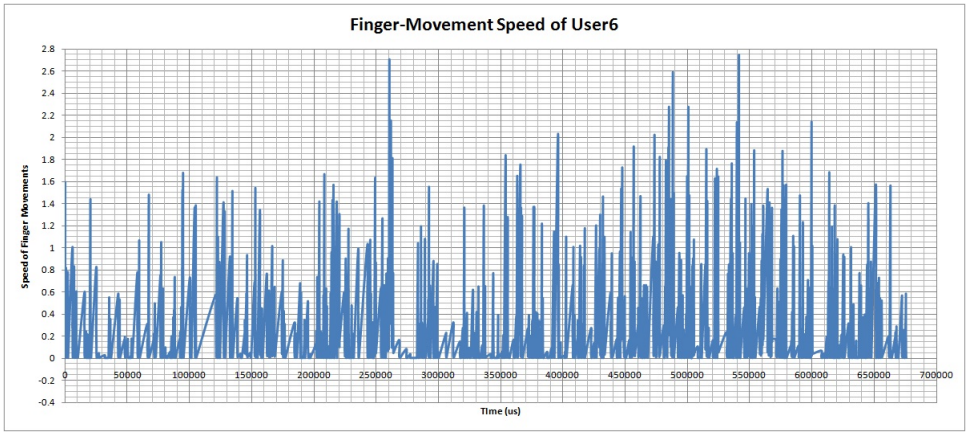


Figure 4.13: Graphical representation of the finger-movement speed of user 6 in full test.

Figure 4.13 shows the speed of all the press-move points from user 6 in the reading process from 0 to 68000 microseconds (μs). A range from 0 to 2.8 speed of finger movements is plotted in the graph. Notice that there are some of the points located in the value 0 because the user 6 stopped and held the finger on the screen during one sliding move event and then continued to move the finger with out leaving the touch screen. The maximum speed is around 2.7 and the changes of the point speed is not very large. This user appears to be a active and fast reader.

To show an easy visual impression of the distribution, a histogram is created (figure 4.14). This shows an estimate of the probability distribution of the speed values. The speed of finger movements from user 6 is most frequently in the range from 0.00012 to 0.179. The maximum of the frequency is about 5000.

User 14

User 14 (figure 4.15 and figure 4.16) has the speed of finger sliding movements values from 0 to 0.8 which is a much smaller range than the user 6. Looking

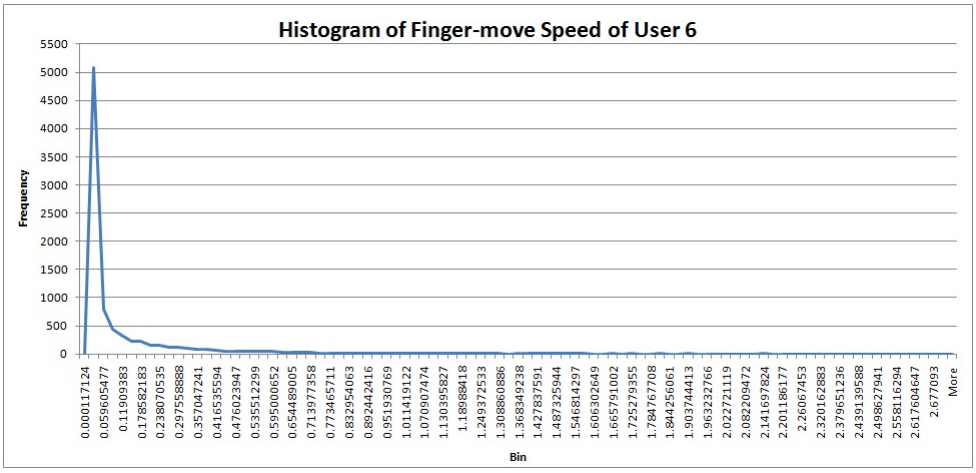


Figure 4.14: Histogram of the finger-movement speed of user 6 in full test

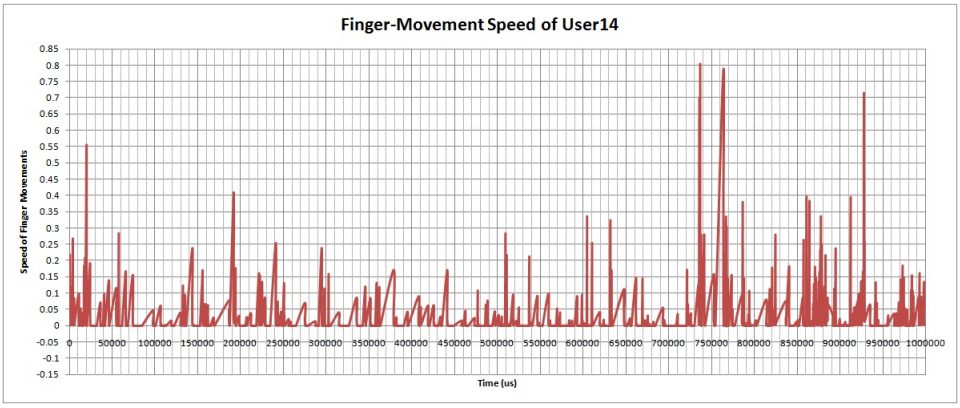


Figure 4.15: Graphical representation of the finger-movement speed of user 14 in full test.

at the time line, the graph offers values from 0 to 1000000 μs. This indicates a longer finger sliding time in total which is used by user 14 in reading process than the previous user. This user appears to be a very careful but slow reader.

The histogram of the finger-move speed from user 14 is not that different from the user 6 on the visual trend of the line. Looking at the frequency, however, figure 4.16 provides values from 0 to about 10000. This is almost double times than user 6.

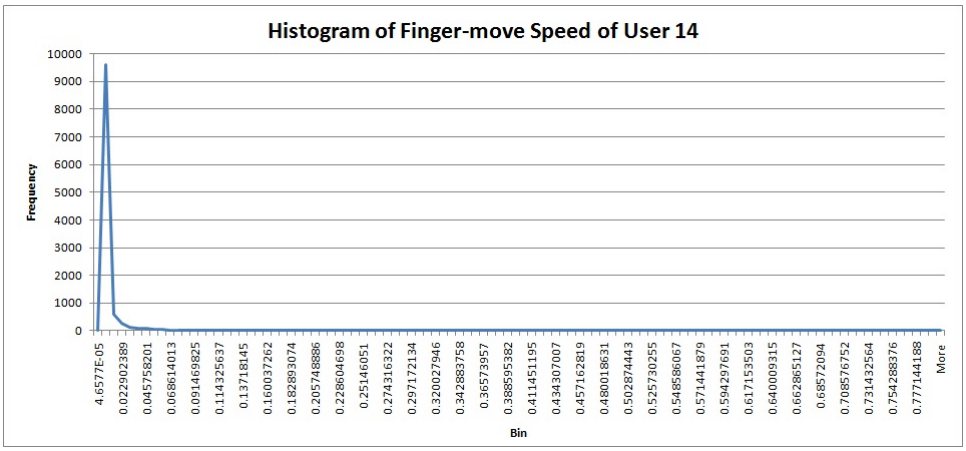


Figure 4.16: Histogram of the finger-movement speed of user 14 in full test.

User 17

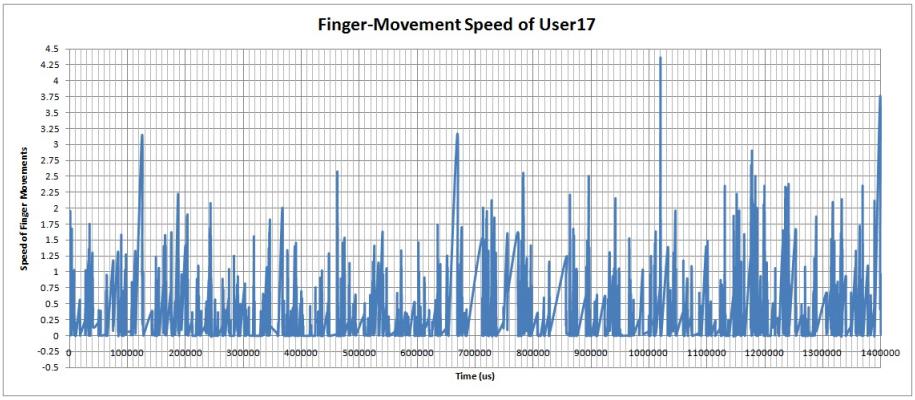


Figure 4.17: Graphical representation of the finger-movement speed of user 17 in full test.

Figure 4.17 shows a range from 0 μs to 1400000 μs time for speed of finger movements from user 17. This is a longer time of finger sliding in total than the user 6 and user 14 used in the full test. The speed of the finger-move points have values from 0 to about 4.5, which is a much larger range than the previ-

ous two subjects.

As shown in figure 4.18, the overall appearance of the user 17's histogram is familiar with user 6 and user 14. Looking at the specific values, in fact the maxmun value of frequency is lower than the other two subjects. The points with extremely small numbers are not given proper emphasis in the histogram figures due to the extreme values. But these values have very small effect on the results which can be ignored.

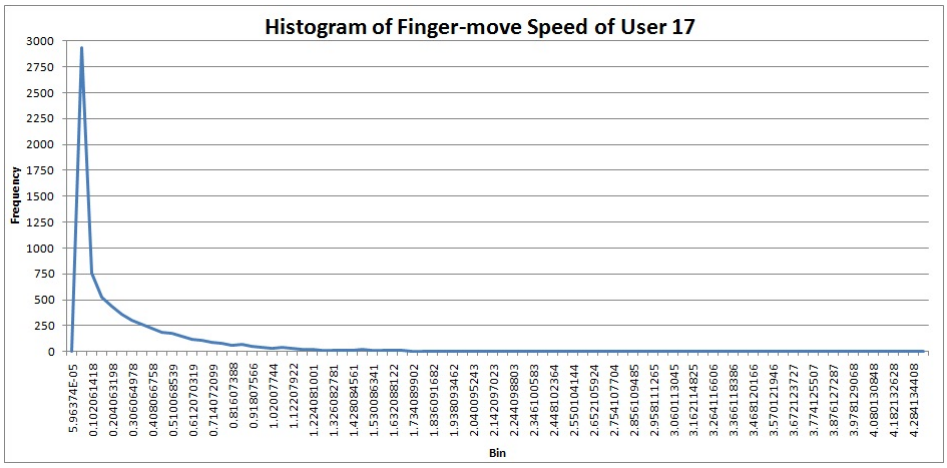


Figure 4.18: Histogram of the finger-movement speed of user 17 in full test.

Compared to user 6 and user 14, the results from user 17 show a huge value difference in speed of finger-movements and the frequency of the value points. This gives reason to hope a difference in finger sliding movements which would differentiate the users.

User 18

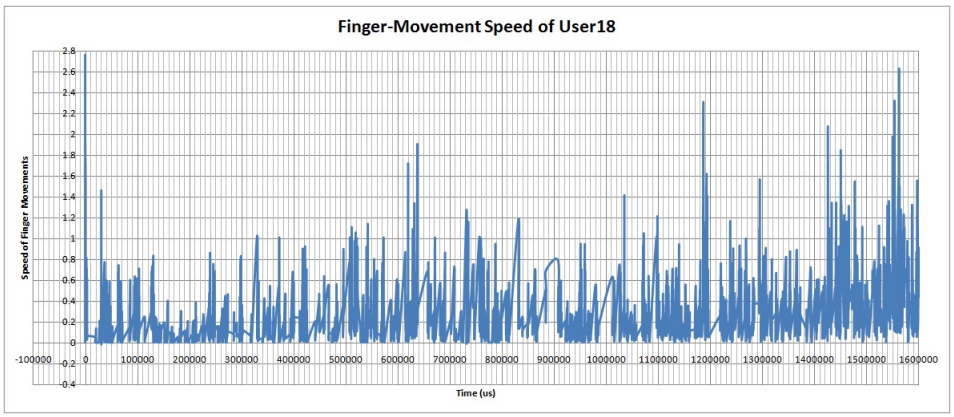


Figure 4.19: Graphical representation of the finger-movement speed of user 18 in full test.

The last user in the selected group, user 18 (figure 4.19 and figure 4.20) shows a finger-move speed range from 0 to 2.8 which is the same value range as user 6. Looking at the detail values, most of the points are smaller 1 while speed of finger movements from user 6 are mostly blew 1.6. Then the mean value can be a feature to distinguish these two users. The time line is from 0 μs to 1600000 μs. This denotes user 18 used the longest total time of sliding finger movements compared to the other three subjects in the reading process.

Histogram of speed of finger movements from user 18 is shown in figure 4.20. The frequency values are from 0 to 1800 which is the smallest value range in the selected 4 users. This indicates the values are not highly concentrated in the dataset and user 18 has a relatively large changes in velocity of finger movements on the screen.

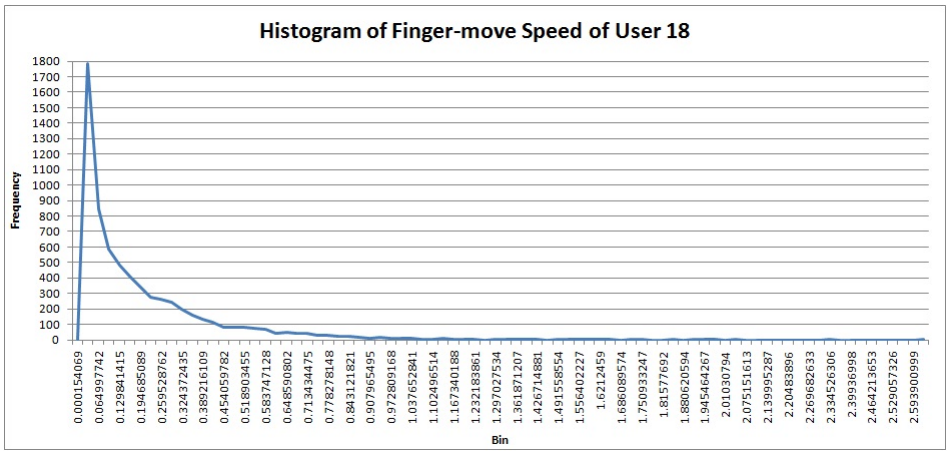


Figure 4.20: Histogram of the finger-movement speed of user 18 in full test.

User 18 has the longest time used in sliding on the screen and the lowest frequency appeared in the histogram of finger-move speed among the selected 4 users. This user appears to be a very active but slow reader.

Sample Comparison

The data features of finger-sliding speed from user 6, user 14, user 17 and user 18 in the full test can be found in table 4.5. The points of finger hold during each sliding-move event make the minmum value same for all users. To solve this, values 0 were removed from the datasets. The maximum speed values range from 0.8 to 4.335. The minimums velocity of speed are 0.00012, 4.658E-05, 5.964E-05 and 0.00015 individually corresponding to user 6, user 14, user17 and user 18. The mean values spring from 0.006 to 0.241. Looking at the standard deviation, it proves to be between 0.025 to 0.355. The coefficient of variation is in the range from 1.2551 to 4.054.

As shown in table 4.5, user 14 has the smallest maximum, minimum, mean and standard deviation values in the 4 samples. This user appears to slide very slowly on the touch screen compared to the other three subjects. The

User	Maximum	Minimum	Mean	Standard Deviation	Coefficient of Variation
User 6	2.737	0.00012	0.111	0.236	2.124
User 14	0.8	4.658E-05	0.006	0.026	4.054
User 17	4.335	5.964E-05	0.241	0.355	1.475
User 18	2.691	0.00015	0.184	0.231	1.255

Table 4.5: Comparison of data features from speed of finger movements from the 4 selected users in full test.

feature values from user 6 and user 18 are close to each other. These two users emerge to have similar reading habits with finger movements on the screen. User 17 has the largest changes from maximum speed to minimum speed in the selected 4 users. The average of the finger speed from user 17 is also much larger than the other three users. This user appears to slide very fast with finger on the tablet.

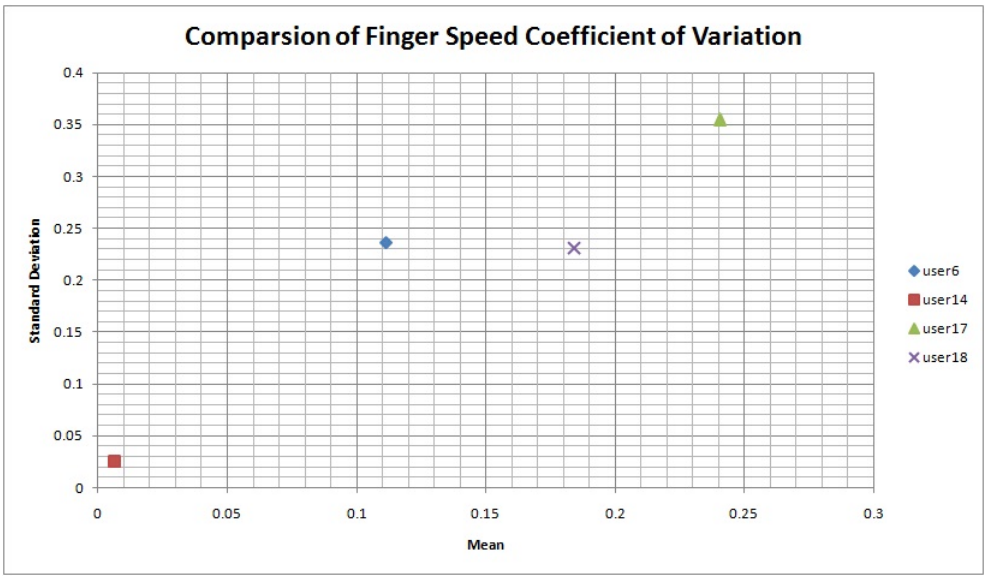


Figure 4.21: Comparison of coefficient of variation in the selected 4 users.

A representation of coefficient of variation from these sample data is shown in figure 4.21. User 14 is in the lowest position because of all the smallest feature values in table 4.4. User 6 and user 18 are in the familiar positions and hanged in the same level of standard deviation. These confirm the two subjects have the similar finger-move habits during the reading process. User 17 is in the top of the figure with a higher coefficient of variation than other three users.

Table 4.5 and figure 4.21 shows a large value difference in the data features of finger-move speed on the screen among the selected users. This gives a reason to hope a method to differentiate the users based on the difference in finger movements.

Chapter 5

Discussion

This chapter covers the analysis of the experimental results and discussion about some of the challenges found in the project. Since the goal of the thesis is to find the difference of touch characteristics of individual users using touch devices, it will discuss the parameter features for each sample to show their touch characters. It will also involve suggestions of improvement for some weaknesses of this project.

5.1 Data Analysis

As discussed in the previous chapter, the subjects in the full test have huge different characteristics in several touch parameters including counting of different type of touch points, touch point locations, finger pressure force and finger-move speed. The data features for datasets of these parameters show the touch styles of these individual users and indicate how to distinguish these users using the touch device.

5.1.1 X-location of Touch Points

Figure 5.1 represents the mean values of the x-position of touch points from the 20 subject in full test. The y-coordinates are ignored because of the small difference for each user in this axis. The x-position is the only consideration of the horizontal coordinate which can show the bimanual usage as it will have a much larger spread.

Looking at figure 5.1, the mean and standard deviation values are plotted. All the users hold different feature value of their datasets. The average values of x-position of touch points show the centroid of each series. The larger value means the more right the centroid of this serie located on the screen. User 1 has the largest value which is over 380 in x-axise while user 4 holds the smallest value less than 150. There is a particular user sample has standard deviation value worth noticing. User 18 has a largest value compared to the other subjects, which means this user holds the biggest different dataset values of x-position in the full test. Looking at the figure 4.8 in chapter 4 section

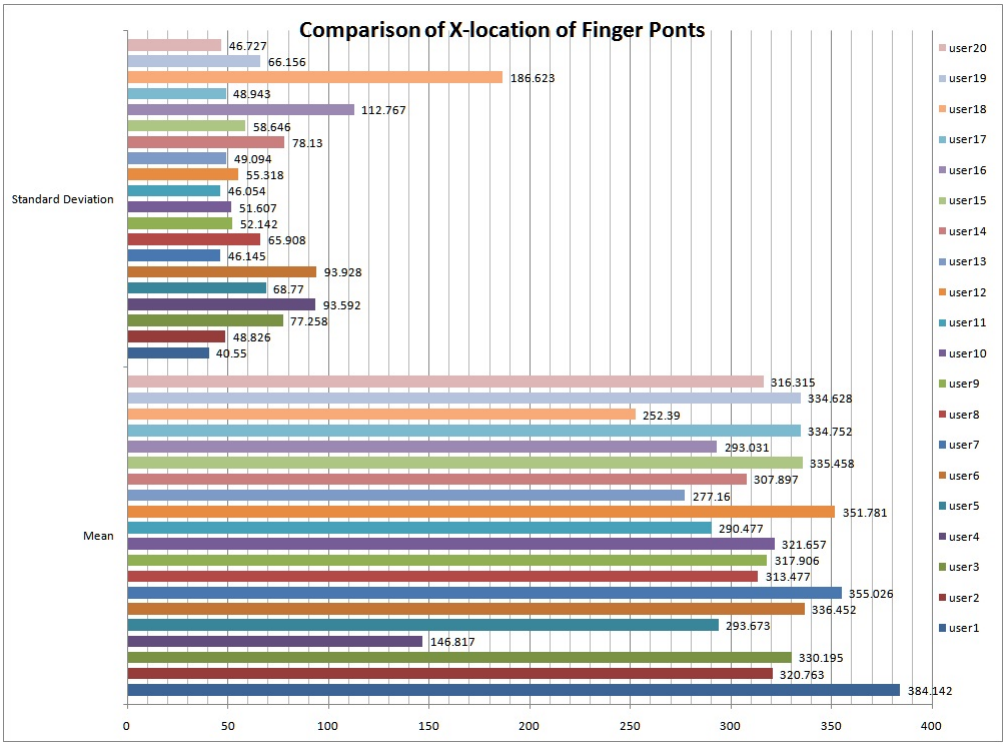


Figure 5.1: Comaprison of x-positions of touch points in full test.

4.2, user 18 used the fingers from two hands during the reading process. This value confirms the bimanual usage of the touch device.

Figure 5.2 shows a scatter plot of the mean and standard deviation x-position of touch points from 20 users in the full test. It reflects the coefficient of variation values of each subject and gives a intuitive way to see how different in the parameter of x-location values of these touch points from the 20 users. User 1, user 4 and user 18 are the three points which have extreme positions in the graph because of their huge difference in the style of touching the screen. User 18 has a much larger standard deviation value than the other subjects, which means the x-location of the points from user 18 are very different. Large value of standard deviation is caused by the big spread area of touch points on the screen, which indicates the user probably used two hands to touch the screen during the reading process. Mean values shows the centroids of touch points' area for the users. As shown in figure 5.2, user 1 has the largest mean value which represents most of the touch points from this user are located on the right side of the screen. The smallest mean value of x-location for touch points is from user 4. In this graph, each user has a unique point with mean and deviation values which can be used to show the touch characteristics of individual users.

The skewness values of touch points' x-locations from the 20 users in full test are shown in figure 5.3. These values show the degree of the spread of the touch points from 20 users in x-location. Looking at the graph, there are 5

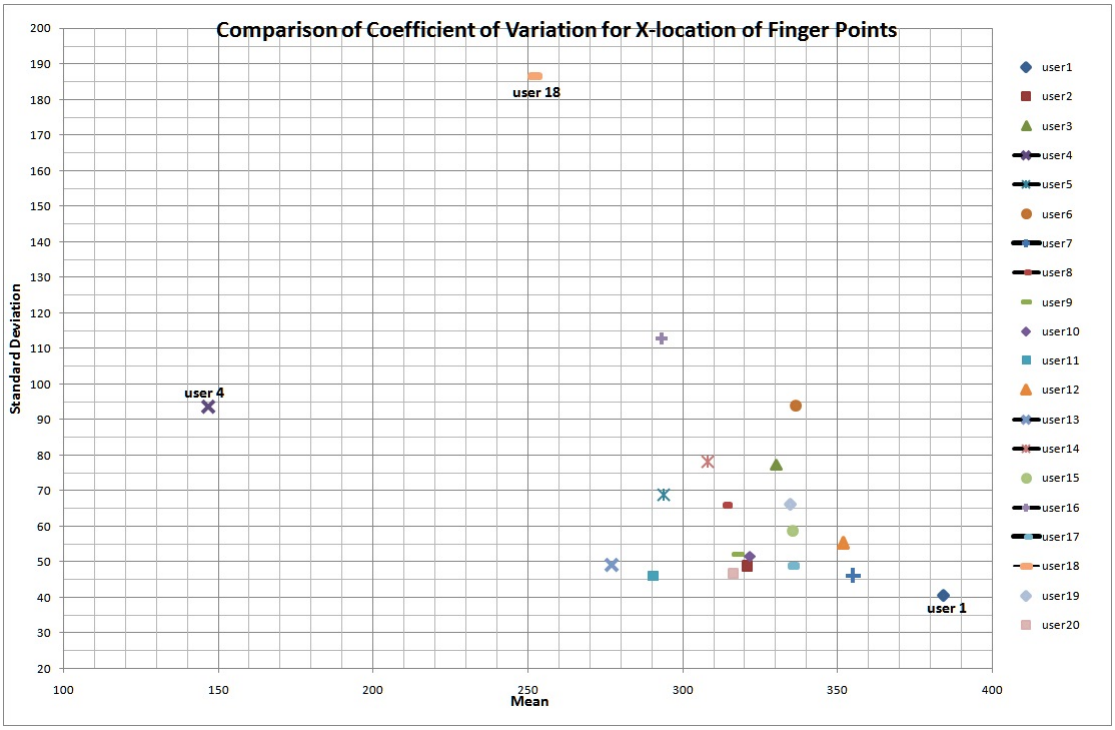


Figure 5.2: Coefficient of Variation Comparison of x-positions of touch points in full test.

users have the positive skewness values, which means the touch points from these users are mostly lying to the left side of the centroid (mean). Larger skew value indicates the touch points have a greater overall deviation relative to the mean value in x-location. 15 users prefer to put their touch points to the right side of the centroid of the touch points area. The skewness values have a huge difference among these subjects, which is a good feature of x-location parameter can be used to distinguish different users.

Summarily, the mean, standard deviation and skewness are chosen to be the data features of x-location parameter of touch points to differentiate individual users with significantly value difference.

5.1.2 Finger Pressure Force

Figure 5.4 summaries the mean and standard deviation values of finger pressure force from the 20 users. As shown in the graph, no subject share the same feature values with another individual user. User 1 and user 18 have a larger average finger force compared to the other subjects, which indicates these two users have the habit to press heavily on the screen. The average finger force on the screen from user 17 is the smallest one in this figure, which is almost half value of user 18 and one third value of user 1. User 17 appears to prefer pressing the screen very gently when use a touch device.

Figure 5.5 shows the comparison of coefficient of variation for finger pressure force of the 20 subject in the full test, which describes how different the

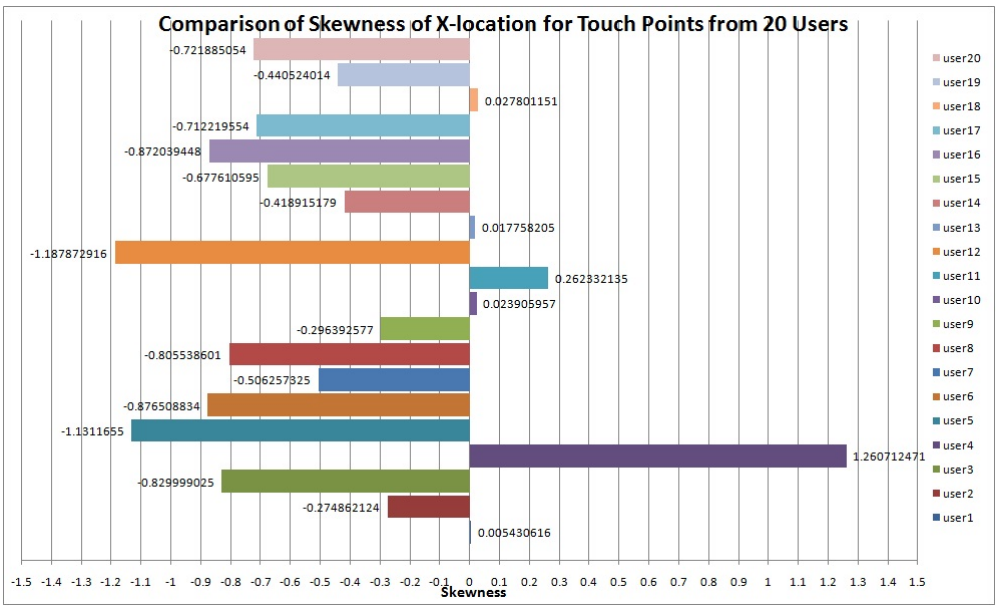


Figure 5.3: Comparison of skewness of x-location of touch points from 20 users.

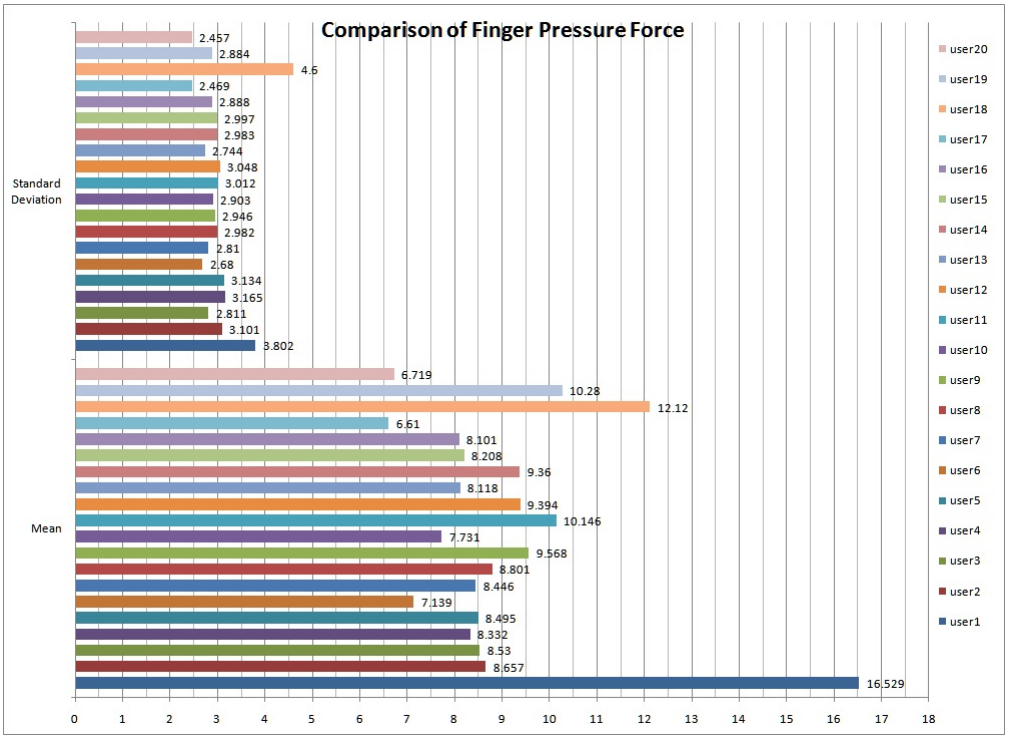


Figure 5.4: Comparison of finger pressure force from touch points in full test.

finger force each user used to touch the screen. User 1 and user 18 are the two points which hold the positions far away from the other users. These two users obviously have the habit of pressing the touch screen heavily. The points of other subjects are concentrated in a area, from 6.5 to 10.5 in mean and standard deviation ranges from 2.5 to 3.25. There are no two points covering with each other. All the users have different feature values in this parameter to show their different characteristics of finger force when use a touch device.

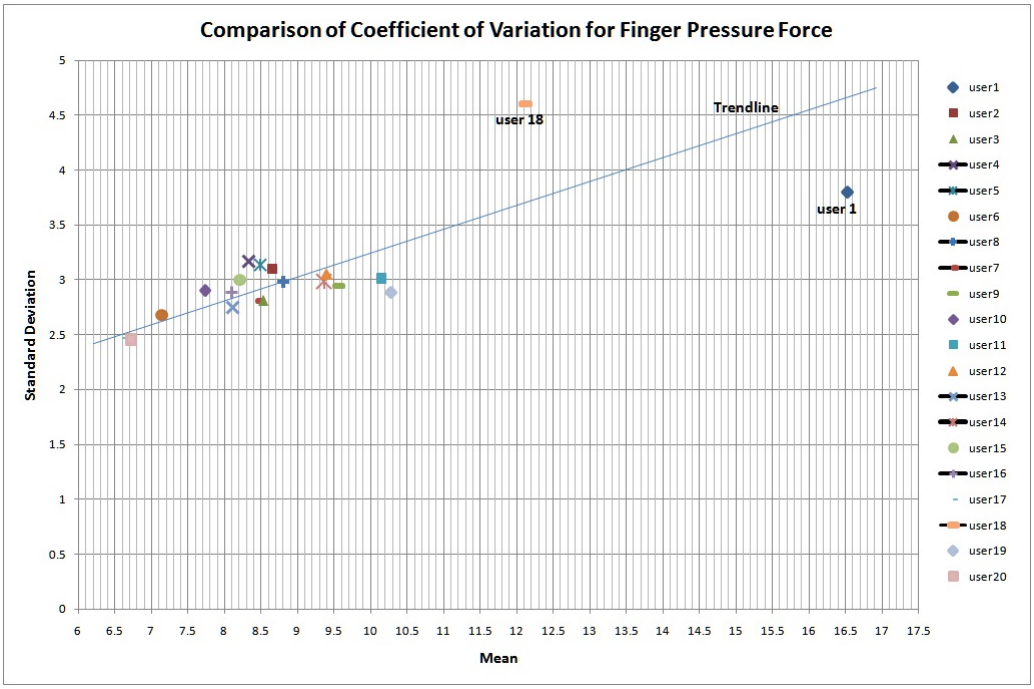


Figure 5.5: Coefficient of Variation Comparison of finger pressure force for 20 users in full test.

Looking at figure 5.5, the trends of these user points are worth noticing. Large mean value are corresponding to a large standard deviation for the users in the graph. One probable trendline is applied and shows a linear growth of mean and standard deviation trend. Since these two value features have positive correlation, only one value is enough to identify users from each other. The mean value of finger pressure force is chosen as the feature of finger force parameter to differentiate users.

5.1.3 Speed of Finger Movements

The average and standard deviation values of finger-move speed from the 20 subjects in the full test are shown in figure 5.6. User 17 and user 18 have an extremely larger mean value of finger-move speed than the other users. These two users obviously prefer to slide fast on the screen during the reading process with a touch device. Strongly contrasted, user 1 and user 14 hold a very small average of finger-move speed compared to the other subjects. Especially for user 14, the mean value of this parameter is almost 23 times smaller than

user 17. User 1 and user 14 appear to have the habit of sliding very slowly on the screen when use a touch device. Looking at the values of standard deviation in the graph, the maximum value is given to user 17. User 3, user 6 and user 18 are in the second largest value ranges. These 4 users change their speed of finger movements a lot when they are reading through a touch device in the full test. User 1 and user 14 are the two subjects hold a smaller value of standard deviation of this parameter, which means these two users not only prefer to sliding slowly on the screen but also have the habit to maintain a relative smooth speed during the reading process.

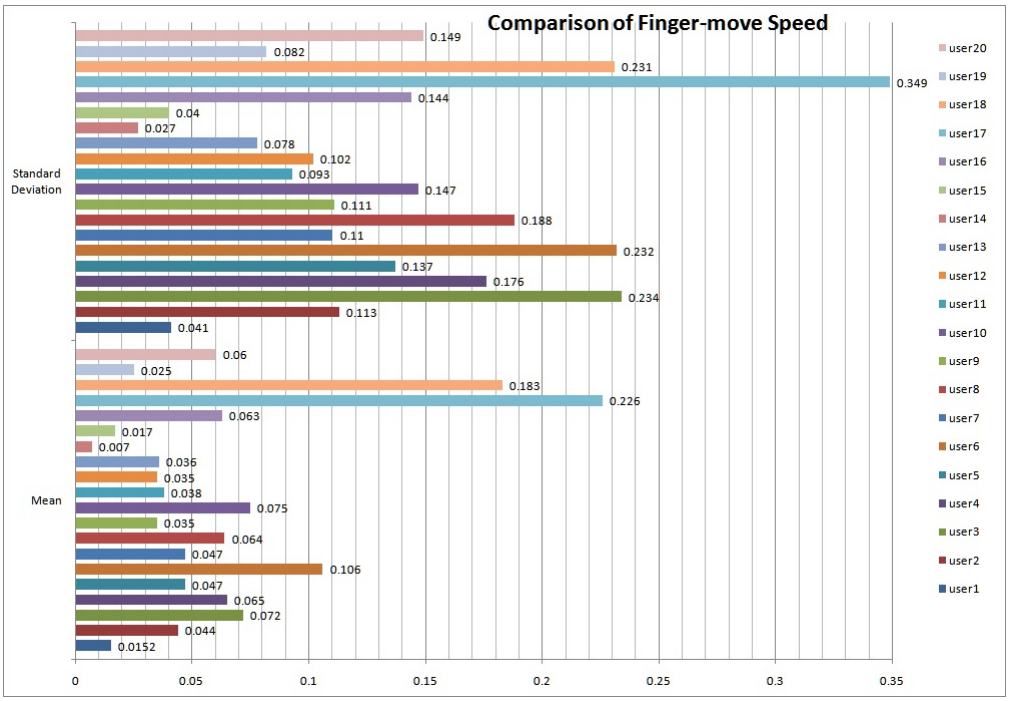


Figure 5.6: Comparison of finger-move speed for the 20 subjects in full test.

As shown in figure 5.7, the mean and standard deviation values of finger-move speed for these 20 users are ploted to show the different coefficient variation for each dataset. The user 17 point is on the top right of the graph which is far away from the area the other points centralized by holding the largest mean and standard deviation values. This user is proved to slide very fast on the touch screen and prefer to change the speed frequently during the reading process. User 6 and user 18 also have a very large average finger-move speed in the full test compared to other subjects. User 3, user 6 and user 18 are in a similar level on vertical axis because of the closed standard deviations. They are also like variable speed when use a touch device. The user 14 point, located on the lowest left of the graph, shows a habit of slow and stable speed of finger movements on the screen during the reading process in the full test.

The trendline in figure 5.7 shows the mean and standard deviation are positive correlated. The mean of finger-move speed is chosen as one of the feature parameters to discriminate users.

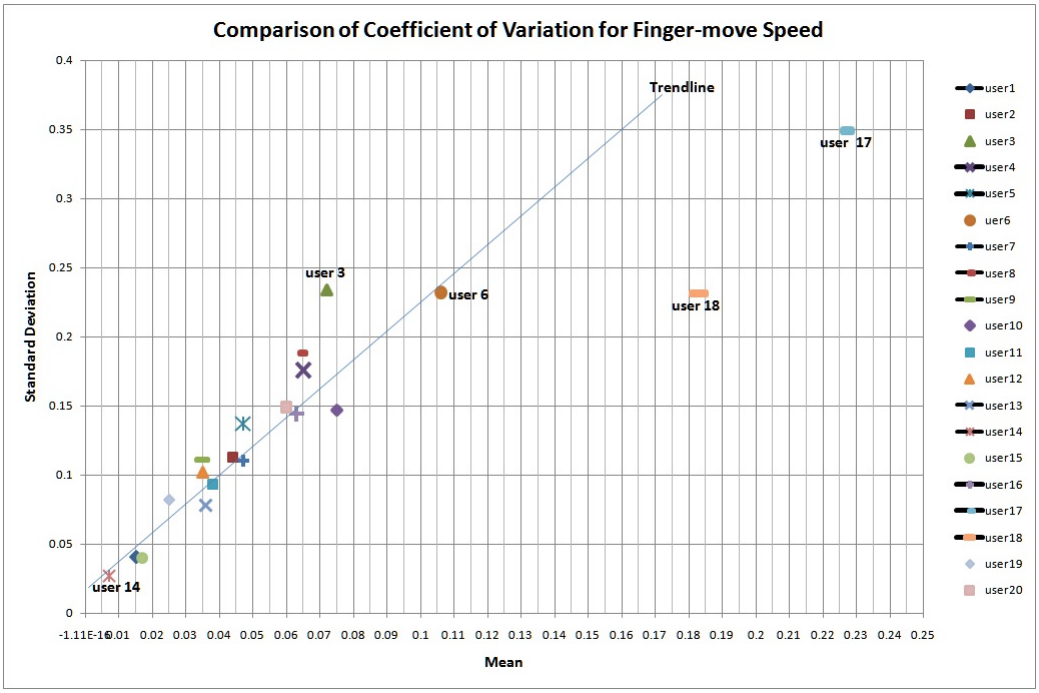


Figure 5.7: Coefficient of Variation Comparison of finger-move speed for 20 users in full test.

Since the average finger pressure force and finger-move speed are chosen to be the final parameters to distinguish individual users, a graph is created to indicate the variance of the users in these two parameters. The comparison of mean of finger pressure force and the finger-move speed from the 20 users are plotted in figure 5.8. As shown in the graph, each user is identified by the finger-sliding mean in x-location and finger force mean in y-coordinate. User 1 ,which is the point on the top left in the figure, has a extremely small average of finger-move speed on the screen while holds a largest finger pressure force in the 20 users. User 17 and user 18 appear to slide a lot during the reading process with the touch device. With this figure, every user can be discriminated because of the huge difference. This gives a reason to hope the implementation of finger movements based on biometric authentication for touch devices.

5.2 Sample Profile

Each sample in full test consist a number (DownCount) of press-down points and a set (MoveCount) of press-move points in a unit time (1 minute). A parameter PointsDistribution (pd) is defined by DownCount and MoveCount as described in formula 5.1. The coefficient of variation (cv_x) for x-location of finger points is calculated with the mean (mean_xlocation) and the standard deviation (st_xlocation) of this parameter by the formula 5.2. The skewness (sk_x) of x-location of touch points is chosen as one parameter to judge the

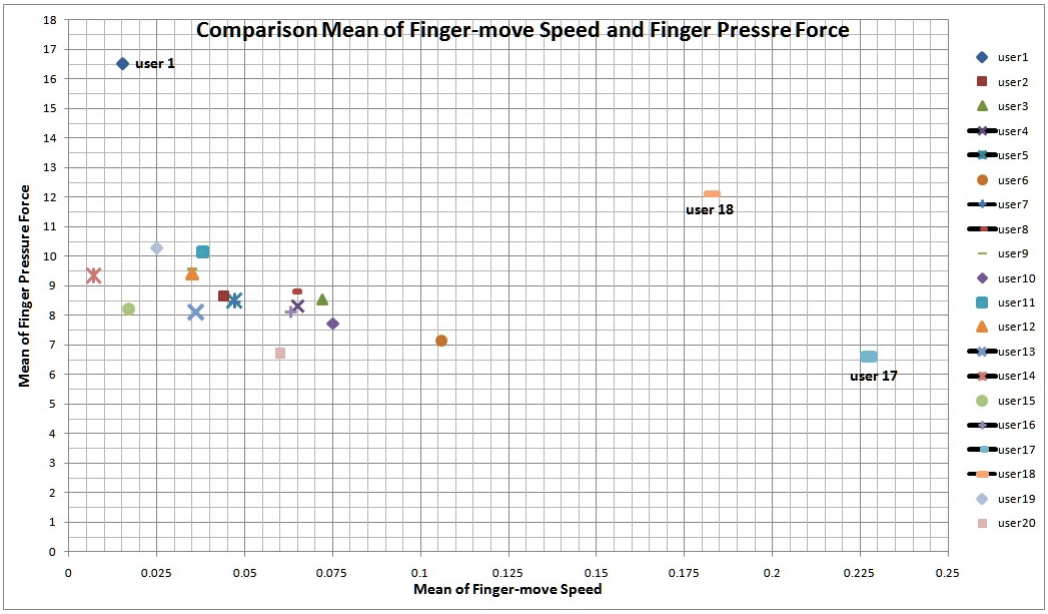


Figure 5.8: Comparison of mean value of finger force and finger-move speed for 20 users in full test.

spread variance in x-coordinates for different users. The mean of finger force (mean_force) and the finger-move speed mean (mean_speed) are selected into the final parameters to indicate the characteristics of finger pressure force and sliding-move speed when the users touch the screen. These 5 touch parameters, which are the final parameters to compose the user profiles, reveal the properties of different touch style of individual users.

$$pd = \frac{MoveCount}{DownCount} \tag{5.1}$$

$$cv\_x = \frac{st\_xloaction}{mean\_xlocation} \tag{5.2}$$

Based on the previous selected parameters, the profiles can be created to show the touch characters of the individual users and could be used to identify the individuals. Table 5.1 is the touch information profile of the user samples in chapter 4, including user 6, user 14, user 17 and user 18.

user	pd	cv_x	sk_x	mean_force	mean_speed
user 6	23.559	0.279	-0.877	7.139	0.106
user 14	60.363	0.255	-0.419	9.360	0.007
user 17	8.634	0.146	-0.712	6.610	0.226
user 18	7.494	0.739	0.028	12.120	0.183

Table 5.1: Touch character profile of the 4 selected subjects.

Looking at this user profile, user 6 and user 14 have a much larger pb value than user 17 and user 18, which means the previous two users have more fin-

ger sliding movements on the screen while the user 17 and user 18 prefer to press down the screen with single touch points. User 18 has an extremely large cv_x value and the only positive sk_x value among these 4 subjects. This is caused by the large standard deviation and big spread area of touch points on screen, showing the bimanual usage of user 18. User 17 holds the smallest mean of finger pressure force and the largest average finger force is resided to user 18. User 17 slides fast on the screen with the largest mean_speed but this parameter is extremely small for user 14.

Each profile shows the touch characteristics from one user with 5 touch parameters which are chosen because of the good user recognition. These profiles can identify a user and distinguish to other individuals based on the different values of the touch parameters. This approach gives good results of user identification and probably can be applied to the users, who use the touch devices, in the real world to build their personal touch profiles for biometric authentication.

5.3 Samples

Experimental testing and analysis always need big enough sample size and a proper amount of data. This project would benefit from having more users in the full test and comparing more samples in the data analysis. However, getting a large amount of data is difficult especially with the time constraints. This project collected the touch information from individual users through an android application in the Samsung Galaxy Tab. Every reading process for testing was around 15 minutes and the full test was totally dependent on the only one tablet. Long time horizon of test and restrictive usage of testing equipment make it is formidable to acquire a larger amount of data in a limited time.

The results of the full test are separately based on the data from one user with one test. Users with high variance or anomalous usage of the touch device would greatly affect the final comparison results. A larger amount of testing users and sample data could possibly reduce the impact of abnormal samples and improve the final results.

5.4 Touch Parameters

The project collects and construes several touch parameters including account of different type of finger points, x-location of touch points, finger pressure force and finger-move speed from individual users. These parameters show the distinct characters of personal touch styles. The analysis would benefit from getting more touch parameters. The task of analyzing more touch features is arduous in the limited time. And more parameters possibly make the comparisons too confused to find the emphases. Since the research of touch

screen recognition is a very new area. Studying several possible key parameters to get the main difference of touch characteristics from users first, is a reasonable start of primary stage. For further research, more touch parameters should be inspected closer as further work.

The analysis of the touch parameters chooses mean, standard deviation and coefficient of variation as value features for each dataset. The results are satisfactory or even surprising with these simple calculations. More analysis modules or mathematical functions could be applied in filtering touch characters for personal profile, to obtain more accurate parameters, which would result in more trustworthy results.

5.5 Problems Encountered

In the beginning of thinking this project in mind, how to detect the finger activities and get the touch information of the finger points from users are great difficulties. The Samsung Galaxy Tab is the available touch device for experiment which is supporting android system. Since the android has an object of touch event which can be detected by the function of touch screen and contains a number of touch information from each finger point, the solution is to develop an android application to recognize the finger movements and collect the touch parameters.

When developing the android application, one problem is how to create the log file for different users and write the personal touch information into the right file. One solution is to design a graphic window to obtain the username and create the log file based on this name before the reading process, and use the file-output stream to open and stored the data into log files. So in the full test, the first step is to type the name of each user before starting the reading process.

Choosing the testing samples and reading material for full test are conundrums in the project. To get trustworthy results, a wide range of 20 samples are chosen, people in different gender, age, origin and the proficiency of using touch devices. Because of the wide sample range, it is difficult to choose a document which can make all users find interesting during a long reading time horizon. The final solution for the reading material is a comic paper, which is proved to be an interesting reading for most of users.

During the data analysis, how to filter the value features from the datasets to show the characters of each touch parameter is a great challenge. With a huge amount of data, complicated modules or functions are difficult to implement. To solve this problem, mean, standard deviation, coefficient of variation and histogram are applied to the dataset to filter the value features and show the touch traits of each user.

The parameter results of the location of touch points are surprising, but the data is a little messy and not easy to compare. To make a better comparison, the x-locations of the touch points are filtered out and the y-coordinates values are ignored because of the small variance for each user in this axis.

5.6 Reliability and Scalability

In this thesis, the aim is to compare different touch parameters from individual users to see the personal touch characters and the variance between each other. Then the suggested user profile based on several selected parameters is created to indicate the personal touch style of using the touch device. These profiles could be used to identify a user by the finger movements on the touch screen. The reliability of the biometric recognition is not satisfactory without achieving a high accuracy of identification. The research of biometric authentication for touch screen is just in the infancy. With the widespread usage of touch-screen, more focus have been assembled from scientific area. The finger movements based on biometric authentication could be used as a auxiliary method of identification for touch devices.

A login authentication process could take place on touch devices for security check. Based on the huge difference of touch-parameter results in this thesis, the touch biometric authentication could be actualized. With the implementation of finger movements of recognition, the system could repeat the user identification process automatically by itself without reminding the users. Storing personal touch profiles requires very little system resource. More physical space will be needed if a touch device is communal for many users. There are very few choices for collecting touch information from users through touch devices. This project implemented the detection of finger activities and collection of touch information by developing an android application in a tablet. How to build up the mechanism of collecting data of finger movements on screen on other platforms is a big conundrum of scalability.

5.7 Biometric Variation

A profile of a user used for authentication is defined by physical characters or usual behaviors. These biometric credentials could possibly be changed by some unexpected events such as breading fingers or suffering serious illness. The physical characteristics and personal behaviors would change a lot comparing to the original profile. And the emotional changes could also affect personal style of using the touch devices. A regular updating profile process could be one solution to reduce the impact of these factors.

Chapter 6

Conclusion

As the fact that the usage of touch screen is rapidly growing because of user friendly and quick access mechanism, increasingly important data could be stored into the touch devices. In order to strengthen the data protection and confidentiality of a touch device, an authentication system is required. Although there are various authenticate methods, biometric identification is one of the among the most promising areas of research. Therefore, this project focuses on the study of finding and testing the touch parameters of individual users and build the personal profiles to show the characteristics of the users using the touch device.

With the aim of comparing the characters of individual users touching the screen, this thesis has collected and compared several touch parameters including account of different type of touch points, x-location of finger points, finger pressure force and speed of finger movements from different subjects. By applying the simple data analysis such as mean, standard deviation and skewness, the comparison results are satisfactory with showing huge difference of touch parameters for the testing users between each other. With the high identifiable parameters, user profiles of using a touch device are created for further authentication. Hopefully, the results of this thesis will be valuable for the further research of finger movements based on biometric authentication for touch devices.

6.1 Future Work

This thesis collected and compared different touch parameters of finger movements on touch screen from individual users. Several parameters with good user recognition were chosen to build sample profiles which could indicate the touch characteristics of each user. More work should move forward to the value test to confirm the accuracy of these parameters. It would be important to do another full test for the same users with different reading materials to obtain a new dataset of touch information. The comparisons between the new data and profiles could give the accuracy of identification for the touch parameters.

This project analyzed each personal data of the whole reading time. It would be interesting to divide the time horizon, each subject used to finish the reading process, to some time units such as 5 minutes, 2 minutes or 1 minute and analyze the touch parameters during each time unit to see if they still have the same characteristics.

Future work can also consider different types of touch devices. This thesis used the Samsung Galaxy Tab as testing equipment, more test could be performed with diverse touch devices to achieve reliable and generalizable results.

Bibliography

[1] A Lin Hong; Yifei Wan; Jain. Fingerprint image enhancement: algorithm and performance evaluation. *Pattern Analysis and Machine Intelligence*, 20(8):777–789, 1998.

[2] Tom Olzak. Keystroke dynamics: Low impact biometric verification. 2006.

[3] Robin Snyder. Using ethical hacking to educate users about secure passwords by cracking insecure passwords using readily available software. In *Proceedings of the 39th Annual Confer-ence of the Association of Small Computer Users in Education. Myrtle Beach, South Carolina*, 2006.

[4] Arash Habibi Lashkari; Samaneh Farmand; Dr. Omar Bin Zakaria; Dr. Rosli Saleh. Shoulder surfing attack in graphical password authentication. *International Journal of Computer Science and Information Security*, 6(2):145–154, 2009.

[5] Dr.Wayne C.Summers and Dr.Edward Bosworth. Password policy: The good, the bad, and the ugly. In *Proceedings of the winter international synposium on Information and communication technologies*, pages 1–6, 2004.

[6] Takahiro Watanabe; Yasunobu Nohara; Kensuke Baba Sozo Inoue; Hiroto Yasuura. On authentication between human and computer. In *Fourth IEEE International Conference on Pervasive Computing and Communications Workshops*, 2006.

[7] Stephen Brewster. *Digital Applications for Cultural and Heritage Institutions*. Ashgate Publishing Company.

[8] Matt Bishop. What is computer security? In *Security Privacy, IEEE*, 2003.

[9] Dieter Gollman. *Computer Security*. John Wiley Sons, 2004.

[10] Steve Suehring and Robert Ziegler. *Linux Firewalls, Third Edition*. Sams Publishing, 2005.

[11] Intrusion definition
http://www.thefreedictionary.com/intrusion.

[12] Karen Scarfone and Peter Mell. Guid to intrusion detection and prevention systems. Technical report, NIST, US Dep. of Commerce, 2007. 800-94.

[13] Lin Ying ; Zhang Yan ; Ou Yang-jia. The design and implementation of host-based intrusion detection system. Technical report, Intelligent Information Technology and Security Informatics (IITSI), 2010 Third International Symposium on, 2010.

[14] Shimonsk R.J. Firewall Book Period.

[15] Iptable
http://www.netfilter.org/projects/iptables/.

[16] D. Brent Chapman and Elizabeth D. Zwicky. *Building Internet Firewalls*. O'Reilly Associates, Inc. Sebastopol, CA, USA, 1995.

[17] Seymour Bosworth and M.E.Kabay. *Computer Security Handbook, Fourth Edition*. John Wiley Sons, Inc. New York, NY, USA, 2002.

[18] Maja Pusara and Carla E. Brodley. User reauthentication via mouse movements. In *VizSEC/DMSEC '04 Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security*, 2004.

[19] Benny Pinkas and Tomas Sander. Securing passwords against dictionary attacks. In *Proceeding CCS '02 Proceedings of the 9th ACM conference on Computer and communications security*, 2002.

[20] Donn Seeley. Password cracking: a game of wits. *Magazine Communications of the ACM CACM*, 32(6), 1989.

[21] Fernando L. Podio and Jeffrey S. Dunn. Biometric authentication technology: From the movies to your desktop. Technical report, National Institute of Standard and Technology (NIST), 2004.

[22] Alice Osborn. Biometrics history
http://www.video-surveillance-guide.com/biometrics-history.htm.

[23] Mamta Kothavale; Robert Markworth; Parmajit Sandhu. Computer security ss3: Biometric authentication. Technical report, University of Birmingham, 2004.

[24] Arun Ross and Ani Jain. Multimodal biometrics: An overview. In *Proceedings of the 12thEuropean Signal Processing Conference*.

[25] K. ; Amirfattahi R azdanpanah, A.P. ; Faez. Multimodal biometric system using face, ear and gait biometrics. In *Information Sciences Signal Processing and their Applications (ISSPA)*, 2010.

[26] H.C.Lee and R.E.Gaensslen. Advances in fingerprint technology. *CRC Press, New York:Elsevier*, 1991.

[27] Ashbaugh and David R. Ridgeology. *Journal of Forensic Identification*, 41(1):16–64, 1991.

[28] Mamta Kothavale; Robert Markworth; Parmajit Sandhu. Biometric Authentication
http://www.cs.bham.ac.uk/ mdr/teaching/modules03/security/students/SS3/handout/.

[29] Bhanu Bir and Chen Hui. *Human Ear Recognition by Computer, 1st Edition*. Springer, 2008.

[30] Marios Savvides; B. V. K. Vijaya Kumar; P. K. Khosla. Cancelable biometric filters for face recognition. In *17th International Conference on Pattern Recognition (ICPR'04) - Volume 3*, pages 922–925, 2004.

[31] John H Payne. Biometric face recognition for application screening. 2000.

[32] A. Iannarelli. *Ear Identification*. Paramont Publishing, 1989.

[33] Mark Nixon. Ear recognition may beat face biometrics. *NewScientist*, 23.

[34] Jeffrey E. Boyd and James J. Little. Biometric gait recognition. *ComputerScience*, 3161:19–42, 2005.

[35] H Crawford. Keystroke dynamics: Characteristics and opportunities. In *Privacy Security and Trust (PST), 2010 Eighth Annual International Conference*, pages 205–212, 2010.

[36] Keystroke dynamics history.

[37] Gaines R.; Lisowski W.; Press S.; Shapiro N. Authentication by keystroke timing: Some preliminary results. Technical report, Technical Report Rand report R-256-NSF, 1980.

[38] Joyce R.and Gupta G. Identity authentication based on keystroke latencies. *Communications of the ACM*, 33:168–176.

[39] Gokcay (Unlu) D. User identification through neural network algorithms, 1991.

[40] Monrose F.; Reiter M. K.; Wetzel S. Password hardening based on keystroke dynamics. In *Proceedings of sixth ACM Conference on Computer and Communications Security, CCCS*, pages 73–82, 1999.

[41] Alex Andersen. Biometric authentication and identification using keystroke dynamics with alert levels, 2007.

[42] Ahmed Awad E. Ahmed and Issa Traore. A new biometric technology based on mouse dynamics. *IEEE Transactions on Dependable and Secure Computing*, 4:165–179, 2007.

[43] A.A.E. Ahmed and I. Traore. Anomaly intrusion detection based on biometrics. In *Information Assurance Workshop. IAW '05. Proceedings from the Sixth Annual IEEE SMC*, pages 452–453, 2005.

[44] Maja Pusara and Carla E. Brodley. User re-authentication via mouse movements. In *Proceedings of the ACM workshop on Visualization and data mining for computer security*, pages 1–8, 2004.

[45] Adam Weiss; Anil Ramapanicker; Pranav Shah; Shinese Noble; Larry Immohr. Mouse movements biometric identification: A feasibility study. In *Proceedings of Student/Faculty Research Day, CSIS*, 2007.

[46] Douglas A.Schulz. Mouse curve biometrics. In *Biometric Consortium Conference*, pages 1–6, 2007.

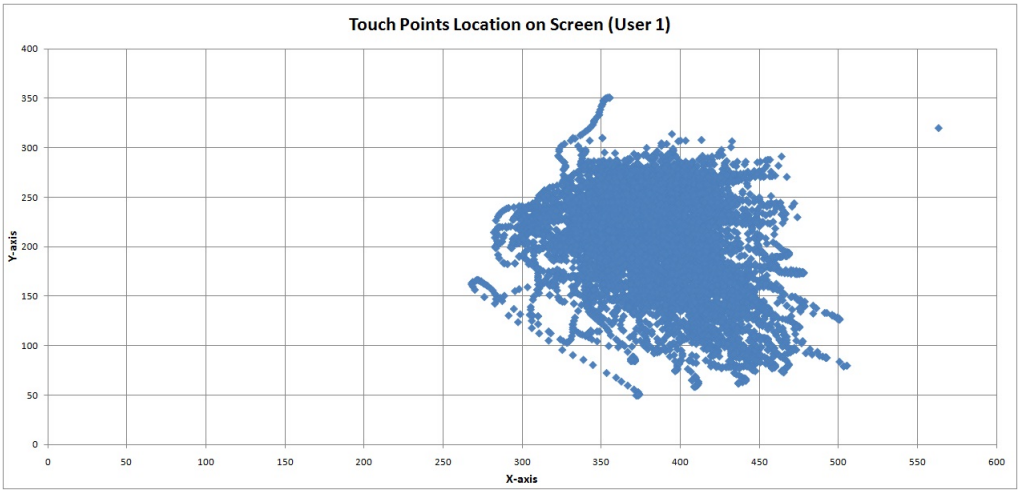
[47] Touch Screen
http://inventors.about.com/library/inventors/bltouch.htm.

[48] Rick Rogers. *Android Application Development*. O'Reilly, 2009.

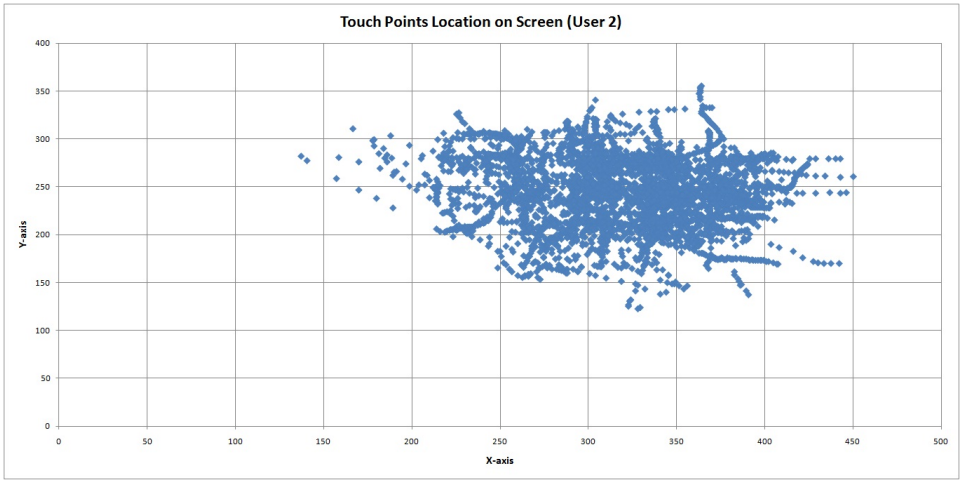
Appendix A

Touch Points Location

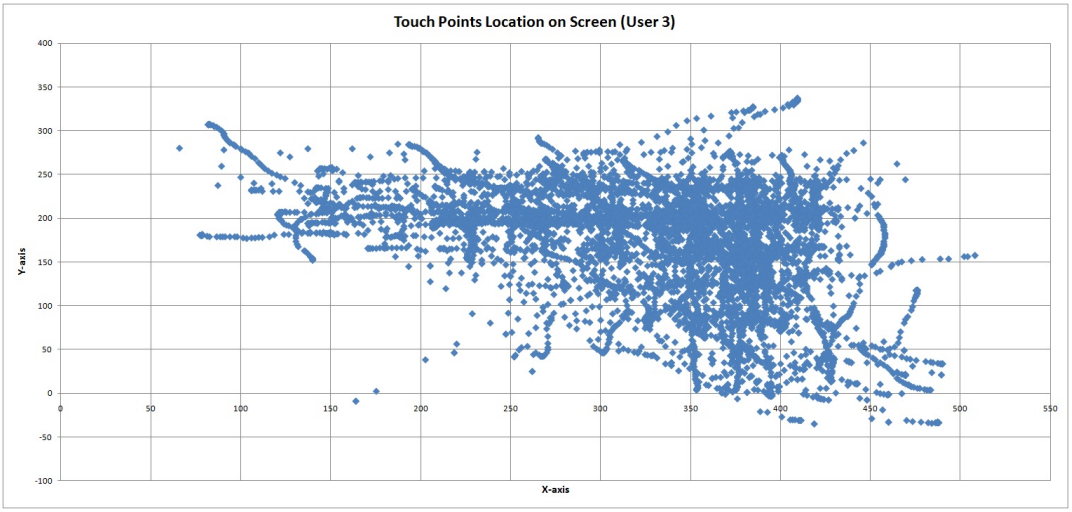
A.1 Touch Points Location on Screen from User1



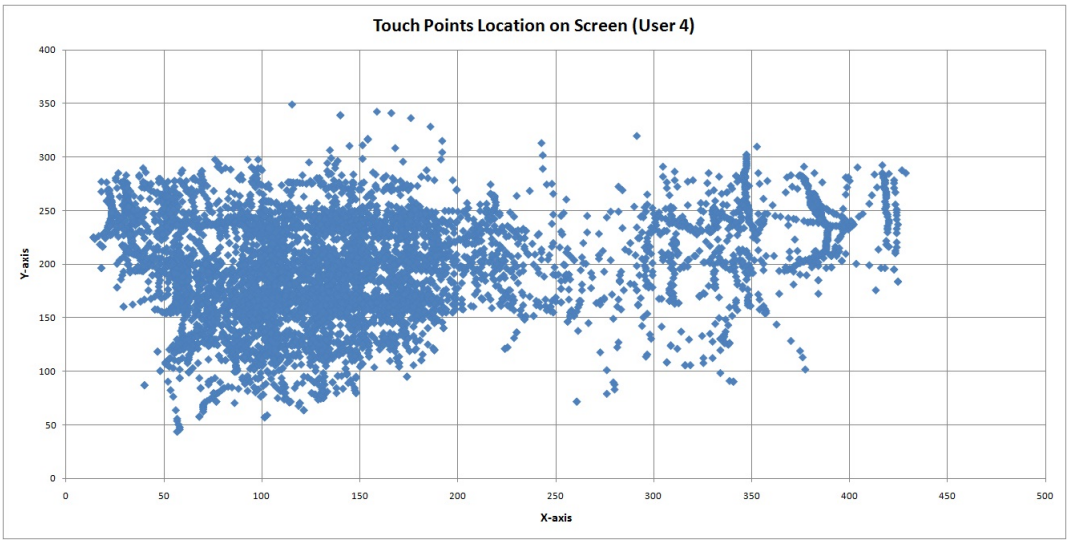
A.2 Touch Points Location on Screen from User2



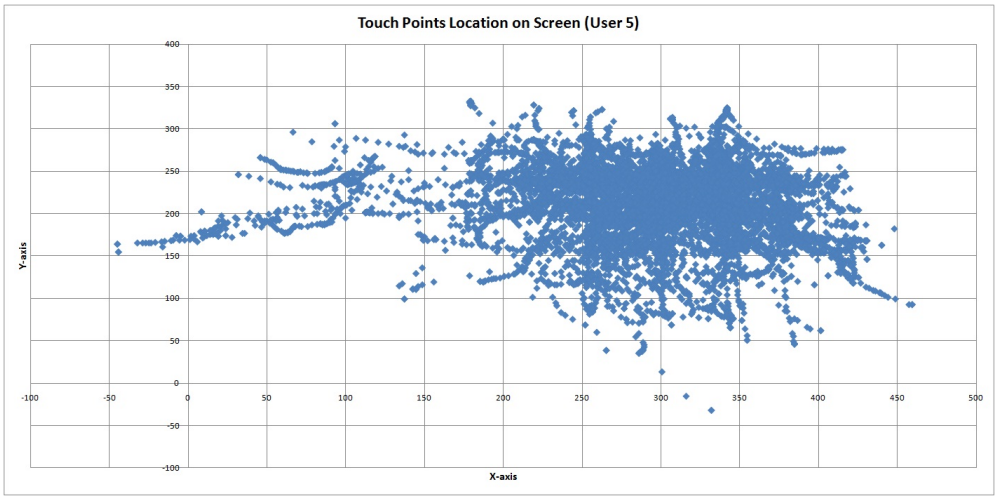
A.3 Touch Points Location on Screen from User3



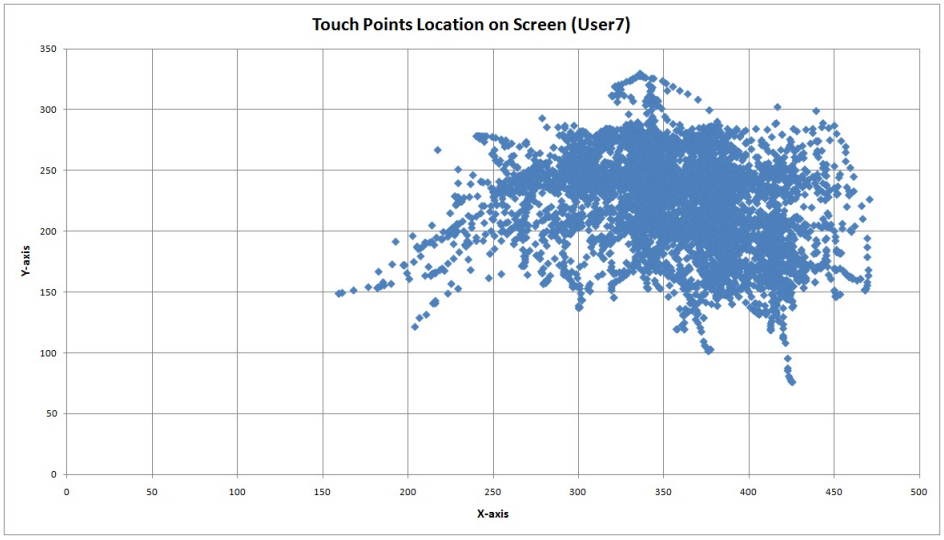
A.4 Touch Points Location on Screen from User4



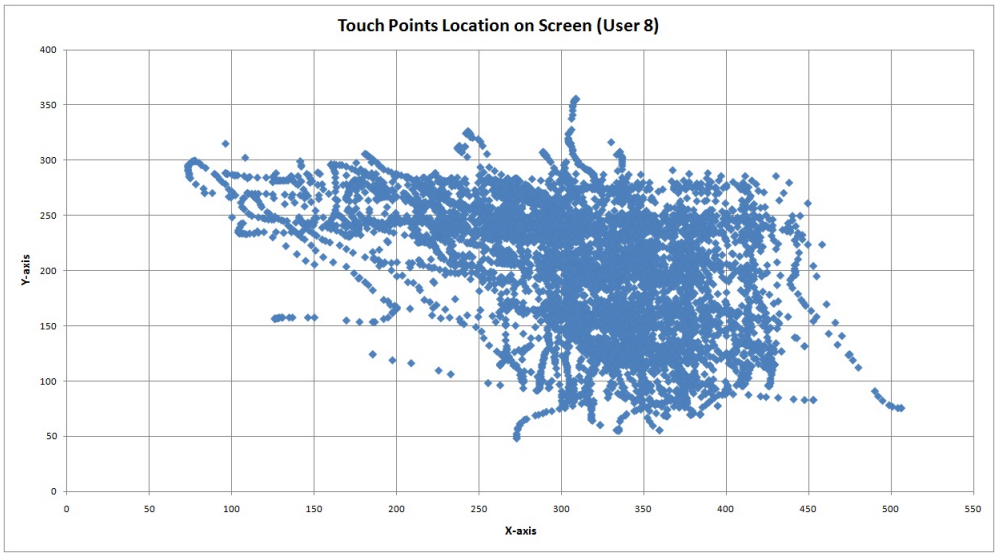
A.5 Touch Points Location on Screen from User5



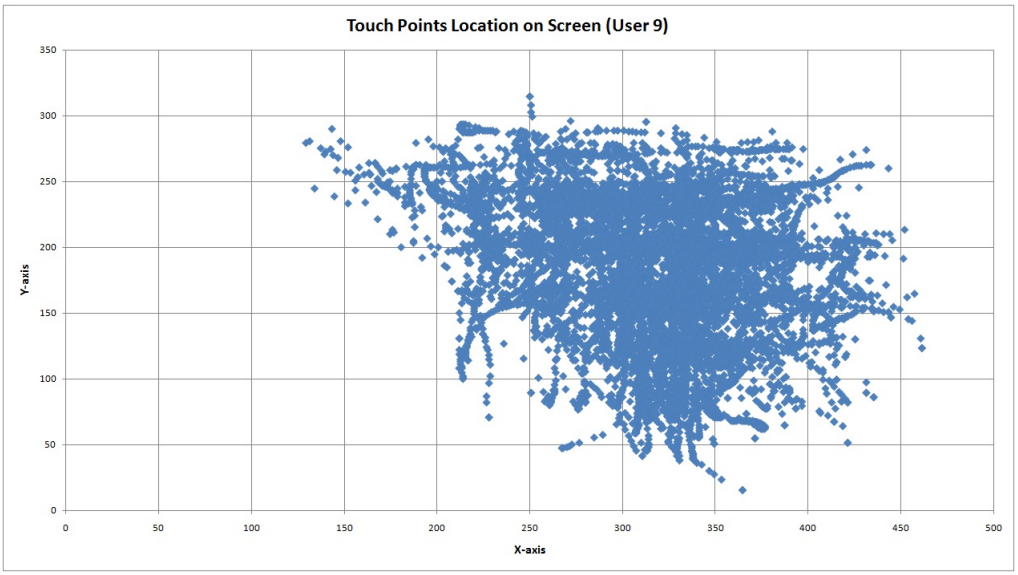
A.6 Touch Points Location on Screen from User7



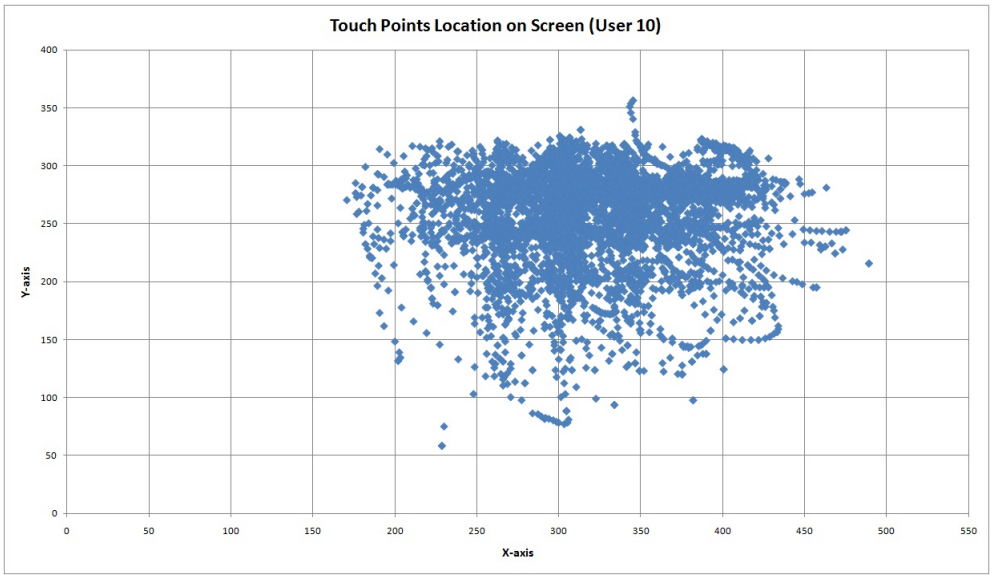
A.7 Touch Points Location on Screen from User8



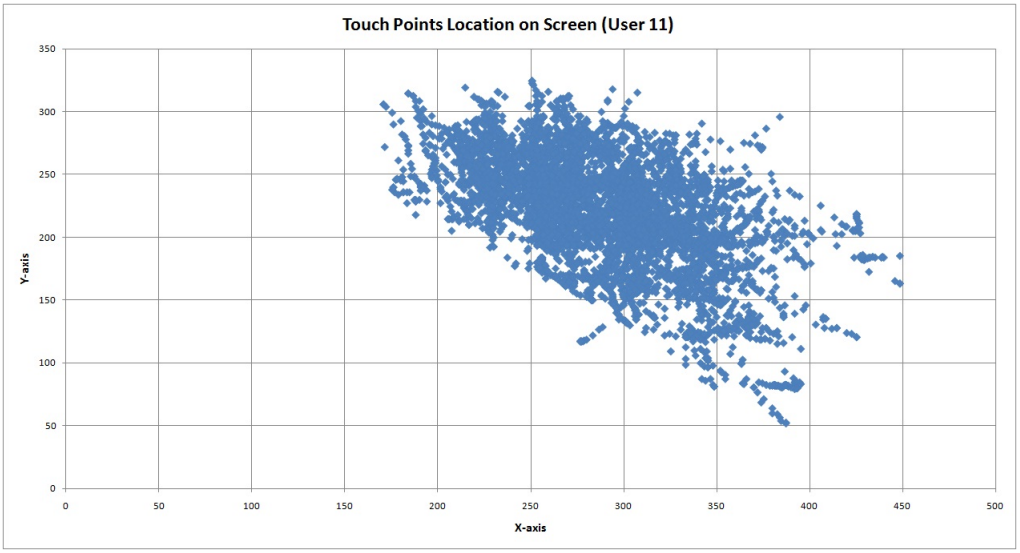
A.8 Touch Points Location on Screen from User9



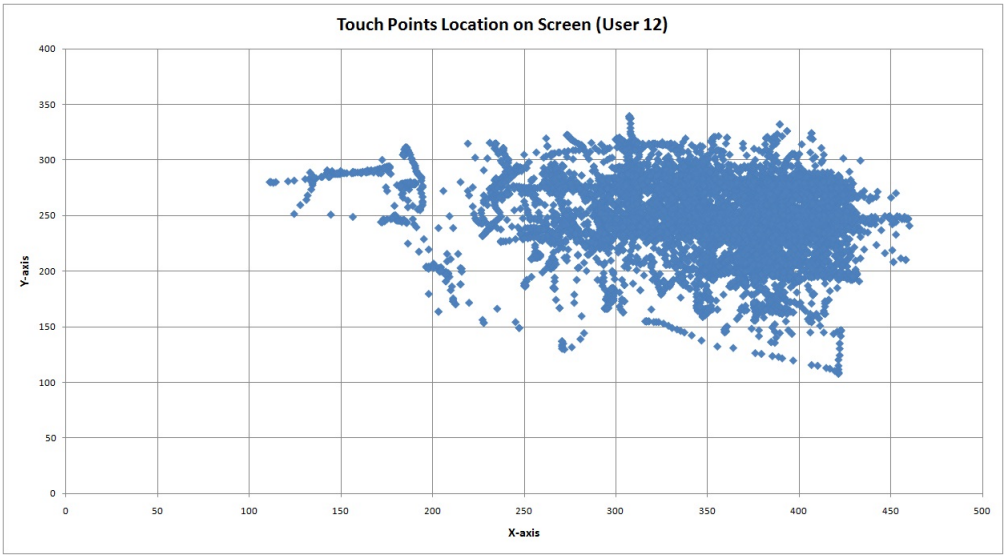
A.9 Touch Points Location on Screen from User10



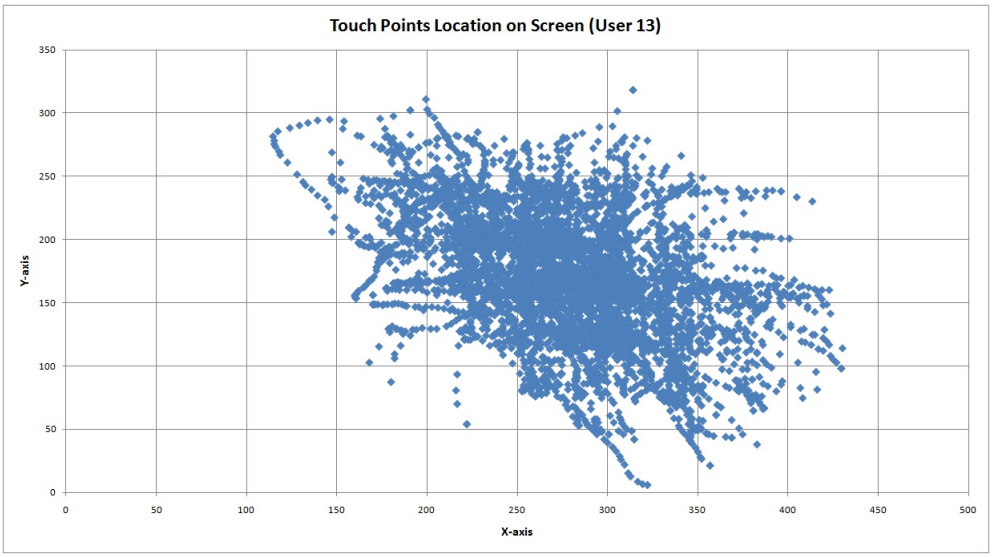
A.10 Touch Points Location on Screen from User11



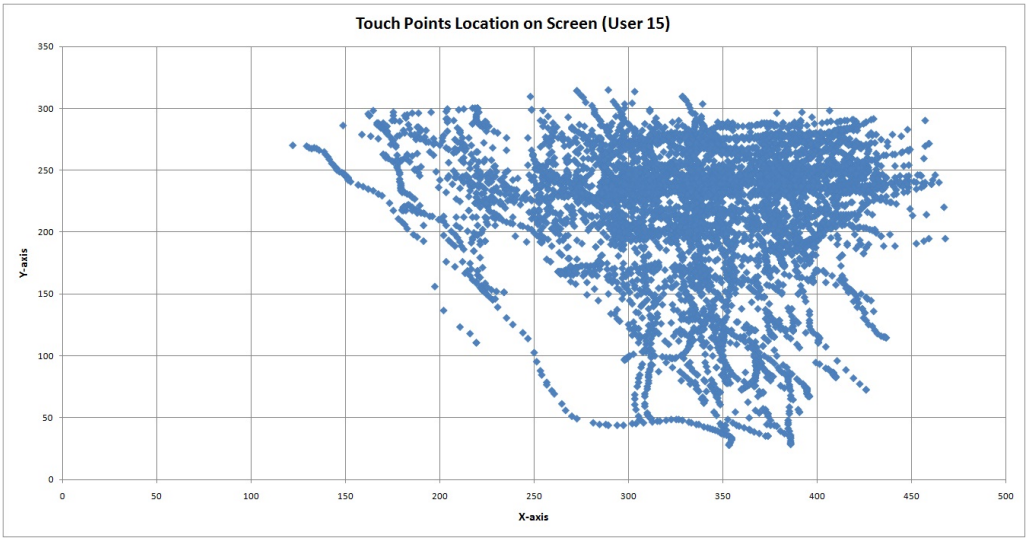
A.11 Touch Points Location on Screen from User12



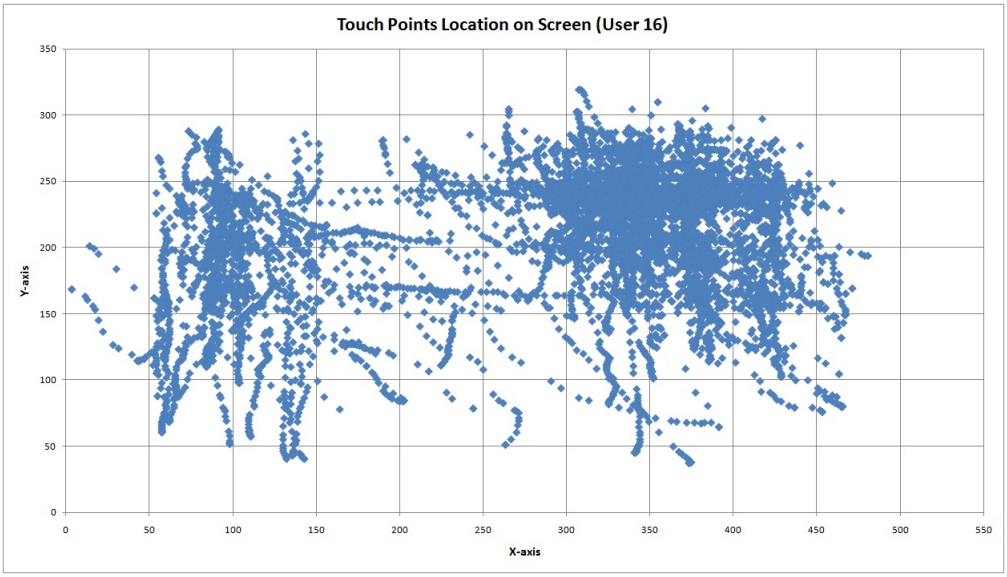
A.12 Touch Points Location on Screen from User13



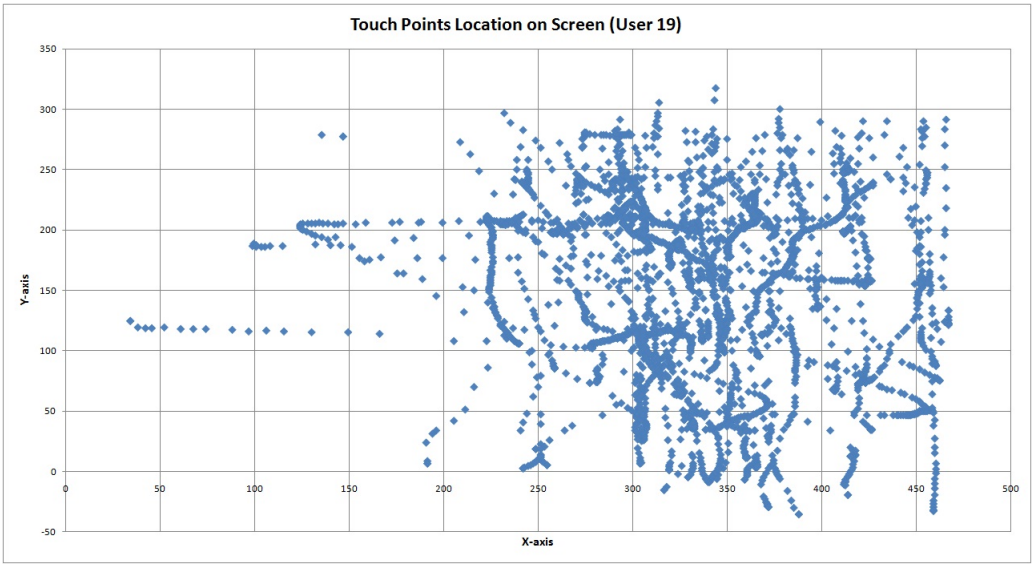
A.13 Touch Points Location on Screen from User15



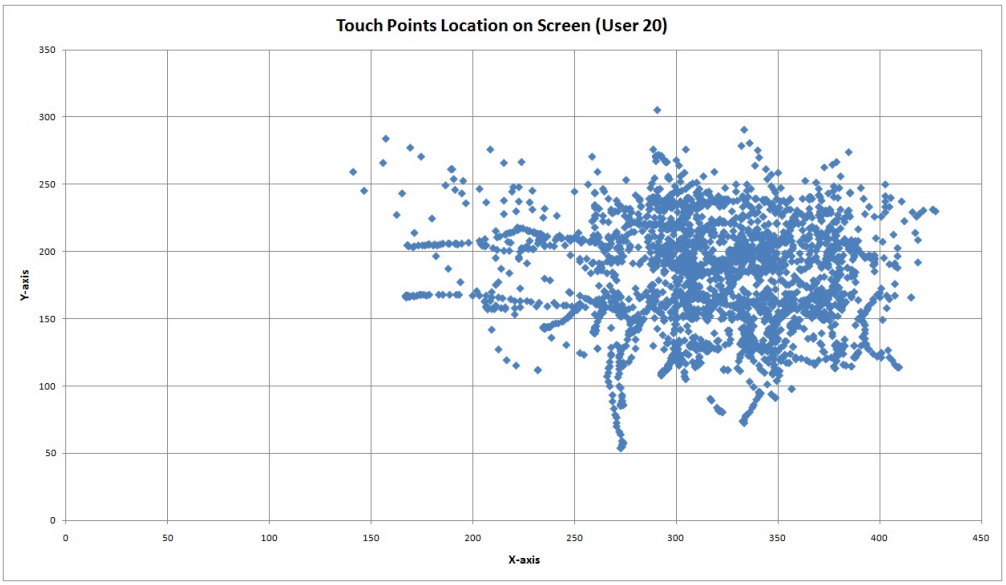
A.14 Touch Points Location on Screen from User16



A.15 Touch Points Location on Screen from User19



A.16 Touch Points Location on Screen from User20



Appendix B

The Android Application

B.1 JAVA Script for Creating Log Files Based on Typing Usernames

```
1  package net.sf.andpdf.pdfviewer;
2
3  import java.io.File;
4  import java.util.ArrayList;
5  import java.util.List;
6
7  import android.app.ListActivity;
8  import android.content.Context;
9  import android.content.Intent;
10 import android.os.Bundle;
11 import android.os.Environment;
12 import android.util.Log;
13 import android.view.Menu;
14 import android.view.MenuItem;
15 import android.view.View;
16 import android.view.View.OnClickListener;
17 import android.widget.Button;
18 import android.widget.CheckBox;
19 import android.widget.EditText;
20 import android.widget.ListView;
21 import android.widget.TextView;
22
23 import java.io.File;
24 import java.io.FileNotFoundException;
25 import java.io.FileOutputStream;
26 import java.io.FileInputStream;
27 import java.io.IOException;
28 import java.io.OutputStreamWriter;
29 import java.io.UnsupportedEncodingException;
30
```

```
31 public class PdfFileSelectActivity extends ListActivity{
32
33 private static final String TAG = "PDFVIEWER";
34 public final static String PREFS_NAME = \\
35 "PDFViewerPrefs";
36 public static final String PREFS_PDFFILENAME = \\
37 "resultfile";
38 public static final String PREFS_ANTIALIAS = \\
39 "antialias";
40 public final static String DEFAULTPDFFILENAME = \\
41 "xiaoli01";
42 public static final boolean DEFAULTSHOWIMAGES = true;
43 public static final boolean DEFAULTANTIALIAS = true;
44
45 public static final String EXTRA_PDFFILENAME = \\
46 "net.sf.andpdf.extra.PDFFILENAME";
47 public static final String EXTRA_SHOWIMAGES = \\
48 "net.sf.andpdf.extra.SHOWIMAGES";
49 public static final String EXTRA_ANTIALIAS = \\
50 "net.sf.andpdf.extra.ANTIALIAS";
51
52 private EditText mFilename;
53 private EditText mOutput;
54 private CheckBox mAntiAlias;
55 private Button mShow;
56 private Button mExit;
57
58 private SimplePersistence persist;
59 public static final int BROSWER_ID = Menu.FIRST;
60 private TextView mPath;
61 private String rootPath = "/";
62
63 private List<String> items = null;
64 private List<String> paths = null;
65
66 final static String FOLDER = "/android_xiaoliData/";
67 public static String targetPath = "";
68
69 /** Called when the activity is first created. */
70 @Override
71 public void onCreate(Bundle savedInstanceState) {
72 super.onCreate(savedInstanceState);
73 setContentView(R.layout.pdf_file_select);
74
75 mFilename = (EditText) findViewById(R.id.filename);
76 mOutput = (EditText) findViewById(R.id.output);
77 mAntiAlias = (CheckBox) findViewById(R.id.cbAntiAlias);
```

```
78  mShow = (Button) findViewById(R.id.btShow);
79  mExit = (Button) findViewById(R.id.btExit);
80
81  mShow.setOnClickListener(ShowPdfListener);
82  mExit.setOnClickListener(ExitListener);
83
84  // load persisted values
85  persist = new SimplePersistence(this, PREFS_NAME);
86  String pdffilename = persist.getString \\
87  (PREFS_PDFFILENAME, DEFAULTPDFFILENAME);
88  boolean antiAlias = persist.getBoolean \\
89  (PREFS_ANTIALIAS, DEFAULTANTIALIAS);
90  mFilename.setText(pdffilename);
91  mAntiAlias.setChecked(antiAlias);
92  }
93  protected void setFileSelectView(int layoutResId){
94  mFilename = (EditText) findViewById(R.id.filename);
95  mOutput = (EditText) findViewById(R.id.output);
96  mAntiAlias = (CheckBox) findViewById(R.id.cbAntiAlias);
97  mShow = (Button) findViewById(R.id.btShow);
98  mExit = (Button) findViewById(R.id.btExit);
99
100 mShow.setOnClickListener(ShowPdfListener);
101 mExit.setOnClickListener(ExitListener);
102 }
103
104 @Override
105 protected void onStop() {
106 super.onStop();
107 persistValues();
108 }
109
110 private void persistValues() {
111 String resultfile = mFilename.getText().toString();
112 boolean antiAlias = mAntiAlias.isChecked();
113 persist.putString(PREFS_PDFFILENAME, resultfile);
114 persist.putBoolean(PREFS_ANTIALIAS, antiAlias);
115 persist.commit();
116 }
117
118 private void showText(String text) {
119 Log.i(TAG, text);
120 mOutput.setTag(text);
121 }
122
123 OnClickListener ExitListener = new OnClickListener() {
124 public void onClick(View v) {
```

```
finish();
}
};

OnClickListener ShowPdfListener = new OnClickListener(){
public void onClick(View v) {
persistValues();
String resultfile = mFilename.getText().toString();
boolean antiAlias = mAntiAlias.isChecked();
Intent intent = new Intent(PdfFileSelectActivity.this,\\
PdfViewerActivity.class).putExtra(EXTRA_PDFFILENAME,\\
resultfile) .putExtra(EXTRA_ANTIALIAS, antiAlias);
startActivity(intent);
String foldername = Environment. \\
getExternalStorageDirectory().getPath()+ FOLDER;
File folder = new File(foldername);
if (folder != null && !folder.exists()) {
    if (!folder.mkdir() && !folder.isDirectory())
       {
         Log.d(TAG, "Error: make dir failed!");
         return;
        }
      }
targetPath = foldername + resultfile;
File targetFile = new File(targetPath);
if (targetFile != null) {
   if (targetFile.exists()) {
      targetFile.delete();
    }
   OutputStreamWriter osw;
   try{
       osw = new OutputStreamWriter(
       new FileOutputStream(targetFile,true),"utf-8");
       try {
            osw.write("press_type;point_x_screen; \\
              point_y_screen;point_x_container; \\
            point_y_container;pressure;point_size;\\
            finger_force;event_start_time; \\
            event_end_time;event_time; \\
            move_dx;move_dy;speed_x;speed_y\n");
osw.flush();
osw.close();
} catch (IOException e) {
   // TODO Auto-generated catch block
   e.printStackTrace();
  }
} catch (UnsupportedEncodingException e1) {
```

```
172   // TODO Auto-generated catch block
173   e1.printStackTrace();
174   } catch (FileNotFoundException e1) {
175   // TODO Auto-generated catch block
176   e1.printStackTrace();
177   }
178 }
179 }
180 };
181
182 @Override
183 public boolean onCreateOptionsMenu(Menu menu) {
184 super.onCreateOptionsMenu(menu);
185 menu.add(0, BROSWER_ID, 0, "Browser...");
186 return true;
187 }
188
189 @Override
190 public boolean onOptionsItemSelected(MenuItem item) {
191 switch(item.getItemId()){
192 case BROSWER_ID:
193         {
194         setContentView(R.layout.file_explorer);
195         mPath = (TextView) findViewById(R.id.mPath);
196         getFileDir(rootPath);
197         }
198         break;
199         default:
200         break;
201         }
202         return super.onOptionsItemSelected(item);
203 }
204
205 /** Get the file structure */
206 private void getFileDir(String filePath) {
207 mPath.setText(filePath);
208 items = new ArrayList<String>();
209 paths = new ArrayList<String>();
210 File f = new File(filePath);
211 File[] files = f.listFiles();
212 if (!filePath.equals(rootPath)) {
213         items.add("back2root");
214         paths.add(rootPath);
215         items.add("back2up");
216         paths.add(f.getParent());
217         }
218 for (int i = 0; i < files.length; i++) {
```

```
219         File file = files[i];
220         items.add(file.getName());
221         paths.add(file.getPath());
222         }
223 setListAdapter(new MyAdapter(this, items, paths));
224 }
225
226 @Override
227 protected void onListItemClick(ListView l, View v, \\
228 int position, long id) {
229 File file = new File(paths.get(position));
230 String fName = file.getName();
231 if (file.isDirectory())
232 getFileDir(paths.get(position));
233 else if (fName.substring(fName.lastIndexOf(".") + 1, \\
234 fName.length()).toLowerCase().equals("pdf"))
235 updateFileSelected(file);
236 else
237         ;//Do nothing
238 }
239
240 private void updateFileSelected(File file) {
241         setContentView(R.layout.pdf_file_select);
242         mFilename = (EditText) findViewById \\
243       (R.id.filename);
244         mFilename.setText(file.getAbsolutePath());
245         mAntiAlias = (CheckBox) findViewById \\
246      (R.id.cbAntiAlias);
247             mShow = (Button) findViewById(R.id.btShow);
248             mExit = (Button) findViewById(R.id.btExit);
249
250       mShow.setOnClickListener(ShowPdfListener);
251       mExit.setOnClickListener(ExitListener);
252
253          // load persisted values
254             persist = new SimplePersistence(this, \\
255       PREFS_NAME);
256             boolean antiAlias = persist.getBoolean \\
257      (PREFS_ANTIALIAS,DEFAULTANTIALIAS);
258       mAntiAlias.setChecked(antiAlias);
259         }
260 }
```

B.2 JAVA Script for Collecting Touch Parameters from Users During Reading Process

```
1  package net.sf.andpdf.pdfviewer;
2
3  import java.io.File;
4  import java.io.FileInputStream;
5  import java.io.FileNotFoundException;
6  import java.io.FileOutputStream;
7  import java.io.IOException;
8  import java.io.InputStream;
9  import java.io.OutputStream;
10 import java.io.OutputStreamWriter;
11 import java.io.RandomAccessFile;
12 import java.io.UnsupportedEncodingException;
13 import java.nio.channels.FileChannel;
14
15 import android.app.Activity;
16 import android.content.Context;
17 import android.content.Intent;
18 import android.graphics.Bitmap;
19 import android.graphics.Canvas;
20 import android.graphics.Color;
21 import android.graphics.Paint;
22 import android.graphics.RectF;
23 import android.graphics.Bitmap.Config;
24 import android.net.Uri;
25 import android.os.Bundle;
26 import android.os.Debug;
27 import android.os.Environment;
28 import android.util.Log;
29 import android.view.Menu;
30 import android.view.MenuItem;
31 import android.view.MotionEvent;
32 import android.view.View;
33
34 import com.sun.pdfview.PDFFile;
35 import com.sun.pdfview.PDFImage;
36 import com.sun.pdfview.PDFPage;
37 import com.sun.pdfview.PDFPaint;
38
39 public class PdfViewerActivity extends Activity {
40
41         private static final int STARTPAGE = 1;
42         private static final float STARTZOOM = 1.0f;
43         private static final String TAG = "PDFVIEWER";
```

```
44
45          private final static int MEN_NEXT_PAGE = 1;
46          private final static int MEN_PREV_PAGE = 2;
47          private final static int MEN_ZOOM_IN   = 3;
48          private final static int MEN_ZOOM_OUT  = 4;
49          private final static int MEN_BACK      = 5;
50
51          private GraphView mGraphView;
52          private String pdffilename;
53          private String resultfile;
54          private PDFFile mPdfFile;
55          private int mPage;
56          private float mZoom;
57        private File mTmpFile;
58        private PDFPage mPdfPage;
59        private Thread backgroundThread;
60
61   /** Called when the activity is first created. */
62   @Override
63  public void onCreate(Bundle savedInstanceState) {
64     super.onCreate(savedInstanceState);
65     if (mGraphView == null) {
66           mGraphView = new GraphView(this);
67            Intent intent = getIntent();
68            Log.i(TAG, ""+intent);
69            byte[] pdfBinary = null;
70            if (intent != null) {
71            if ("android.intent.action.VIEW".equals \\
72          (intent.getAction())) {
73            resultfile = intent.getDataString();
74            pdffilename = Environment.\\
75          getExternalStorageDirectory().getPath() \\
76          +"/android_xiaoliData/testFile.pdf";
77            pdfBinary = readUriContent(intent.getData());
78           }
79           else {
80           resultfile = getIntent(). \\
81           getStringExtra(PdfFileSelectActivity. \\
82         EXTRA_PDFFILENAME);
83           pdffilename = Environment.\\
84         getExternalStorageDirectory(). \\
85         getPath()+"/android_xiaoliData/testFile.pdf";
86             }
87          }
88
89          if (pdffilename == null)
90             pdffilename = "no file selected";
```

```
91            boolean showImages = getIntent().\\
92             getBooleanExtra \\
93          (PdfFileSelectActivity.EXTRA_SHOWIMAGES, \\
94            PdfFileSelectActivity.DEFAULTSHOWIMAGES);
95             PDFImage.sShowImages = showImages;
96             boolean antiAlias = getIntent().\\
97            getBooleanExtra \\
98             (PdfFileSelectActivity.EXTRA_ANTIALIAS, \\
99             PdfFileSelectActivity.DEFAULTANTIALIAS);
100            PDFPaint.s_doAntiAlias = antiAlias;
101
102            parsePDF(pdffilename, pdfBinary);
103
104            setContentView(mGraphView);
105
106            mPage = STARTPAGE;
107            mZoom = STARTZOOM;
108            startRenderThread(mPage, mZoom);
109         }
110 }
111
112 private void showRenderStatus(Canvas c, int x,\\
113  int y, Paint p) {
114 int maxCmds = PDFPage.getParsedCommands();
115 int curCmd = PDFPage.getLastRenderedCommand()+1;
116 c.drawText("PDF-Commands: "+curCmd+"/"+maxCmds, x, y, p);
117  }
118
119 private synchronized void startRenderThread \\
120 (final int page,final float zoom) {
121         if (backgroundThread != null)
122           return;
123         mGraphView.mOffX = 0;
124         mGraphView.mOffY = 0;
125         mGraphView.uiInvalidate();
126         backgroundThread = new Thread(new Runnable() {
127         @Override
128         public void run() {
129         try {
130          if (mPdfFile != null) {
131                 showPage(page, zoom);
132          }
133         } catch (Exception e) {
134         Log.e(TAG, e.getMessage(), e);
135         }
136          backgroundThread = null;
137 }
```

```
138 });
139 updateImageStatus();
140 backgroundThread.start();
141 }
142 
143 private void updateImageStatus() {
144 if (backgroundThread == null) {
145         mGraphView.uiInvalidate();
146         return;
147 }
148 mGraphView.uiInvalidateText();
149 mGraphView.postDelayed(new Runnable() {
150         @Override public void run() {
151         updateImageStatus();
152         }
153         }, 1000);
154 }
155 
156 @Override
157 public boolean onCreateOptionsMenu(Menu menu) {
158 super.onCreateOptionsMenu(menu);
159 menu.add(Menu.NONE, MEN_PREV_PAGE, Menu.NONE, \\
160 "Previous Page");
161 menu.add(Menu.NONE, MEN_NEXT_PAGE, Menu.NONE, \\
162 "Next Page");
163 menu.add(Menu.NONE, MEN_ZOOM_OUT, Menu.NONE, "Zoom Out");
164 menu.add(Menu.NONE, MEN_ZOOM_IN, Menu.NONE, "Zoom In");
165 menu.add(Menu.NONE, MEN_BACK, Menu.NONE, "Back");
166 return true;
167 }
168 
169     /**
170      * Called when a menu item is selected.
171      */
172  @Override
173  public boolean onOptionsItemSelected(MenuItem item) {
174   super.onOptionsItemSelected(item);
175   switch (item.getItemId()) {
176   case MEN_NEXT_PAGE: {
177             nextPage();
178             break;
179   }
180   case MEN_PREV_PAGE: {
181             prevPage();
182             break;
183    }
184    case MEN_ZOOM_IN: {
```

```
185             zoomIn();
186             break;
187     }
188     case MEN_ZOOM_OUT: {
189             zoomOut();
190             break;
191     }
192     case MEN_BACK: {
193       finish();
194       break;
195     }
196 }
197 return true;
198 }
199
200 private void zoomIn() {
201   if (mPdfFile != null) {
202     if (mZoom < 4) {
203             mZoom *= 1.5;
204             if (mZoom > 4)
205             mZoom = 4;
206             startRenderThread(mPage, mZoom);
207             }
208    }
209 }
210
211 private void zoomOut() {
212   if (mPdfFile != null) {
213     if (mZoom > 0.25) {
214             mZoom /= 1.5;
215             if (mZoom < 0.25)
216             mZoom = 0.25f;
217             startRenderThread(mPage, mZoom);
218     }
219  }
220 }
221
222 private void nextPage() {
223  if (mPdfFile != null) {
224   if (mPage < mPdfFile.getNumPages()) {
225     mPage += 1;
226     startRenderThread(mPage, mZoom);
227    }
228   }
229 }
230
231 private void prevPage() {
```

```
232  if (mPdfFile != null) {
233    if (mPage > 1) {
234              mPage -= 1;
235              startRenderThread(mPage, mZoom);
236     }
237  }
238 }
239
240 private class GraphView extends View {
241  private String mText;
242  private float mLastX;
243  private float mLastY;
244  public float mOffX;
245  public float mOffY;
246  private long fileMillis;
247  private long pageMillis;
248  Canvas mCan;
249  Bitmap mBi;
250
251 public GraphView(Context context) {
252    super(context);
253     mOffX = 100;
254     mOffY = 100;
255
256     setPageBitmap();
257     setBackgroundColor(Color.TRANSPARENT);
258 }
259
260 private void showText(String text) {
261   Log.i(TAG, "ST='"+text+"'");
262   mText = text;
263   uiInvalidate();
264 }
265
266 private void uiInvalidate() {
267    postInvalidate();
268 }
269
270 private void uiInvalidateText() {
271   postInvalidate(0, 40, 320, 60);
272 }
273
274 private void setPageBitmap() {
275 mBi = Bitmap.createBitmap(200, 200, Config.RGB_565);
276    mCan = new Canvas(mBi);
277    mCan.drawColor(Color.RED);
278    Paint paint = new Paint();
```

```
279     paint.setColor(Color.BLUE);
280     mCan.drawCircle(50, 50, 50, paint);
281     paint.setStrokeWidth(0);
282     paint.setColor(Color.BLACK);
283      mCan.drawText("Bitmap", 10, 50, paint);
284 }
285 
286 @Override
287 public boolean onTouchEvent(MotionEvent event) {
288   super.onTouchEvent(event);
289   OutputStreamWriter osw;
290   long downtime = 0;
291   if (event.getAction() == MotionEvent.ACTION_DOWN) {
292     mLastX = event.getRawX();
293     mLastY = event.getRawY();
294     float x = event.getX();
295     float y = event.getY();
296     float pressure = event.getPressure();
297     downtime = event.getDownTime();
298     long eventtime = event.getEventTime();
299     float size = event.getSize();
300     long time = eventtime - downtime;
301     float force = pressure * size;
302 
303     try{
304     osw = new OutputStreamWriter(
305     new FileOutputStream(PdfFileSelectActivity.\\
306     targetPath,true),"utf-8");
307             try {
308 osw.write("press_down;"+String.valueOf(mLastX)+\\
309 ";"+ String.valueOf(mLastY)+";"+String.valueOf(x)\\
310 +";"+ String.valueOf(y)+";"+String.valueOf(pressure)\\
311 +";"+String.valueOf(size)+";"+String.valueOf(force)\\
312  +";"+ String.valueOf(downtime)+";"+String.valueOf \\
313 (eventtime)+";"+ String.valueOf(time)+"\n");
314              osw.flush();
315              osw.close();
316             } catch (IOException e) {
317             // TODO Auto-generated catch block
318              e.printStackTrace();
319             }
320     } catch (UnsupportedEncodingException e1) {
321                // TODO Auto-generated catch block
322                 e1.printStackTrace();
323              } catch (FileNotFoundException e1) {
324                // TODO Auto-generated catch block
325                 e1.printStackTrace();
```

```
            }
  }
else if (event.getAction() == MotionEvent.ACTION_MOVE) {
    float rawx = event.getRawX();
    float rawy = event.getRawY();
    float dx = rawx-mLastX;
    float dy = rawy-mLastY;
    float x = event.getX();
    float y = event.getY();

    long eventtime = event.getEventTime();
    downtime = event.getDownTime();
    long time = eventtime - downtime;
    float pressure = event.getPressure();
    float size = event.getSize();
    float force = pressure * size;
    float speedx = dx/time;
    float speedy = dy/time;
    try{
            osw = new OutputStreamWriter(  \\
      new FileOutputStream(PdfFileSelectActivity.\\
      targetPath,true),"utf-8");
    try {
osw.write("press_move;"+String.valueOf(rawx)+\\
";"+String.valueOf(rawy)+";"+String.valueOf(x)+\\
";"+String.valueOf(y)+";"+String.valueOf(pressure)\\
+";"+String.valueOf(size)+";"+String.valueOf(force)\\
+";"+String.valueOf(downtime)+";"+String.valueOf \\
(eventtime)+";"+String.valueOf(time)+";"+ \\
String.valueOf(dx)+";"+String.valueOf(dy)+";"+ \\
String.valueOf(speedx)+ ";"+String.valueOf(speedy)+"\n");
    osw.flush();
    osw.close();
     } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
     }
   } catch (UnsupportedEncodingException e1) {
    // TODO Auto-generated catch block
           e1.printStackTrace();
   } catch (FileNotFoundException e1) {
            // TODO Auto-generated catch block
            e1.printStackTrace();
   }
 mLastX = rawx;
 mLastY = rawy;
 mOffX += dx;
```

```
 mOffY += dy;
 uiInvalidate();
}
 else if (event.getAction() == MotionEvent.ACTION_UP) {
   float rawx = event.getRawX();
   float rawy = event.getRawY();
   float x = event.getX();
   float y = event.getY();
   float pressure = event.getPressure();
   float size = event.getSize();
   float force = pressure * size;
   long eventtime = event.getEventTime();
   long presstime = eventtime - downtime;
   try{
             osw = new OutputStreamWriter(  \\
       new FileOutputStream(PdfFileSelectActivity. \\
       targetPath,true), "utf-8");
             try {
osw.write("press_up;"+String.valueOf(rawx)+";"\\
+String.valueOf(rawy)+";"+String.valueOf(x)+";"\\
+String.valueOf(y)+";"+String.valueOf(pressure)+ \\
";"+String.valueOf(size)+";"+String.valueOf(force)\\
+";"+String.valueOf(downtime)+";"+String.valueOf \\
(eventtime)+";"+String.valueOf(presstime)+"\n");
             osw.flush();
             osw.close();
            } catch (IOException e) {
               // TODO Auto-generated catch block
                e.printStackTrace();
               }
             } catch (UnsupportedEncodingException e1) {
               // TODO Auto-generated catch block
                e1.printStackTrace();
             } catch (FileNotFoundException e1) {
                // TODO Auto-generated catch block
                  e1.printStackTrace();
               }
     }
     return true;
}

@Override
protected void onDraw(Canvas canvas) {
 Paint paint = new Paint();
canvas.drawColor(Color.LTGRAY);

  paint.setStrokeWidth(0);
```

```
  paint.setColor(Color.BLACK);
  canvas.drawText("PdfViewer: "+mText, 10, 20, paint);

  float fileTime = fileMillis*0.001f;
  float pageTime = pageMillis*0.001f;
  canvas.drawText("seconds: parse="+fileTime+ \\
  " show="+pageTime, 10, 40, paint);

  showRenderStatus(canvas, 10, 60, paint);
    // draw the normal strings
paint.setColor(Color.BLUE);
canvas.drawCircle(mOffX, mOffY, 5, paint);
canvas.drawCircle(mOffX+mBi.getWidth(),\\
 mOffY, 5, paint);
canvas.drawCircle(mOffX+mBi.getWidth(),\\
 mOffY+mBi.getHeight(), 5, paint);
canvas.drawCircle(mOffX, mOffY+mBi.\\
getHeight(), 5, paint);
canvas.drawBitmap(mBi, mOffX, mOffY, paint);
 }
}

private void showPage(int page, float zoom) \\
throws Exception {
  long startTime = System.currentTimeMillis();
   try {
        mPdfPage = mPdfFile.getPage(page, true);
        int num = mPdfPage.getPageNumber();
      int maxNum = mPdfFile.getNumPages();
      float wi = mPdfPage.getWidth();
      float hei = mPdfPage.getHeight();
      String pageInfo= new File(pdffilename).getName()\\
      + " - " + num +"/"+maxNum+ ": " + wi + "x" + hei;
        mGraphView.showText(pageInfo);
        Log.i(TAG, pageInfo);
        RectF clip = null;
         // free memory from previous page
         mGraphView.setPageBitmap();
         mGraphView.mBi = mPdfPage.getImage( \\
(int)(wi*zoom),(int)(hei*zoom), clip, true, true);
          mGraphView.uiInvalidate();
        } catch (Throwable e) {
        Log.e(TAG, e.getMessage(), e);
mGraphView.showText("Exception: "+e.getMessage());
        }
    long stopTime = System.currentTimeMillis();
    mGraphView.pageMillis = stopTime-startTime;
```

```
}

private void parsePDF(String filename,\\
byte[] pdfBinary) {
 long startTime = System.currentTimeMillis();
 try {
  if (pdfBinary != null) {
        long len = pdfBinary.length;
       mGraphView.showText("uri '" + filename + \\
      "' has " + len + " bytes");
            openFile(null, pdfBinary);
           }
   else {
        File f = new File(filename);
        long len = f.length();
        if (len == 0) {
        mGraphView.showText("file '" + filename + \\
     "' not found");
        }
        else {
        mGraphView.showText("file '" + filename + \\
      "' has " + len \\
      + " bytes");
        openFile(f, null);
        }
   }
} catch (Throwable e) {
        e.printStackTrace();
mGraphView.showText("Exception: "+e.getMessage());
}
 long stopTime = System.currentTimeMillis();
 mGraphView.fileMillis = stopTime-startTime;
}

public void openFile(File file, byte[] pdfBinary) \\
throws IOException {
  byte[] buf = pdfBinary;
  if (buf == null)
  buf = readBytes(file);
  mPdfFile = new PDFFile(new ByteBuffer(buf));
 mGraphView.showText("Anzahl Seiten:" \\
+ mPdfFile.getNumPages());
}

private byte[] readBytes(File srcFile) \\
throws IOException {
long fileLength = srcFile.length();
```

```
514 int len = (int)fileLength;
515 byte[] result = new byte[len];
516 FileInputStream fis = new FileInputStream(srcFile);
517 int pos = 0;
518 int cnt = fis.read(result, pos, len-pos);
519 while (cnt > 0) {
520    pos += cnt;
521    cnt = fis.read(result, pos, len-pos);
522 }
523 return result;
524 }
525
526 private byte[] readUriContent(Uri uri) {
527   byte[] result = null;
528   try {
529    InputStream is = getContentResolver().\\
530    openInputStream(uri);
531    int size = is.available();
532    result = new byte[size];
533    int pos = 0;
534    int cnt = is.read(result, pos, size-pos);
535    while (cnt > 0) {
536      pos += cnt;
537      cnt = is.read(result, pos, size-pos);
538     }
539   is.close();
540   }
541   catch (Exception e) {
542           Log.e(TAG, e.getMessage(), e);
543   }
544   return result;
545 }
546
547 @Override
548 protected void onDestroy() {
549    super.onDestroy();
550   if (mTmpFile != null) {
551     mTmpFile.delete();
552     mTmpFile = null;
553   }
554  }
555
556  Runnable mRenderTask = new Runnable() {
557  public void run() {
558  }
559  };
```

```
}
```

B.3 XML File of Window Layout for Typing Username Creating Log File

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android= \\
   "http://schemas.android.com/apk/res/android"
   android:orientation="vertical"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent"
    >

<!-- filename -->
    <TextView
        android:layout_width="wrap_content" \\
        android:layout_height="wrap_content"
        android:layout_weight="0"
        android:paddingBottom="0dip"
        android:text="Creat Test User:"/>
    <EditText
        android:id="@+id/filename"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:paddingBottom="0dip"
        android:text=""
android:hint="please enter your name (e.g. xiaoli01)" />

        <!-- Show Images -->
    <CheckBox android:id="@+id/cbAntiAlias"
            android:paddingBottom="24sp"
                android:paddingTop="24sp"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="antialias" />

        <!-- Button 'Show' -->
        <LinearLayout android:layout_width="fill_parent"
        android:layout_height="wrap_content" \\
        android:gravity="center_horizontal"
        android:orientation="horizontal" >
        <Button android:id="@+id/btShow" \\
        android:layout_width="120px"
android:layout_height="40px" android:text="Start">
</Button>
```

```
        <Button android:id="@+id/btExit" \\
        android:layout_width="120px"
        android:layout_height="40px" \\
        android:text="Exit"></Button>
        </LinearLayout>

        <!-- Output -->
    <EditText android:id="@+id/output"
        android:layout_width="fill_parent" \\
          android:autoText="true"
        android:capitalize="sentences"
        android:layout_weight="1"
 android:freezesText="true" android:layout_height="0dip"
 android:text="[enter your name and press 'start']">
        <requestFocus />
    </EditText>
    <ListView
                android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:id="@android:id/list"
            android:visibility="invisible"
        />

</LinearLayout>

```

B.4 XML File of Window Layout for Reading Process of Showing Page Content

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android= \\
"http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@drawable/violet"
    >
        <!-- Current absolute path -->
           <TextView
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:id="@+id/mPath"
            android:padding="5px"
            android:textSize="18sp"
            android:textColor="@drawable/white"
```

```
17      />
18      <!-- Current File Explorer Hierarchy -->
19          <ListView
20                  android:layout_width="wrap_content"
21              android:layout_height="wrap_content"
22              android:id="@android:id/list"
23          />
24
25  </LinearLayout>
```