ACCELEROMETER BASED HANDWRITTEN CHARACTER RECOGNITION USING DYNAMIC TIME WARPING

A Thesis Submitted to
the Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in Electronics and Communication Engineering

by
Esra TUNÇER

July 2016
İZMİR

We approve the thesis of **Esra TUNÇER**

Examining Committee Members:

Assist. Prof. Dr. Mehmet Zübeyir ÜNLÜ
Department of Electrical and Electronics Engineering, Izmir Institute of Technology

Assoc. Prof. Dr. Şevket GÜMÜŞTEKİN
Department of Electrical and Electronics Engineering, Izmir Institute of Technology

Assoc. Prof. Dr. Mehmet ENGİN
Department of Electrical and Electronics Engineering, Ege University

19 July 2016

Assist. Prof. Dr. Mehmet Zübeyir ÜNLÜ
Supervisor
Department of Electrical and Electronics Engineering, Izmir Institute of Technology

Prof. Dr. Mehmet Salih DİNLEYİCİ
Head of the Department of
Electrical and Electronics Engineering

Prof. Dr. Bilge KARAÇALI
Dean of the Graduate School of
Engineering and Sciences

ACKNOWLEDGEMENTS

At the end of my M.Sc. thesis, firstly I would like to express my sincere gratitude to my parents for their endless support. Nothing would be done without them.

I would like to thank my supervisor Assist. Prof. Dr. Mehmet Zübeyir ÜNLÜ for his trust and recommendations throughout my research. I would like to express my gratitude to Assoc. Prof. Dr. Şevket GÜMÜŞTEKİN and Assoc. Prof. Dr. Mehmet ENGİN for accepting to be a committee member.

I would like to thank my friends Simge VARUŞ, Ayşenur BAYRAM and Merve Bedeloğlu for their sincere and endless friendship. And I would like to thank to Bilal Orkan OLCAY and Aslı TAŞÇI for encouraging me and for being volunteer during the data acquisition.

I would like to thank to Gökmen AŞCIOĞLU for his friendship, recommendations and support of data acquiring.

ABSTRACT

ACCELEROMETER BASED HANDWRITTEN CHARACTER RECOGNITION USING DYNAMIC TIME WARPING

Character and gesture recognition are one of the most studied topics in recent years. Character recognition studies are generally based on image processing. Only a few studies can be found about character recognition as gesture recognition. Gesture recognition is making the computers understand human body movements by using different kind of knowledge of the environment. This knowledge can be obtained by image or sensor-based efforts. Accelerometer is the most used sensor in gesture recognition, so in this study a 3-axis accelerometer is used.

In this thesis, English alphabet’s lowercase characters are used. A ring-like device which contains accelerometer in it is used. After obtaining the acceleration data of each character with 20 repetitions, we apply filtering, segmentation and normalization preprocessing steps for each signal. Since there are different accelerations and decelerations between each repetitions, Dynamic Time Warping (DTW) algorithm has been chosen to determine the similarities between signals. DTW is an algorithm that uses the amplitude values of the signals, so it is weak to amplitudes that shift in time domain. To overcome this shortcoming, the method called Derivative Dynamic Time Warping (DDTW) has been applied to the acceleration signals. DTW and DDTW methods have been compared and we have found that even we remove the normalization step; DDTW gives better results than DTW. By comparison of linear alignment and DTW, the results show that DTW gives better recognition rates for signals with different accelerations and decelerations. DTW also gives better result for the different length signals.

ÖZET

DİNAMIK ZAMAN BÜKME METODU KULLANARAK İVMEÖLÇER TABANLI EL YAZISI KARAKTER TANIMA

Karakter ve vücut hareketi tanıma son yıllarda en çok çalışılan alanlardan bir tanesidir. Literatürdeki karakter tanıma çalışmaları çoğunlukla görüntü işleme tabanlı olmakla birlikte bu tezde yapılan çalışmada sensör tabanlı bir yöntem uygulanmıştır. Literatürde vücut hareketlerinin tanımlanması yoluyla yapılan karakter tanıma çalışmalarının sayısı ise oldukça sınırlıdır. Vücut hareketi tanıma kısaca bilgisayarların insan hareketlerini anlaması ve bilgisayarların bu yolla kontrolüdür. Çevre değişkenlerini algılamak için kameralar veya çeşitli sensörler kullanılır. Vücut hareketlerini tanıma çalışmalarında ise sıklıkla ivme sensörleri kullanılmaktadır.

Bu çalışmada İngiliz alfabesindeki küçük harfler veri tabanındaki sınıfları oluşturmuştur. Bu harflerin yazılma aşamasındaki dinamiklerini ölçmek amacıyla yüzük şeklinde bir yapıya ilişkilendirilen ivme sensörü kullanılmıştır. Harflerin ivme verileri her bir harf için 20 tekrarlı olarak elde edildikten sonra tüm sinyaller bir önişlem aşaması olarak filtreleme, bölütleme ve normalize etme işlemlerinden geçirilmiştir. Tek bir harf için bile tekrarlar arasında farklı ivmelenme değerleri oluştuğundan ve farklı uzunluklarda sinyaller elde edildiğinden sinyaller arasındaki benzerliği belirlemek amacıyla Dinamik Zaman Bükme (DTW) metodu sinyaller arasındaki benzememeyi bulmak için seçildi. Bu yöntem sinyallerin genlik değerlerini kullandığı için eşleşmesi gereken noktaların genlik değerlerindeki farklılıklarda yetersiz kalabiliyor. Bu sorunun çözümü için Türevli Dinamik Zaman Bükme metodu sinyallere uygulanmış buna ek olarak DTW'yi hızlandırmak için sinyaller nicelenip ayrıca incelenmiştir. Deneysel çalışmalarda normalize işlemi olmadığı durumda bile DDTW, DTW'e göre daha yüksek tanıma oranı vermiştir. Lineer eşleşme ile DTW'nin karşılaştırılmasında da zamana göre farklı ivmelenmeye sahip sinyallerde lineer olmayan eşleşmenin daha bir tanıma oranı verdiği görülmüştür.

TABLE OF CONTENTS

LIST OF FIGURES viii

LIST OF TABLES xi

LIST OF ABBREVIATIONS xii

CHAPTER 1. INTRODUCTION TO CHARACTER RECOGNITION 1

1.1. Optical Character Recognition 2

1.2. Character Recognition As Gesture Recognition 5

CHAPTER 2. PHYSICS BACKGROUND 10

2.1. Motion 10

2.1.1. Position 10

2.1.2. Velocity & Acceleration 11

2.2. Acceleration Sensor, Gravitational Force 14

CHAPTER 3. RECOGNITION 20

3.1. Linear Alignment 20

3.1.1. Euclidean Distance 21

3.1.2. Manhattan Distance 22

3.1.3. Chessboard Distance 22

3.2. Nonlinear Alignment - Dynamic Time Warping (DTW) 23

3.2.1. Singularity Problem And DDTW 29

3.2.2.Comparison Of Other Alternative Methods To Increase The Performance 31

CHAPTER 4. EXPERIMENTAL WORK AND RESULTS 34

4.1.Hardware 34

4.1.1.Accelerometer 35

4.1.2.Microcontroller 36

4.1.3.Micro SD Card Module 37

4.2.Preprocessing 37
4.2.1.Filtering 38
4.2.2.Separation 46
4.2.3.Normalization 49
4.3.Results 51
4.3.1.Raw Data 51
4.3.2.Derivative Data 62
4.3.3.Data After Quantization 71

CHAPTER 5. CONCLUSION 78

REFERENCES 81

LIST OF FIGURES

Figure **Page**

Figure 1. First Alphabets 1
Figure 2. Characters in English Alphabet 2
Figure 3. Study Areas of Optical Character Recognition 3
Figure 4. Scurry 6
Figure 5. Gestures 8
Figure 6. TI eZ430-Chronos Watch 9
Figure 7. Position Graph as an example 11
Figure 8. Velocity Graph of the Position Graph in Figure 7 12
Figure 9. Position Graph with changeable slope 13
Figure 10. No Gravitational Force 15
Figure 11. No Gravitational Force and the structure is moved to left direction 16
Figure 12. Gravitational Force with 1g 16
Figure 13. Gravitational Force with 1g and the structure is moved to left direction 17
Figure 14. The raw data signals of ‘a’ character 18
Figure 15. The raw data signals of ‘b’ character 18
Figure 16. The raw data signals of ‘c’ character 19
Figure 17. Distance of $\boldsymbol{s}$ and $\boldsymbol{t}$ points 21
Figure 18. Repetitions of the same movement 23
Figure 19. Linear Alignment and Nonlinear Alignment of the signals in Figure 18 24
Figure 20. An example of distance matrix 25
Figure 21. Flowchart for creating the Distance matrix 25
Figure 22. Some of the possible warping paths 28
Figure 23. Warping path restricted by a warping window 29
Figure 24. True matching and matching with singularities 30
Figure 25. Overall Structure of the Hardware 34
Figure 26. Data Acquisition Device 35
Figure 27. MPU-6050 36
Figure 28. ARM mbed NXP LPC1768 Development Board 36
Figure 29. Micro SD Card Module 37
Figure 30. Simple Block Diagram of Preprocessing 38

Figure 31. Block Diagram of Overall System 38
Figure 32. Single-Sided Amplitude Spectrum of acceleration signal............................ 39
Figure 33. Single-Sided Amplitude Spectrum of acceleration signal............................ 40
Figure 34. Frequency response of the MAF 43
Figure 35. Comparing the frequency responses of the MAF... 44
Figure 36. Raw data and filtered data (L=15)... 45
Figure 37. Single-Sided Amplitude Spectrum of raw data and filtered data 45
Figure 38. Filtered repetitions of the lower case ‘a’ character 46
Figure 39. Amplitude change between samples of the signal... 47
Figure 40. Detecting the first and last data points of the signal..................................... 48
Figure 41. Repetition signals before normalization... 50
Figure 42. Repetition signals after normalization.. 51
Figure 43. Accuracy of each character (linear alignment with Euclidean) 52
Figure 44. Accuracy of each character (linear alignment with Manhattan) 53
Figure 45. Accuracy of each character (linear alignment with Chessboard).................. 54
Figure 46. Accuracy of each character (DTW with Euclidean) 55
Figure 47. Accuracy of each character (DTW with Manhattan) 56
Figure 48. Accuracy of each character (DTW with Chessboard).................................. 57
Figure 49. Accuracy of each character (Correlation Coefficient) 58
Figure 50. Accurate recognition rate of methods for raw data 59
Figure 51. Time duration of each method for raw data 59
Figure 52. Accurate recognition rate of methods for raw data (user independent) 61
Figure 53. Time duration of each method for raw data (user independent) 61
Figure 54. Accuracy of each character (Linear Alignment with Euclidean) 62
Figure 55. Accuracy of each character (Linear Alignment with Manhattan)................. 63
Figure 56. Accuracy of each character (Linear Alignment with Chessboard) 64
Figure 57. Accuracy of each character (DTW with Euclidean) 65
Figure 58. Accuracy of each character (DTW with Manhattan) 66
Figure 59. Accuracy of each character (DTW with Chessboard).................................. 67
Figure 60. Accuracy of each character (Correlation Coefficient) 68
Figure 61. Accurate recognition rate of methods for derivative data 69
Figure 62. Time duration of each method for derivative data 69
Figure 63. Accurate recognition rate of methods for raw data (digits)........................... 70
Figure 64. Accurate recognition rate of methods for derivative data (digits)................. 70

Figure 65. Accuracy of each character (linear alignment) 71
Figure 66. Accuracy of each character (DTW with Euclidean) 72
Figure 67. Accuracy of each character (DTW with Manhattan) 73
Figure 68. Accuracy of each character (DTW with Chessboard) 74
Figure 69. Accurate recognition rate of methods for data after quantization 75
Figure 70. Time duration of each method for data after quantization 75
Figure 71. Overall recognition rates of DTW 76
Figure 72. Overall duration of DTW 76

LIST OF TABLES

Table **Page**

Table 1. Slope of the position graph in Figure 7 12

Table 2. Quantization Values... 31

Table 3. Quantization values that used in this thesis 32

Table 4. Recognition test for characters (linear alignment with Euclidean) 52

Table 5. Recognition test for characters (linear alignment with Manhattan) 53

Table 6. Recognition test for characters (linear alignment with Chessboard)................ 54

Table 7. Recognition test for characters (DTW with Euclidean) 55

Table 8. Recognition test for characters (DTW with Manhattan) 56

Table 9. Recognition test for characters (DTW with Chessboard)................................. 57

Table 10. Recognition test for characters (Correlation Coefficient) 58

Table 11. Recognition test for characters (Linear Alignment with Euclidean).............. 62

Table 12. Recognition test for characters (Linear Alignment with Manhattan)............. 63

Table 13. Recognition test for characters (Linear Alignment with Chessboard) 64

Table 14. Recognition test for characters (DTW with Euclidean) 65

Table 15. Recognition test for characters (DTW with Manhattan) 66

Table 16. Recognition test for characters (DTW with Chessboard)............................... 67

Table 17. Recognition test for characters (Correlation Coefficient) 68

Table 18. Recognition test for characters (linear alignment).. 71

Table 19. Recognition test for characters (DTW with Euclidean) 72

Table 20. Recognition test for characters (DTW with Manhattan) 73

Table 21. Recognition test for characters (DTW with Chessboard)............................... 74

LIST OF ABBREVIATIONS

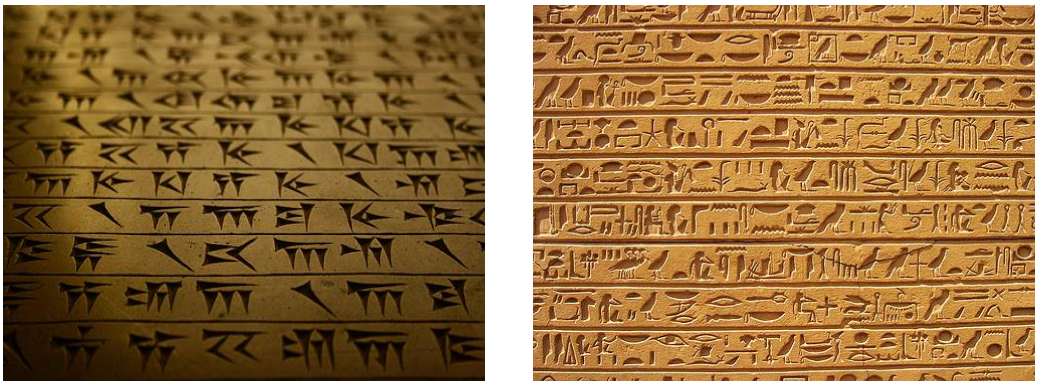
OCR .... Optical Character Recognition
PCA .... Principal Component Analysis
HMM .... Hidden Markov Model
DTW .... Dynamic Time Warping
MEMS .... Micro Electro Mechanical Systems
TI .... Texas Instruments
ZCD .... Zero Crossing Detector
PNN .... Probabilistic Neural Network
SI .... International System of Units
DDTW .... Derivative Dynamic Time Warping
ADC .... Analog to Digital Converter
DAC .... Digital to Analog Converter
DFT .... Discrete Fourier Transform
FFT .... Fast Fourier Transform
MAF .... Moving Average Filter
DSP .... Digital Signal Processing

CHAPTER 1

INTRODUCTION TO CHARACTER RECOGNITION

A character or a letter is a symbol that related to a man-made sound. The group of letters that contain all man-made sound in a language is that language's alphabet. Every civilization needs writing because they need to store and pass the knowledge on to the next generations. There are two different beliefs that why the writing was invented. One of these beliefs is that the writing is invented for religion and the other belief is that the writing is invented for trade and taxation.

Before the first alphabet was invented, the writing is like looking at a picture and understanding the meaning of the picture. That kind of writing can be seen in the Chinese characters and Egyptian hieroglyphics, or on cuneiform wedges produced by pressing a stylus into soft clay. Each character of these alphabets represents a word, so it can be said that the writing was limited to a small well educated people since it is hard to learn each and every character that represents each word. (Schumm, 2014)



(a) (b)

Figure 1. First Alphabets (a) Cuneiform script (b) Egyptian hieroglyphics (Source:("Çivi yazısı nedir?," 2011)) (Source:("Hieroglyph," 2016))

During the second millennium B.C Semitic speaking people started to use some of the adapted Egyptian hieroglyph characters to represent the sounds of their language. This script was considered as the first writing that use the alphabet and in this system the characters are only consonants but not vowels. It was written from right to the left and this alphabet contained of 22 characters that everyone can learn and write easily. So its use became accessible and widespread. By the 8th century this alphabet had spread to Greece. Some characters were kept and some of them removed but the biggest change was using the vowel characters. By the 5th century BC the writing direction had changed the way we use today, from left to the right. As time progresses, the Greek alphabet lead to the other alphabets like Latin and Cyrillic. (Schumm, 2014)

Nowadays, so many different alphabets are used. Latin alphabet, also called Roman alphabet, most widely used alphabetic writing system in the world. It is the script of the English language and the languages of the most of Europe.

Some languages that use the Latin alphabet add extra letters to indicate the special sounds belong to that language. Turkish is one of the languages that have different voices than English, so Turkish alphabet has some different letters that Latin alphabet does not contain. These letters are Ç-ç, Ğ-ğ, I-ı, Ö-ö, Ş-ş, Ü-ü. Also the Latin alphabet has 3 additional letters, which means there are 3 voices that does not used in Turkish alphabet. These not used letters are Q-q, W-w and X-x. So the Turkish alphabet contains 29 letters.

Aa Bb Cc Dd Ee Ff Gg Hh Ii Jj Kk Ll Mm Nn Oo Pp
Qq Rr Ss Tt Uu Vv Ww Xx Yy Zz

Figure 2. Characters in English Alphabet

1.1. Optical Character Recognition

Nowadays, optical character recognition is one of the most popular research areas of pattern recognition and classification. Optical character recognition (OCR) is the transformation of printed, handwritten or typed character (letter) images into the machine or computer readable characters.

The OCR is generally used for passport documents, bank statements, computerized receipts, business cards, mail or any suitable documentation. And it is also used by libraries to digitized the books and make them accessible online for more than one user at the same time without contact. The most advantageous part of it is that by digitizing, it can be easy and faster to find the suitable books or sources that contain a specified topic. Some university libraries like Harvard, Stanford, Michigan, Oxford and the New York Public Library work with Google to digitalize the books from their collections so the users can search these books from Google.("Google Checks Out Library Books," 2004)

There are so many web sites that can transform scanned or photographed text images into the text files that preferred. One of these web sites is OnlineOCR.net that is a free web-based Optical Character Recognition software (OCR) that allows to transform scanned PDF documents, faxes, photographs or digital camera images into the searchable electronic documents like Adobe® PDF, Microsoft Word®, Microsoft Excel® and txt. OnlineOCR.net supports 46 languages such as English, Danish, Dutch, French, German, Italian, Portuguese, Russian, Spanish, Japanese, Chinese and Korean. ("Online OCR," 2016)

The image below shows the different study areas of optical character recognition. OCR can be split into the two main groups which are offline OCR and online OCR. In offline OCR, the recognition is done for the characters or the text, written before the recognition process. But in online OCR, the recognition and writing processes are done at the same time.

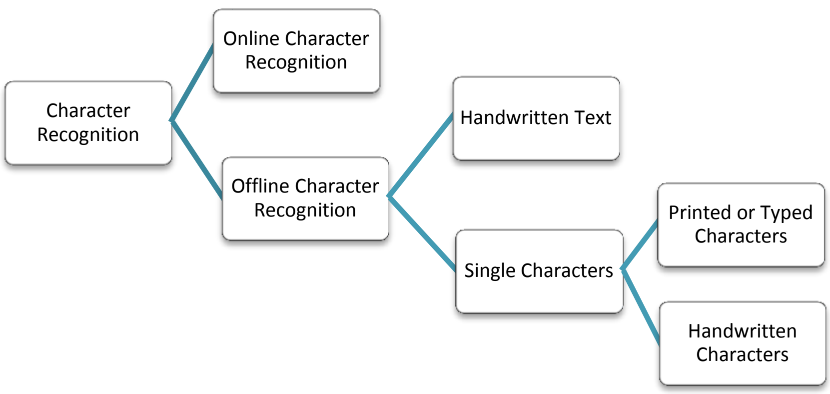


Figure 3. Study Areas of Optical Character Recognition

Also offline OCR can be split into two subgroups: handwritten text recognition and single character recognition which has two subgroups: printed or typed characters and handwritten characters. It may be thought that there is no difference between handwritten character recognition and handwritten text recognition. Actually it is right and wrong at the same time. There is no difference between them if there are blank spaces between every characters or letters in the text but if there are no blank space and if the characters are written continuously in the text there will be a separation problem. The separation of each character from the text can be done easily with the existence of the blank spaces and the text image can be considered as separated character image sequence. In the condition that there are no spaces, other different approaches need to be applied to the text image.

Because of the stable character shape for a single character, the transformations on printed and typed character images are easier than transformations on handwritten character images. There are so many studies on printed or typed characters recognition and the accurate recognition rate of these studies is very high. But generally, accurate recognition rate on handwritten characters is not that much high comparing to the printed or typed characters recognition. The main reason of the less rated accuracy is that, even for a single character, it is almost impossible to write the character in the exactly same way. The recognition techniques used for printed or typed characters are not enough to recognize the characters or to reach the accuracy rate that can be reached with printed or typed characters. The techniques used for printed character images can be improved or new techniques can be used or other extra knowledge about writing process can be used with images.

Generally, in every OCR study, before the recognition the preprocessing step needs to be applied to the raw text images. In preprocessing step, firstly the text images are needed to be segmented into the lines of the text, then lines segmented into the words and finally the words segmented into the characters or letters. After segmentation the text into characters, each image of the characters needs to be smoothed and the noise caused from the scanning or photographing process should be removed. By removing the noise, false recognition may be avoided so the recognition rate will be higher.

Almost all of the pattern recognition studies, the most important part of the studies are choosing the features that are essential for the recognition. In view of the fact that the character recognition is also a pattern recognition study, the features that represent the characteristics of the characters, need to be determined.

The recognition can also be done with template matching and using other correlation techniques. In these techniques, there is no feature extraction step. Instead of using the features, the templates or prototypes that represent each class of the characters are expected to be matched with the input image or input character.

1.2. Character Recognition as Gesture Recognition

Gesture recognition is a research topic making computers understand the human body language. To recognize gestures there are two main techniques. One of them is performed by using a camera to obtain the movements of the body parts. In this technique image processing algorithms are used. But this is not a technique that used in our study. In this study, the other technique, which is based on using and obtaining the data from a sensor, is used.

In this project, an acceleration sensor is used to obtain the acceleration values of the writing movement. A 3-axis acceleration sensor is put together with a ring-like device and the writing process is done on a plain surface. We may think that writing is a kind of gesture but with more complex movements. In the literature, researchers, studying with gesture recognition, use simple gestures to find the accurate recognition rate of their algorithms and systems. But in this thesis the aim is to recognize the entire lower case English alphabet, which contains 26 different characters. This means that we have 26 different gestures, with only acceleration data. In the literature, the number of the gestures defined is generally less than this number. Below, some of the related studies in the literature are explained.

In the study of (Perng, 1999) et. al., acceleration sensing glove is introduced. This glove has accelerometers on the finger tips and on the back of the hand. They only use the effect of the gravity on acceleration sensors. This glove is designed for working as a computer mouse. The acceleration sensor on the back of the hand is used for the tilt motion to move the pointer on the screen and other sensors on the fingers are used as the click buttons of the mouse.

A device called Magic Wand is presented by (Sung-Jung Cho, 2004) et al. In their study, acceleration and also angular velocity signals are used to recognize the user's hand movements. To convert the acceleration and angular velocity signals into trajectory, an estimation algorithm is used. For recognition part, Bayesian networks are used to find maximum likelihood between gestures. The accurate recognition rate of this study is 99.2%, which is the highest rate among the studies recognizing gestures, for 13 different gestures which are digits and extra 3 simple gestures.

Kim et al. (Kim, 2005) introduces a new wearable device, developed by Samsung Advanced Institute of Technology, called Scurry. As it can be seen from the Figure 4, it is a glove-like device, which can be worn on the human hand. It includes a controller and two gyroscopes for angular velocity on the back of the user's hand as the base module and four ring type modules, including 2-axis acceleration sensors (accelerometers) on four fingers. It is a device that developed for controlling a computer or a device with the movement of hand and finger clicking.

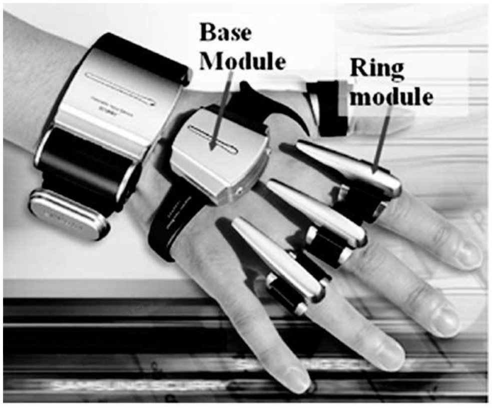


Figure 4. Scurry (Source:(Kim, 2005))

In paper of (Sung-Do Choi, 2006) et al. beside of the acceleration signals they transform the signals into the velocity and position data as new features for 10 digit writing movements. The writings are 3D but with PCA, they transform them into 2D so they reduce the feature dimension. In this study, the DTW algorithm gives 90% accurate recognition rate.

In (Shiqi Zhang, 2008) et al. paper online digit recognition system is presented. The recognition algorithm in the study is Hidden Markov Model (HMM). They also use different quantization techniques to quantize the acceleration data into small integral vectors. The accurate recognition rate of their study is 94.29% for 10 digits. The acceleration trajectories are converted into serial numbers that contain angle information in polar form, and 4 division and 6 division quantization methods are applied to them.

In (Klingmann, 2009)'s thesis an iPhone and its build in accelerometer is used to recognize the gestures. This accelerometer is a 3-axis accelerometer and the measurement range of it is between +2g and -2g. In order to run experiments and find the recognition rate of their algorithm, they use five different and simple gestures which are circle, square, triangle, z and bowling (it is a movement like throwing a bowling ball.). They use HMM for recognition part of the thesis and they achieve the recognition rate of over 90 percent with 10 training sequences.

(Ahmad Akl 2010) et al. proposed a gesture recognition system based on a 3-axis accelerometer. To find the signals matched they use DTW and Affinity Propagation. They use 18 different and simple gestures for recognition and some of their gestures are taken from the Nokia's gesture vocabulary. In their database there are almost 3700 repetitions taken from 7 users for 18 gestures and they suppose that their database is the largest database among the accelerometer based gesture recognition studies. The average accurate recognition rate of their system is 94%.

In (Liu, 2010)'s et. al. study, they introduce an accelerometer based human computer interaction method and its application on virtual reality system. For recognition they use HMM and try to classify 8 basic gestures which are used to move left/turn left, move right/turn right, rotate left, rotate right, move up, move down, confirm/select/zoom in and cancel/delete/zoom out in the virtual reality system. For training 150 repetitions and for testing 50 repetitions for each gesture are used and they obtain 93% recognition accuracy on average.

In (Xu, 2012) et.al. paper, they describe three different gesture recognition methods. They use seven simple gestures for calculating accurate recognition rate of these three methods. These gestures are up, down, circle, left, right, tick and cross obtained from 3-axis acceleration. In their study, they present three different gesture recognition models which are sign sequence and Hopfield based gesture recognition model, velocity increment based gesture recognition model and sign sequence, and template matching based gesture recognition model. The shape of the acceleration signals of each gesture is not important since they only use the alternate sign change in the signal and there is no any filter used in their study to smooth the acceleration data because in their experimental work they notice that the noise does not influence the trend of the acceleration curves. By only using the sign changes in signal, they can reduce the data values of each gesture to a gesture code with 8 numbers. In recognition part they compare this eight valued gesture data with the stored templates. The presented models in their study achieve an overall recognition accuracy of 95.6%.

(Jeen-Shing Wang, 2013) et al. present an accelerometer based pen device for character recognition. The users write the characters on air without any limitation on space. They use DTW to find the similarities between signals and template signals. The gestures used in the study are not simple gestures as in the previous studies (Klingmann, 2009; Mace, 2013), but digits with normal writing style. The average recognition rate of the study is 87.7%.

(Meenaakumari, 2013) et. al. present a MEMS (micro electro mechanical system) accelerometer based hand gesture recognition system. They use triaxial mems accelerometer, microcontroller and zigbee wireless transmission module. To make the recognition accuracy rate higher, they have limited the writing style. The users write digits and characters of the alphabet in digital form which means they can only use 4 different hand gestures. These gestures are up, down, right, left as shown in the Figure 5. And other extra limitation of writing is that after writing of each character a single stroke is needed to make the digits and characters recognizable. This single stroke is done to find the starting and ending points of the writing movement. The handwritten digit recognition rate is 98% and the gesture recognition rate is 98.75%.

1	2	3	4
←	→	↑	↓

Figure 5. Gestures
(Source:(Meenaakumari, 2013))

In (Mace, 2013)'s et. al. paper, they introduce a comparison of the results of naïve Bayesian classification with feature separability weighting and Dynamic Time Warping. They test these techniques with 4 gestures (circle, figure eight, square, star) having five repetitions from five different people. The average accuracy is 97% for the feature separability weighted Bayesian Classifier and 95% for the DTW. They use TI eZ430-Chronos Watch for the accelerometer data. The watch contains a VTI-CMA3000 3-axis accelerometer, with a measurement range of 2g, 8-bit resolution, and 100Hz sampling rate.

Figure 6. TI eZ430-Chronos Watch
(Source:(Texas-Instruments, 2015))

(Patil, 2014) et. al. describe accelerometer based gesture recognition method for digit recognition. But in their study, they write digits in normal writing style not in digital style like in (Meenaakumari, 2013). To write the digits they use 3D input digital pen that contain triaxial accelerometer in it. They generated the feature vectors from the acceleration signal using zero crossing detector (ZCD) and range with Matlab tool. For classification they use Probabilistic Neural Network (PNN) and this system gives 80% accuracy in recognition.

(Shashidhar Patil, 2015) et al. is the most likely study in the literature with our study with gesture type. They use 26 lowercase English characters too and they also use 10 digits. They use DTW as well. The difference between their study and our study is that they write the characters and digits in 3D space and the other difference is that they use accelerations, angular velocities, and orientations of the moving hand so they have more input data to recognition. The recognition rate of their study is 97.95%.

CHAPTER 2

PHYSICS BACKGROUND

2.1. Motion

In this thesis accelerometer based character recognition is performed so character recognition can be thought as gesture recognition. In both ways there is a movement and this movement is measured with single-axis or multi-axis accelerometers. The output of this accelerometer gives simply acceleration of the writing movement. To understand what the acceleration data are and how to obtain it, we need to study briefly the terms of the motion which are position, velocity and acceleration.

2.1.1. Position

To explain the motion of an object, firstly its position is needed to be explained. The place of the object at any particular time is that object's position. More precisely, its position relative to a reference point is needed to be specified. Generally, Earth is used as a reference point. In some other cases, the reference points are not stationary but are in motion.

To describe an object's position in 3 dimensions, generally three variables are used. For representing the horizontal position the variable x is often used. For representing the vertical position the variable y is used and for the third perpendicular axis that points out of the screen page the variable *z* is used.

The objects position changes if an object moves relative to a reference point. This position change can also be named as displacement. The following equation defines the displacement.

$$Displacement = \Delta x = x_f - x_0 \tag{2.1}$$

where x_f is the value of the final position, x_0 is the value of the initial position and Δx is the displacement symbol. The equation 2.1 gives the displacement on X-axis. To find the displacements on Y-axis and Z-axis following equations can be used.

$$Displacement = \Delta y = y_f - y_0 \tag{2.2}$$

$$Displacement = \Delta z = z_f - z_0 \tag{2.3}$$

2.1.2. Velocity & Acceleration

If the change in the position of an object is divided by time duration of this change then it defines the average velocity (Serway, 2008).

$$V_{avg} = \frac{\Delta x}{\Delta t} = \frac{x_f - x_0}{t_f - t_0} \tag{2.4}$$

where V_{avg} is the average velocity, Δx is the position change(displacement), x_f and x_0 are final and initial positions respectively and t_f and t_0 are final and initial times respectively. If the initial time is zero ($t_0 = 0$), then the average velocity equation (2.4) will be

$$V_{avg} = \frac{\Delta x}{\Delta t} = \frac{x_f - x_0}{t_f} \tag{2.5}$$

The SI unit for velocity is meters per second (m/s).

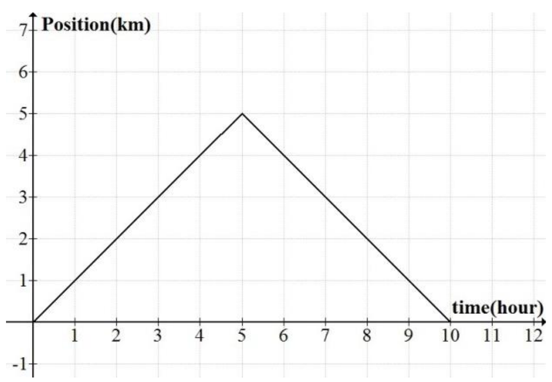


Figure 7. Position Graph as an example

The velocity of an object can be found from the slope of a position graph of this object. So the velocity of an object at a particular time is the value of the slope at that time. If we consider the position graph in the Figure 7 as an example and try to find the slope with the equation below we can obtain the slope values as in Table 1.

$$slope = \frac{x_2 - x_1}{t_2 - t_1} \tag{2.6}$$

Table 1. Slope of the position graph in Figure 7

Time interval	0h-1h	1h-2h	2h-3h	3h-4h	4h-5h	5h-6h	6h-7h	7h-8h	8h-9h	9h-10h
position (x2-x1)	1	1	1	1	1	-1	-1	-1	-1	-1

This equation for slope is the same with the velocity equation 2.4. So the slope values for the time intervals given in Table 1 can be used as the velocity values.

$$V_{avg} = \frac{\Delta x}{\Delta t} = \frac{x_2 - x_1}{t_2 - t_1} = slope \tag{2.7}$$

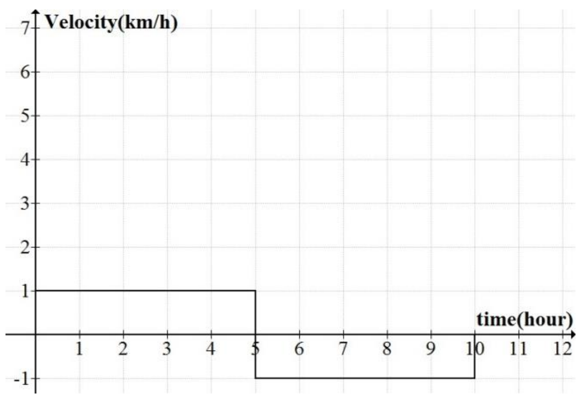


Figure 8. Velocity Graph of the Position Graph in Figure 7

This equality is also true for a position graph that has changeable slope over time. For example, the position-time graph in Figure 9 has changeable slope over time. The red lines show the slope at a particular time.

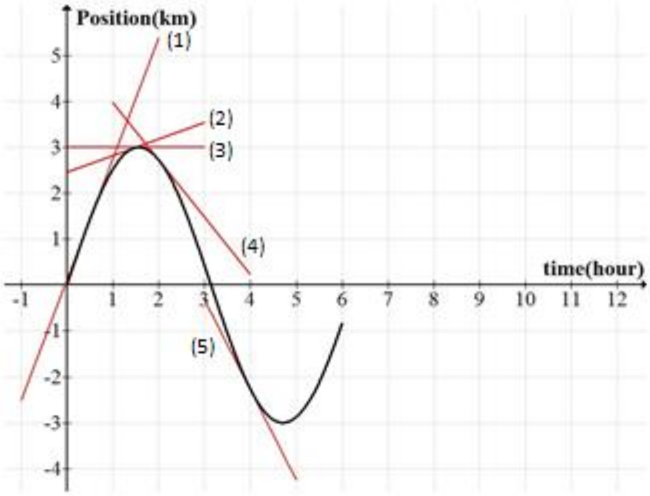


Figure 9. Position Graph with changeable slope

In the curve between the times from 0h to the first peak the slope is positive. (As seen from the lines numbered by (1) and (2)) This means that the velocity is positive and the object moves toward the positive direction. In the curve between the times from the first peak to the second peak (negative valued), the slope is negative ((4) and (5) lines) and this means that the velocity is negative and the object moves toward the negative direction. On the first peak of the graph the slope is zero (line (3)) which means that there is no displacement and the velocity is also zero so the object might be stopped or changed its direction.

If a position graph is curved, which means that the graph is not just made out of straight line parts, the slope is changing so it can be said that the velocity is not same but changeable over time. Changing in the velocity causes the acceleration. So, if an object is accelerating or decelerating, the position graph of this object will be curvy. In the position graph in Figure 9, between the times 0h and 4h there is a negative acceleration since the slope goes from positive to negative and between the times 4h and 6h there is a positive acceleration since the slope goes from negative to positive.

There will be acceleration where the object changes its velocity magnitude but also the acceleration can be obtained by only changing the direction of the movement. If there is no change in magnitude of the velocity or the direction of the movement then it cannot be any acceleration no matter how fast the object is. To be more specific, the acceleration can be defined as.

$$a = \frac{\Delta V}{\Delta t} = \frac{V_f - V_i}{\Delta t} \tag{2.8}$$

where the acceleration is a and it is equal to the difference between the final and initial velocities divided by the time duration. As it can be seen easily from the acceleration formula the acceleration unit is m/s^2.

2.2. Acceleration Sensor (Accelerometer), Gravitational Force (g-force)

Accelerometers or acceleration sensors are electronic devices that can be used to measure the acceleration values of an object. But this acceleration is not exactly the same with acceleration mentioned before. The reason of the difference between them is that even if there is not any movement or anything to make the acceleration, the accelerometer will give an acceleration value as output, which is the acceleration value of Earth. The acceleration value of the Earth on the sea level is almost $9.81\, m/s^2$ and this value is also called as “g”.

Accelerometers are commonly used in industry and in most of the electronic devices that we are using every day. In drones and in navigation systems for aircrafts and missiles very high sensitive accelerometers are used for flight stabilization. Accelerometers are used in tablet computers, smart phones and in some of the digital cameras to determine the rotation of the screen. And this accelerometer in smart phones and in tablet computers are used by most of the games and other applications in Google Play, Apple Store and other platforms to capture the tilt motion(Texas-Instruments, 2005).

Some common types of accelerometers are described as follows in Texas Instrument’s paper named “Accelerometers and how they work”:

- Capacitive
  - As the acceleration is changed, the metal beam or micromachined feature produces change in capacitance.
- Piezoelectric
  - Piezoelectric crystal mounted to mass produces voltage output. This output value is converted to acceleration.
- Piezoresistive
  - As the acceleration is changed, the resistance value of beam or micromachined feature changes.
- Hall Effect
  - Motion converted to electrical signal by sensing of changing magnetic fields
- Magnetoresistive
  - Resistivity of a material changes with magnetic field
- Heat Transfer
  - Location of heated mass tracked during acceleration by sensing temperature (Texas-Instruments, 2005)

To understand easily how the accelerometers work and what is the meaning of the obtained output data we can think a ball and two axis and their two opposite directions.

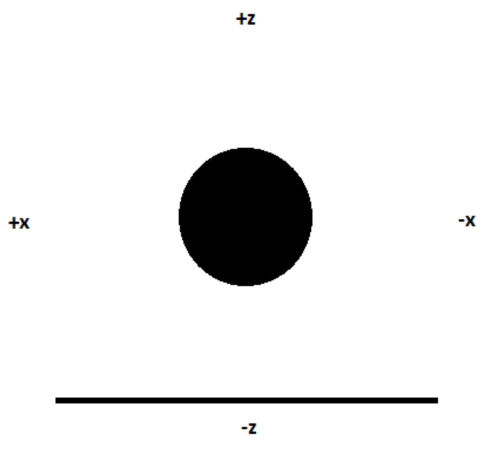


Figure 10. No Gravitational Force

In a condition that there is no gravitational force from the ground the ball will simply float in the air and there will not be any movement to any direction. So this accelerometer gives 0g on both of the two axis as output.

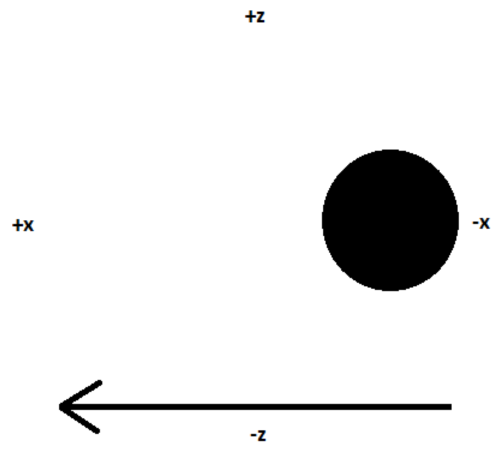


Figure 11. No Gravitational Force and the structure is moved to left direction

In the same condition that there is no gravitational force from the ground, the ball will simply float in the air again, but if the structure with 2-axis is moved suddenly to the left direction with 1g acceleration the ball will be on the –X side with the 1g acceleration. So this accelerometer's output will be -1g on the X-axis and 0g on the Z-axis.

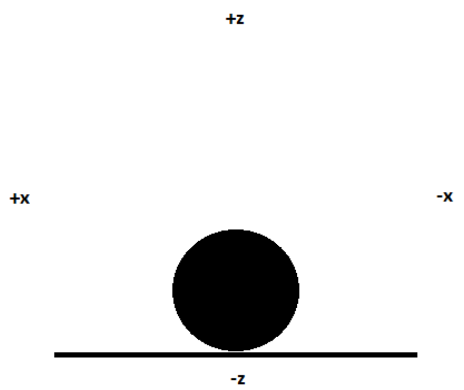


Figure 12. Gravitational Force with 1g

In a condition that there is a gravitational force from the ground like the Earth's gravitational force (1g), the ball will be on the –Z-axis. Even if there is no movement to any direction at the initial condition, this accelerometer gives -1g on the Z-axis and 0g on the X-axis as output.

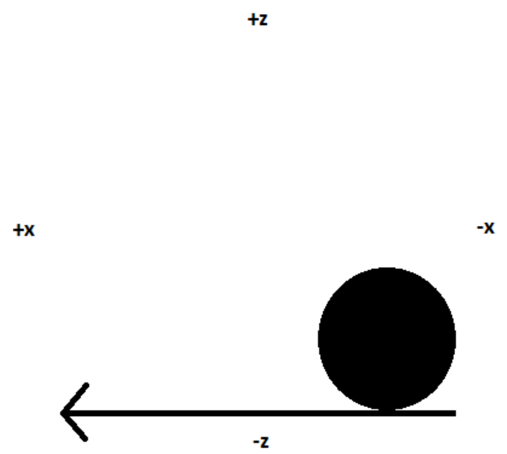


Figure 13. Gravitational Force with 1g and the structure is moved to left direction

In a condition that there is a gravitational force from the ground like the Earth's gravitational force (1g), the ball will be simply on the –Z-axis but if the 2-axis structure is moved suddenly to the left direction with 1g acceleration the ball will be on the –X side with the 1g acceleration. So this accelerometer's output will be -1g on the X-axis and -1g on the Z-axis. As an example the lower case 'a', 'b' and 'c' characters' 3 axis raw acceleration signals are shown in the figures below.

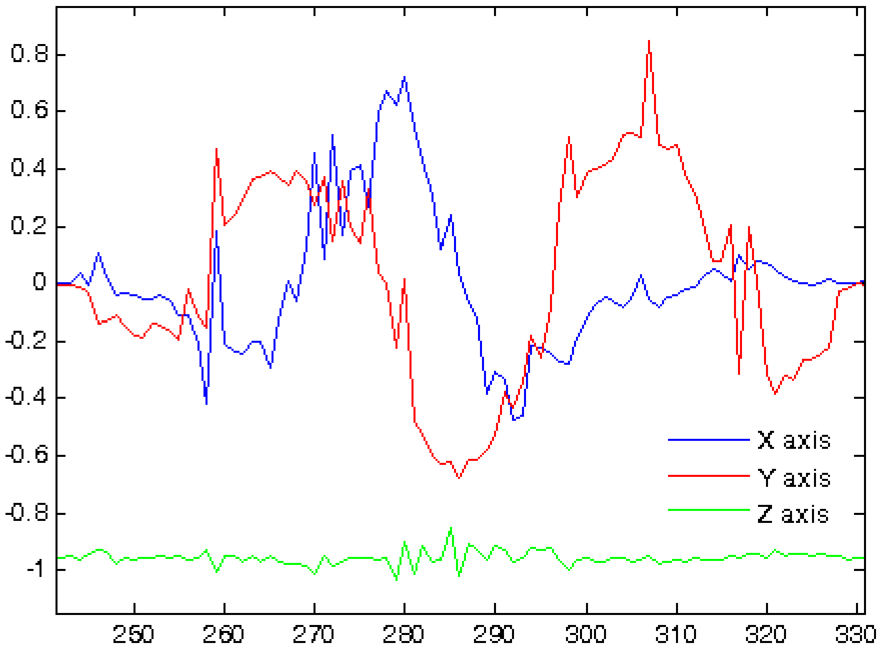


Figure 14. The raw data signals of ‘a’ character

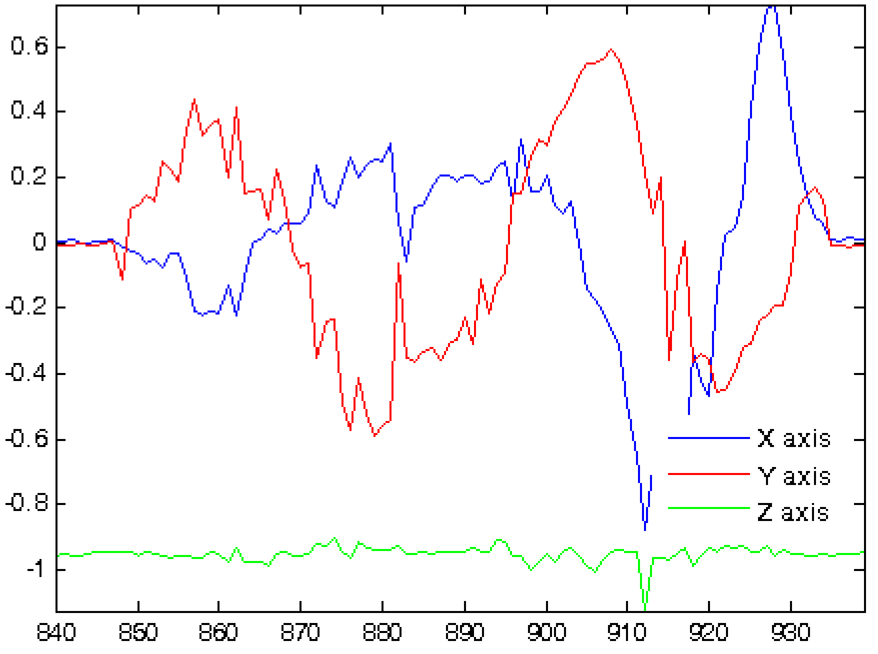


Figure 15. The raw data signals of ‘b’ character

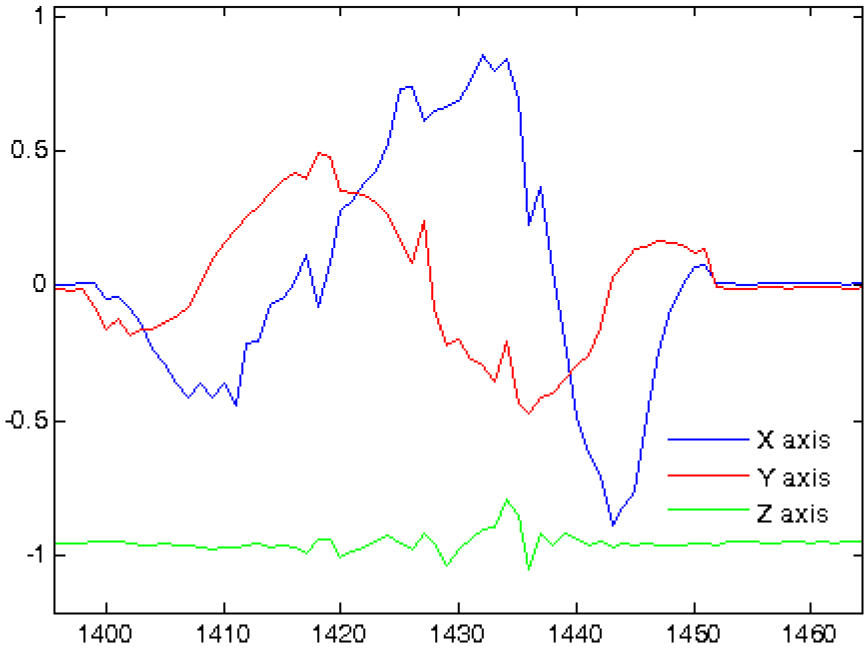


Figure 16. The raw data signals of 'c' character

As it can be seen from the all three figures above, on the Z-axis there is no any major difference during the writing process. Only there are some vibrations caused from the hand movement. The acceleration value of the Z-axis is almost -1g and this acceleration value is expected as the condition explained before.

CHAPTER 3

RECOGNITION

In the last chapter, the acceleration data acquisition is explained. The acceleration values of the writing of the each character are obtained as mentioned. Before the recognition process the signals obtained need to be preprocessed to reach the reliable results. The raw signals are filtered, separated and normalized as explained in Chapter 4. In this chapter, the algorithms used for recognition of the preprocessed signals are explained.

3.1. Linear Alignment

In this thesis, linear alignment means the basic distance calculation. In signal processing, the distance is used to find the similarity between two different data. There are many distance calculation methods like physical distance methods and theoretical distance methods, but in this thesis only the most common physical distance methods are explained and applied.

The main problem of finding the distance between two time series with linear alignment is the signal lengths. In linear alignment the lengths of the two time sequences need to be equal, but in time series it is not possible to obtain exactly the same length signals. So before finding the distance, a preprocessing step is needed to equalize the lengths of the signals in the dataset.

Finding distance between two signals with linear alignment is simply finding the distance between a sample of the first signal and corresponding sample of the other signal. To be more clear, consider s and t signals with the length of L, the distance between them is

$$distance(s,t) = \sum_{k=1}^{L} d(s_k, t_k) \tag{3.1}$$

where d is the distance between signal samples, s_k and t_k are the k^{th} samples of the s and t signals. The distance between the signals is sum of the distance between the samples.

In this thesis, three different distance metrics are used for comparison. These distances are Euclidean distance, Manhattan (Taxicab, City block) distance and Chessboard (Chebyshev, maximum metric) distance.

3.1.1. Euclidean Distance

Euclidean distance is the most common metric in distance measurements (Cassisi, 2012), although it is very weak to distortions in time axis (A. Ratanamahatana, 2004). It is simply the length of the straight line between two points. In one dimension consider the s_1 and s_2 points. The Euclidean distance between these two points is (Cassisi, 2012)

$$\begin{aligned} d_{Euclidean}(s_1, s_2) &= \sqrt{(s_1 - s_2)^2} \\ &= |s_1 - s_2| \end{aligned} \tag{3.2}$$

In two dimension, consider s and t points which are $s = s_1, s_2$ and $t = t_1, t_2$. The Euclidean distance between these two points is (Leskovec, 2014)

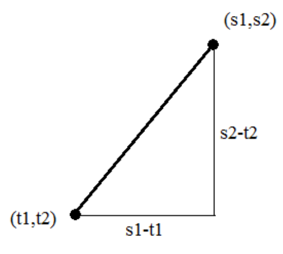


Figure 17. Distance of $\boldsymbol{s}$ and $\boldsymbol{t}$ points

$$d_{Euclidean}(s, t) = \sqrt{(s_1 - t_1)^2 + (s_2 - t_2)^2} \tag{3.3}$$

The distance between s and t signals, which have data on both X and Y-axis, each with length of L with Euclidean distance is

$$distance(s,t) = \sum_{k=1}^{L} \sqrt{\left(s_{x_k} - t_{x_k}\right)^2 + \left(s_{y_k} - t_{y_k}\right)^2} \tag{3.4}$$

where $s = (s_x, s_y)$, $s_x = (s_{x_1}, s_{x_2}, \dots, s_{x_L})$ and $s_y = (s_{y_1}, s_{y_2}, \dots, s_{y_L})$, t= (t_x, t_y), $t_x = (t_{x_1}, t_{x_2}, \dots, t_{x_L})$ and $t_y = (t_{y_1}, t_{y_2}, \dots, t_{y_L})$.

3.1.2. Manhattan Distance

The Manhattan distance is the sum of the distances of each coordinate. It is also known as City Block distance (Cassisi, 2012) or Taxicab distance ("Taxicab geometry," 2016). The Manhattan distance between s and t points in the Figure 17 is (Cassisi, 2012)

$$d_{Manhattan}(s,t) = |s_1 - t_1| + |s_2 - t_2| \tag{3.5}$$

The distance between s and t signals, which have data on both X and Y-axis, each with length of L with Manhattan distance is

$$distance(s,t) = \sum_{k=1}^{L} \left|s_{x_k} - t_{x_k}\right| + \left|s_{y_k} - t_{y_k}\right| \tag{3.6}$$

where $s = (s_x, s_y)$, $s_x = (s_{x_1}, s_{x_2}, \dots, s_{x_L})$ and $s_y = (s_{y_1}, s_{y_2}, \dots, s_{y_L})$, t= (t_x, t_y), $t_x = (t_{x_1}, t_{x_2}, \dots, t_{x_L})$ and $t_y = (t_{y_1}, t_{y_2}, \dots, t_{y_L})$.

3.1.3. Chessboard Distance

The Chessboard distance is the greatest distance value of the each coordinate distance values. It is also known as Chebyshev distance or maximum metric ("Chebyshev distance," 2016). The Chessboard distance between s and t points in the Figure 17 is ("Distance based models ", 2013)

$$d_{Chessboard}(s,t) = \max(\,|s_1 - t_1|\,, |s_2 - t_2|\,) \tag{3.7}$$

The distance between s and t signals, which have data on both X and Y-axis, each with length of L with Chessboard distance is

$$distance(s,t) = \sum_{k=1}^{L} \max(\,\left|s_{x_k} - t_{x_k}\right|\,, \left|s_{y_k} - t_{y_k}\right|\,) \tag{3.8}$$

where $s = (s_x, s_y)$, $s_x = (s_{x_1}, s_{x_2}, \dots, s_{x_L})$ and $s_y = (s_{y_1}, s_{y_2}, \dots, s_{y_L})$, t= (t_x, t_y), $t_x = (t_{x_1}, t_{x_2}, \dots, t_{x_L})$ and $t_y = (t_{y_1}, t_{y_2}, \dots, t_{y_L})$.

3.2. Nonlinear Alignment - Dynamic Time Warping (DTW)

Dynamic Time Warping is a method that used for finding the similarity between two sequences. In time series signals, there is a big problem about time lengths of the signals. Generally, it is not possible to have two signals that have exactly the same length. If we consider the acceleration signals in our case, even if we can have exactly the same length signals by adjusting the signal sample number, it is not possible to make the same movement part at the same time interval with the other repetition movement. It can be seen from the Figure 18.

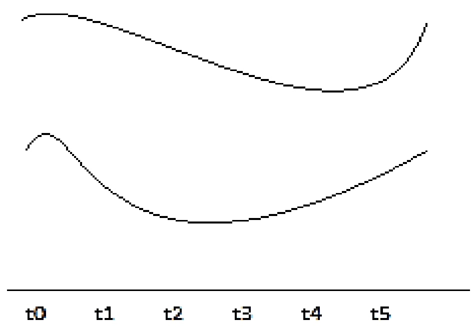


Figure 18. Repetitions of the same movement

The two signals shown in the Figure 18 is considered as acceleration values of movement of the same character. They have almost same overall shape but their minimum and maximum valued points are not on the same time interval. The minimum value of the signal at the top is on the $t4 - t5$ time interval, but the below one's minimum value is on the $t2 - t3$ time interval. It can be said that the below signal's writing is done faster than the above signal or it can be said that that part of the writing is done before the above one.

In linear similarity calculation, the distance measurement is performed between only for the same data points of the two signals as explained in the previous part. But if we try to find the similarity between the signals, which have different accelerations and decelerations in it, with linear alignment as shown in the Figure 19 (left) the linear similarity results will be wrong. The matching on the left side in Figure 19 shows the linear alignment and the right one shows the accurate matching with nonlinear alignment.

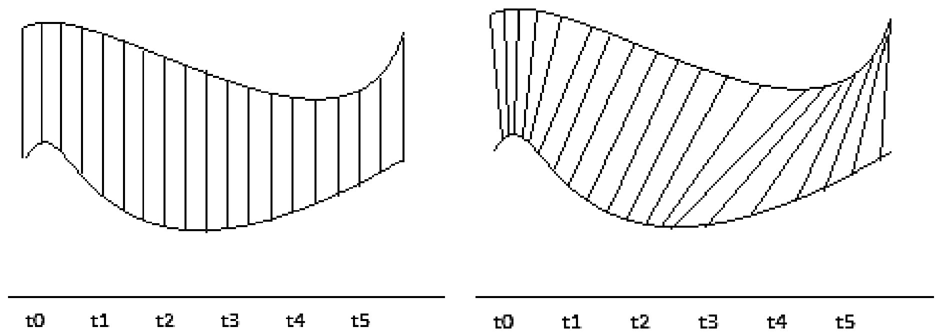


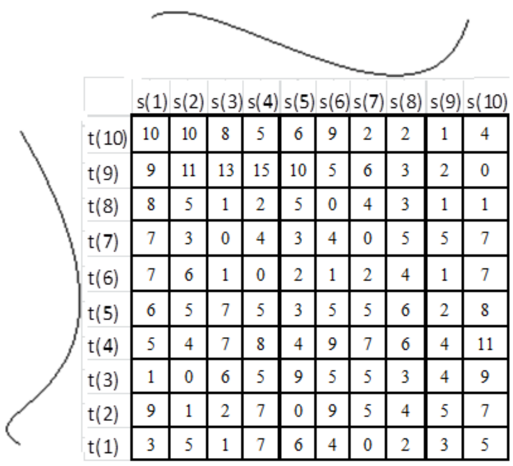
Figure 19. Linear Alignment and Nonlinear Alignment of the signals in Figure 18

The nonlinear matching as shown in the right one in Figure 19 can be implemented with Dynamic Time Warping. DTW is a method that is based on the distance measurements between two data points. The distance formula used in DTW is an Euclidean distance as modified version of (3.2) like in (3.9) (Selina Chu, 2002). The main difference between the distance calculations in DTW and linear alignment is that a data point of the first signal is compared with the all data points of the other signal in

DTW. With such a comparison, a distance matrix is obtained. The aim of DTW is finding the optimal matching by using the distance matrix values.

$$Dist\_matrix(i,j) = \sqrt{(s_i - t_j)^2} \tag{3.9}$$

Let's consider *s* and *t* signals in Figure 18 and create the distance matrix as shown in Figure 20 as an example. To compose the distance matrix of the two signals having only one axis, the algorithm shown in Figure 21 is applied.



	s(1)	s(2)	s(3)	s(4)	s(5)	s(6)	s(7)	s(8)	s(9)	s(10)
t(10)	10	10	8	5	6	9	2	2	1	4
t(9)	9	11	13	15	10	5	6	3	2	0
t(8)	8	5	1	2	5	0	4	3	1	1
t(7)	7	3	0	4	3	4	0	5	5	7
t(6)	7	6	1	0	2	1	2	4	1	7
t(5)	6	5	7	5	3	5	5	6	2	8
t(4)	5	4	7	8	4	9	7	6	4	11
t(3)	1	0	6	5	9	5	5	3	4	9
t(2)	9	1	2	7	0	9	5	4	5	7
t(1)	3	5	1	7	6	4	0	2	3	5

Figure 20. An example of distance matrix

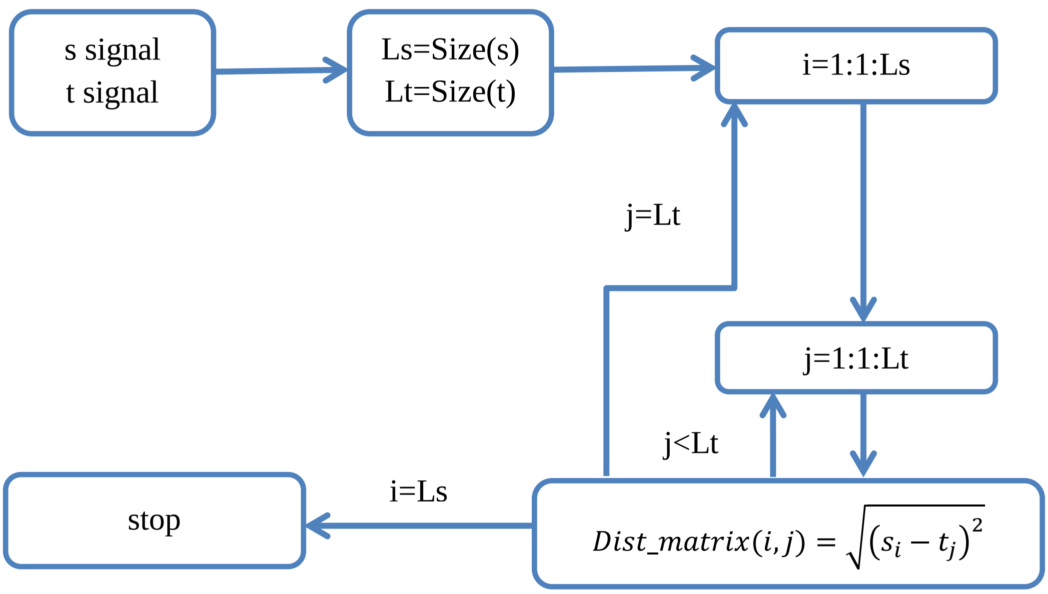


Figure 21. Flowchart for creating the Distance matrix

In this thesis, we use 3-axis accelerometer to obtain the dynamics of writing for 26 characters of the English alphabet. So for a single character repetition there are 3 different data in X-, Y- and Z-axis. But there is not any valuable knowledge in Z-axis, since the writing process is done on the surface parallel to the ground. Only X- and Y-axis are used. The distance equation given in the (3.2) is not suitable for the distance calculation, since we need to use an equation for two dimensions. The equation (3.3) can be modified as

$$Dist\_matrix(i,j) = \sqrt{\left(s_{x_i} - t_{x_j}\right)^2 + \left(s_{y_i} - t_{y_j}\right)^2} \tag{3.10}$$

To find the distance matrix between two signals the Euclidean distance equation explained above is used in classical DTW. But the computing duration of Euclidean distance is so long because of the square and square root operations. So we try to see if other basic distance measurements are faster than the Euclidean distance measurement. This part of our thesis is different than the other studies in the literature about accelerometer based character recognition. We use other two physical distance measurement techniques: **Manhattan** and **Chessboard** distances. The detailed information about them can be found in 3.1.2 and 3.2.2. The Manhattan distance to be used in the distance matrix can be calculated by the following equation.

$$Dist\_matrix(i,j) = \left|s_{x_i} - t_{x_j}\right| + \left|s_{y_i} - t_{y_j}\right| \tag{3.11}$$

And the Chessboard distance matrix can be obtained by the following equation.

$$Dist_{matrix(i,j)} = \max\left(\left|s_{x_i} - t_{x_j}\right|, \left|s_{y_i} - t_{y_j}\right|\right) \tag{3.11}$$

The comparison of performance of the all three methods is given in the section 4.3.

The main aim of DTW is to find the optimal warping path that corresponds to the best matching in the distance matrix. Without any restriction thousands of possible warping paths can be obtained and also the matching path found may contain mismatching or re-matching of the signal parts. Consider the same s and t signals and their warping path W with k points of length. The restrictions are explained in the following (Eamonn J. Keogh, 2001; Qian Chen, 2012).

$$W = w_1, w_2, \dots, w_k, \dots, w_K \qquad \max(L_t, L_s) \leq K < L_s + L_t - 1$$

<u>*Boundary Conditions:*</u> The first and the last points of a warping path need to be at the opposite diagonal corner points of the distance matrix.

$$w_1 = (1,1) \quad \text{then} \quad w_K = (L_t, L_s)$$

<u>*Continuity:*</u> For the next point of the warping path, there cannot be a jumping more than one cell. It is also same for the diagonal adjacent points.

$$w_{k+1} = (a', b'), \quad w_k = (a, b) \quad \text{then} \quad a' - a \leq 1\,, \quad b' - b \leq 1$$

<u>*Monotonicity:*</u> The warping path needs to go to the opposite corner monotonically. This restriction prevents the re-matching problem to occur.

$$w_{k+1} = (a', b'), \quad w_k = (a, b) \quad \text{then} \quad \mathrm{a}' \geq \mathrm{a}, \quad \mathrm{b}' \geq \mathrm{b}$$

Even the restrictions mentioned above are not enough to find the best matched warping path. After restrictions there are so many different warping paths and some of them are shown in Figure 22. To decide the best matching warping path the number of the possible warping paths need to be reduced. At this point, we need to use a concept which is called as warping cost. The warping cost is cumulative distance value of each path. To choose the optimal one the minimum warping costly path should be chosen. Let's consider n different warping path obtained and the minimum cost can be calculated by the following.

$$DTW = min\left(\sum_{k=1}^{K} W_{i_k}\right) \qquad 1 \leq i \leq n \tag{3.11}$$

With the dynamic programming, the optimal path that minimizes the warping cost can be found (Ahmad Akl 2010).

$$D(i,j) = Dist\_matrix(i,j) + min\begin{pmatrix} D(i,j-1) \\ D(i-1,j) \\ D(i-1,j-1) \end{pmatrix} \quad (3.12)$$

In Figure 22, the possible warping paths can be seen. By using the equation above the optimal path is found. The purple line is the optimum path that minimizes the warping cost.

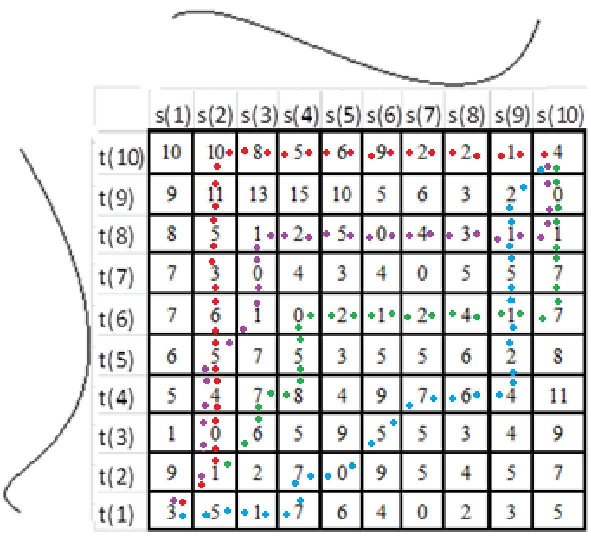


Figure 22. Some of the possible warping paths

We found an optimal path with the equation 3.12, but this path may not be the real optimal path that shows the true matching. The warping path is needed to be around the diagonal line of the distance matrix (Qian Chen, 2012). If the path is tend to be far from this line then some unwanted mismatching can be occur. If the path moves away from the diagonal line this means that the amplitude values of the different parts of the two signals are nearer than the parts of these signals that expected to be matched. To prevent this kind of situation warping windows are used. The warping windows also speed up the DTW algorithm since the distance measurement is only done for the cells inside of the warping window. The length of the warping window can be chosen any number between the length differences of the two signals to all matrix size.

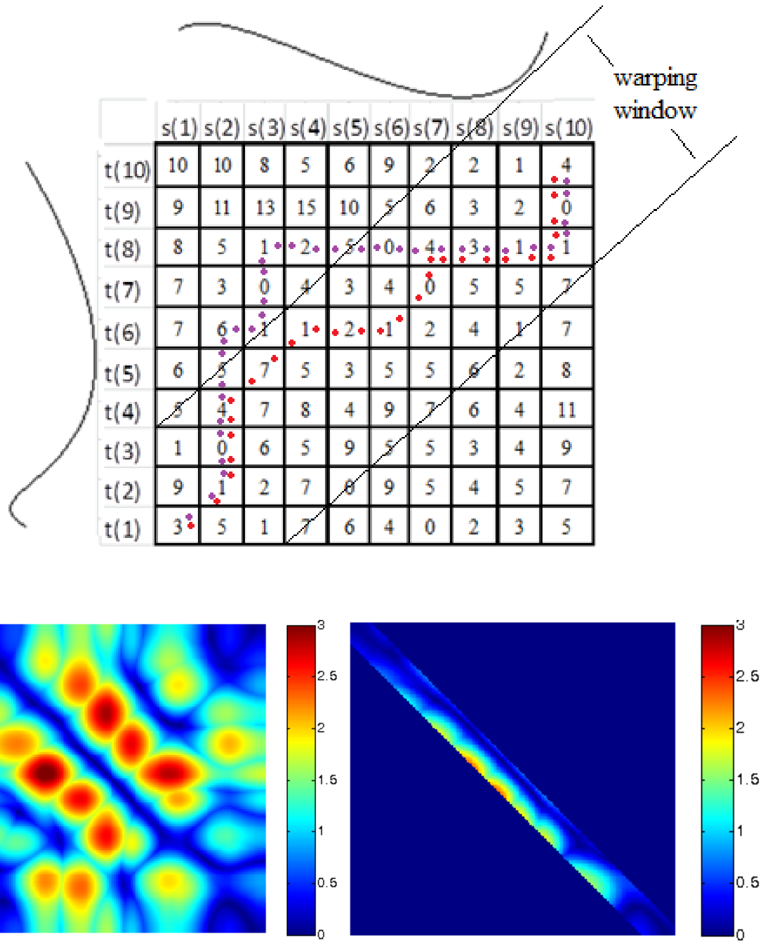


Figure 23. Warping path restricted by a warping window

3.2.1. Singularity problem and DDTW

Time series signals can be differ in time axis or in X axis as mentioned in the previous parts. To solve this situation we use DTW algorithm which is the best method for matching this kind of signals. But time series signals can also be differ in the amplitude or in Y axis. DTW use the amplitude values of the signals to match them. If the signal parts that needed to be matched have different amplitude values, then DTW cannot be successful in here. In this kind of matching, the parts that are not related to each other can be matched just because their amplitude values are nearer or a wide area of a signal can be matched with a point of the other signal. This problem in DTW is named as *singularities* (Culp, 2008).

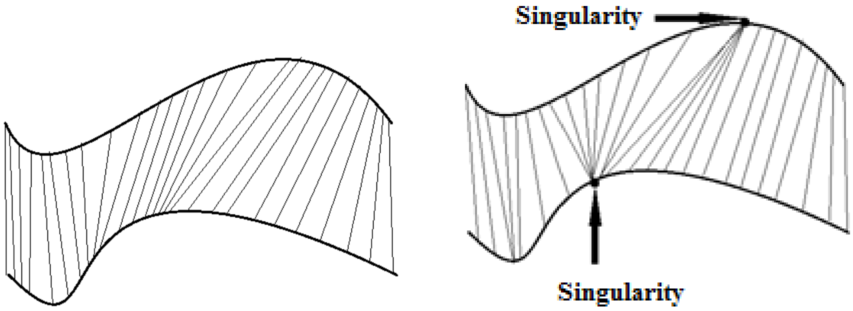


Figure 24. True matching and matching with singularities

(Eamonn J. Keogh, 2001) et al. represent a method to solve this singularities problem. The method they found is named as Derivative Dynamic Time Warping (DDTW). In this method, instead the amplitude of the raw data they use the shape of the signals. By using shape information of the signals, the effect of the difference in the amplitude values between the parts of the signals that need to be matched disappear. To obtain the shape information of the signals they take the first derivative of each signal in the database with the following equation.

$$der[q] = \frac{(q_i - q_{i-1}) + ((q_{i+1} - q_{i-1})/2)}{2} \tag{3.13}$$

The equation above takes the first derivative by taking the average of the difference of the point in process and its left neighbor, and the difference of the same point's left and right neighbors. But for our acceleration signals there is no recognition rate difference between using the equation above and the basic derivative equation. The basic first derivative equation that we use is:

$$der[q] = \frac{(q_i - q_{i-1})}{1} \tag{3.14}$$

which is the difference between the point in process and its left neighbor. And also the normalization process is not applied to the raw data or derivative data. The data used for derivation are filtered and segmented. The normalization is not used in this part since normalization reduces the amplitude difference between signals so we can see the real effectiveness of the derivation.

Our thesis is the first study that uses the derivative dynamic time warping on character recognition. Actually, there is not any study that uses this method with 26 complex gestures.

3.2.2. Comparison of Other Alternative Methods to Increase the Performance

Quantization of Acceleration Data:

(Jiayang Liu, 2009) et al. add a preprocessing step to speed up the DTW algorithm. They quantize the raw acceleration data with the values given in the table below. The acceleration data in their database are between -2g to +2g but very few part of their acceleration values are around +2g and -2g so they use nonlinear quantization. But in our acceleration database, the data are normalized between -1g to +1g in preprocessing part so we change the values of the table below as the values in the next table. Quantization converts the acceleration values with floating point to integer values so it reduces the floating point calculations.

Table 2. Quantization Values
(Source: (Jiayang Liu, 2009))

Acceleration Data	Converted value	Acceleration Data	Converted value
$a > 2g$	16	$-g < a < 0$	-1 to -10
$g < a < 2g$	11 to 15	$-2g < a < -g$	-11 to -15
$0 < a < g$	1 to 10	$a < -2g$	-16
$a=0$	0		

Table 3. Quantization values that used in this thesis

Acceleration Data	Converted value
0g < a<+1g	1 to 8
a=0g	0
-1g<a<0g	-1 to -8

<u>Correlation Coefficient:</u>

Correlation is a method used for measuring the strength and the direction of similarity between two signals (Davide Figo, 2010). Since the writing is on 2-axis, before using the correlation we combine the data of 2-axis by transforming them into the complex number. Covariance is not a useful measure, since the value of it depends on the scales of the measurement. The rescaled version of covariance is correlation coefficient (Joseph Lee Rodgers, 1988), as in following equations.

$$cov(s,t) = \frac{1}{n-1}\sum_{i=1}^{n}(s_i-\mu_s)^*(t_i-\mu_t) \tag{3.15}$$

$$C = \begin{pmatrix} cov(s,s) & cov(s,t) \\ cov(t,s) & cov(t,t) \end{pmatrix} \tag{3.16}$$

$$\rho(s,t) = \frac{1}{n-1}\sum_{i=1}^{n}\frac{(s_i-\mu_s)^*}{\sigma_s}\frac{(t_i-\mu_t)}{\sigma_t} \tag{3.17}$$

$$\rho(s,t) = \frac{cov(s,t)}{\sigma_s\sigma_t} \tag{3.18}$$

$$corr\_coef = \begin{pmatrix} \dfrac{cov(s,s)}{\sigma_s\sigma_s} & \dfrac{cov(s,t)}{\sigma_s\sigma_t} \\ \dfrac{cov(t,s)}{\sigma_t\sigma_s} & \dfrac{cov(t,t)}{\sigma_t\sigma_t} \end{pmatrix} \tag{3.19}$$

$$corr\_coef = \begin{pmatrix} \rho(s,s) & \rho(s,t) \\ \rho(t,s) & \rho(t,t) \end{pmatrix} \tag{3.20}$$

where *s* and *t* are the two signals with the length of n. μ_s and μ_t are the mean values of *s* and *t*, σ_s and σ_t are standard deviations, and $\rho(s,t)$ is the correlation coefficient of *s* and *t* signals. By dividing the covariance with standard deviations, the range of covariance is rescaled from -1 to +1. +1 means the two signals are completely correlated and -1 means the two signals are completely negative correlated.

Since we use complex versions of two axis data to find their correlation coefficients, the values of the equation 3.20 is also complex numbers. For comparison we use the magnitude values of the complex numbers.

CHAPTER 4

EXPERIMENTAL WORK AND RESULTS

4.1. Hardware

To obtain the acceleration values of the characters, a ring-like device containing an accelerometer and an associated circuit is used. This circuit contains a 3-axis accelerometer, a microprocessor and an SD card module. The overall circuit structure is presented in the figure below.

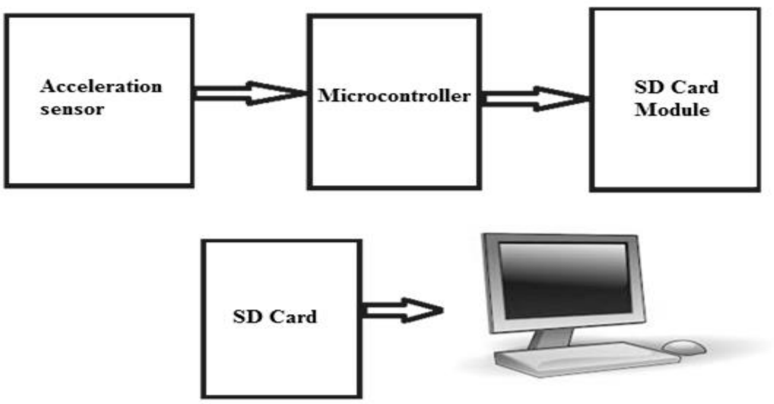


Figure 25. Overall Structure of the Hardware

As it can be seen from the Figure 26, the circuit is a ring type device that users need to wear the ring part of the circuit which contains the 3 axis accelerometer on their middle finger. But the wearing direction of the ring is important here. The user shouldn't change this direction.

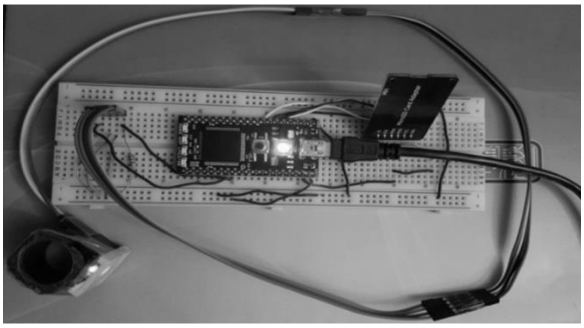


Figure 26. Data Acquisition Device

4.1.1. Accelerometer

MPU-6050 combines a 3-axis gyroscope and a 3-axis accelerometer. It is a small package with the size of 4x4x0.9mm. It can interface with other digital sensors like pressure sensor with its I2C port. It has three 16-bit analog-to-digital converters (ADCs) for digitizing the gyroscope outputs and three 16-bit ADCs for digitizing the accelerometer outputs. Users can choose the gyroscope range of ±250, ±500, ±1000 and ±2000°/sec (dps) and accelerometer range of ±2g, ±4g, ±8g and ±16g ("MPU-6000 and MPU-6050 Product Specification Revision 3.4," 2013).

The 3-axis accelerometer in MPU-6050 has so many features as in listed below:

- It has 3-axis and gives digital output with the programmable range of ±2g, ±4g, ±8g and ±16g
- It has integrated 16-bit ADC for sampling the analog acceleration.
- Accelerometer normal operating current: 500µA
- Low power accelerometer mode current: 10µA at 1.25Hz, 20µA at 5Hz, 60µA at 20Hz, 110µA at 40Hz
- Tap detection
- User-programmable interrupts
- High-G interrupt
- User self-test

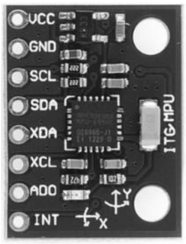


Figure 27. MPU-6050

4.1.2. Microcontroller

mbed NXP LPC1768 prototyping board is used to microcontroller part of the circuit. It has so many important features like consuming ultra-low power. This board is very effective on mobile use because of its small size and high speed computing ability. It gives 3.3V and 5V as an output voltage that can be used as a source voltage for other device or sensor. It is easy to use with Web-based C/C++ programming environment. The mbed NXP LPC1768 also offers other features not found on lower-performance prototyping boards, such as Ethernet, USB OTG, a 12-bit ADC, a 10-bit DAC for a true analog voltage output, in addition to more common interfaces like serial (UART), SPI, I2C, and CAN("ARM mbed NXP LPC1768 Development Board," 2016).

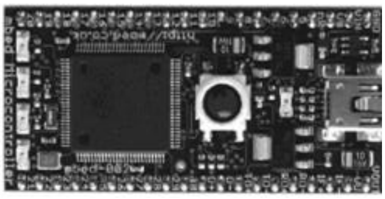


Figure 28. ARM mbed NXP LPC1768 Development Board

4.1.3. Micro SD Card Module

MPU-6050 has 3-axis accelerometer in it and it gives all of these acceleration values as output to microcontroller. mbed NXP LPC1768 microcontroller sends the acceleration values to the Micro SD Card Module and in this module all 3 axis values are saved separately to the Micro SD Card. Also the gyroscope data are sent to this module and they are saved separately as well. But in this thesis we are not using the gyroscope data we only use the acceleration data.

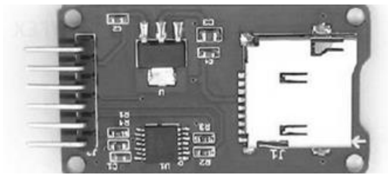


Figure 29. Micro SD Card Module

4.2. Preprocessing

Almost in every study with signals, before recognition part, preprocessing steps need to be applied to the raw data. In this project, the character data obtained from accelerometer also need preprocessing step.

There are three sub steps of the preprocessing. The first and maybe the most important step is choosing the most suitable filter for the data used and filtering them. After obtaining the smoother signals, we should choose and separate the important parts of the signals from the parts that there is no knowledge. And if we consider it for the acceleration signals, these important signal parts are the parts that there is a movement. At the end of the segmentation step, each separated signals need to be normalized to make them more likely with each other. So the amplitude range of the different repetitions of the same movement will be equal and also the maximum and minimum values of each signal will be equal.

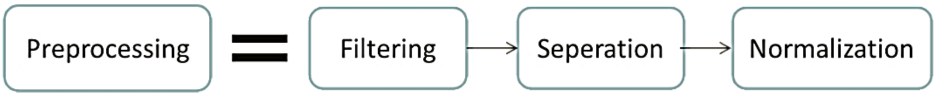


Figure 30. Simple Block Diagram of Preprocessing

The accelerometer used in this project is a 3-axis accelerometer. At the end of every data collection process the accelerometer give 3-axis data, separately. These axis data are perpendicular to each other and all of these perpendicular data saved in different text files. For every character writing the right and left directed accelerations of movement saved in X-axis named text file, the up and down directed accelerations of movement saved in Y-axis named text file and in and out from the writing surface directed acceleration of movement saved in Z-axis named text file. Before recognition step all of these different named text file data need to be preprocessed separately.

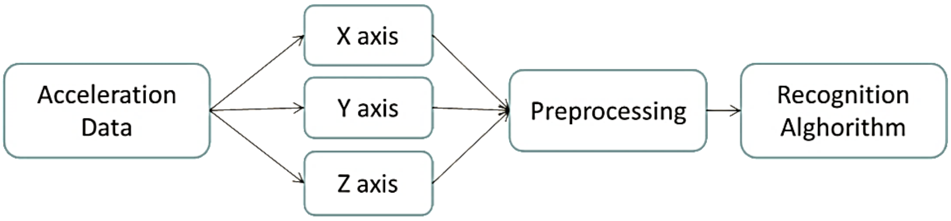


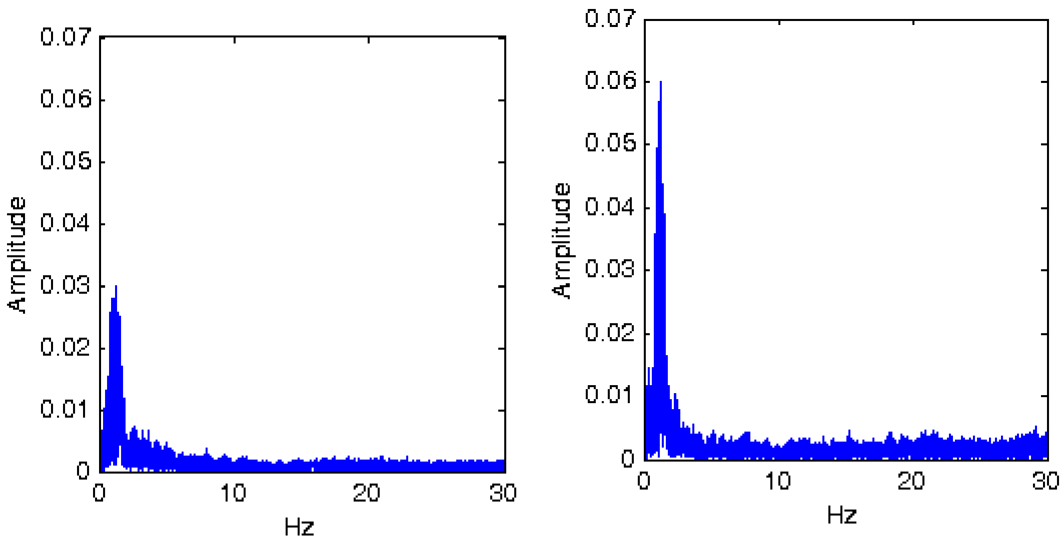
Figure 31. Block Diagram of Overall System

4.2.1. Filtering

Like most of the other sensors used for various data acquisition, accelerometers or acceleration sensors have inner noise. Even in the stable condition like if there is no any movement or physical vibration on accelerometer, still there will be undesired noise on the acceleration data. And before using, these data need to be filtered. Actually in every signal recognition study the signals need to be filtered to remove this undesired noise.

The main problem in using the accelerometer is the inner noise and the noise caused from the hand vibration during the writing process. To remove the noise and to obtain the data that only consist of the acceleration of movement, a proper filter should be chosen. To remove the noise, firstly the frequency of the noise and the real signal should be known. A common use of Fourier transforms is to find the frequency components of a signal buried in a noisy time domain signal (MATLAB2008-Help, 2008). The Matlab function ‘fft’ is used to obtain the frequency spectrum of the signal and noise. “Y = fft(x)” returns the Discrete Fourier Transform (DFT) of vector x, computed with a Fast Fourier transform (FFT) algorithm.

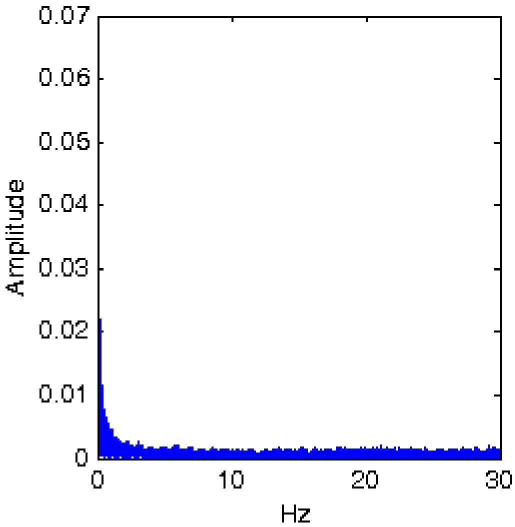
The Figure 32 shows the frequency components of the lower case ‘a’ character signal and accompanying noise. The left one belongs to the X-axis of the writing acceleration signal and the right one belongs to the Y-axis of the writing acceleration signal, respectively. The third one belongs to the Z-axis of the writing, but in the writing process there is no movement in the Z-axis because the writing is done on a surface parallel to the ground, so there is no any considerable acceleration or any frequency that belongs to the movement. Because of this reason only the X-axis and Y-axis data will be used to decide the filter type.



(a) (b)

Figure 32. Single-Sided Amplitude Spectrum of acceleration signal of lower case ‘a’characters (a) X axis (b) Y axis (c) Z axis

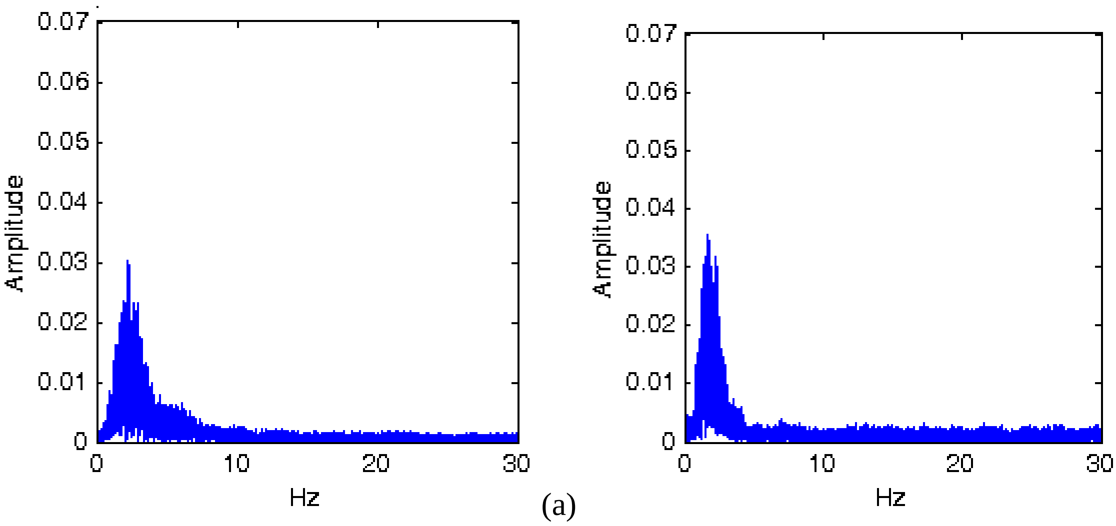
(cont. on next page)



(c)

Figure 32. (cont.)

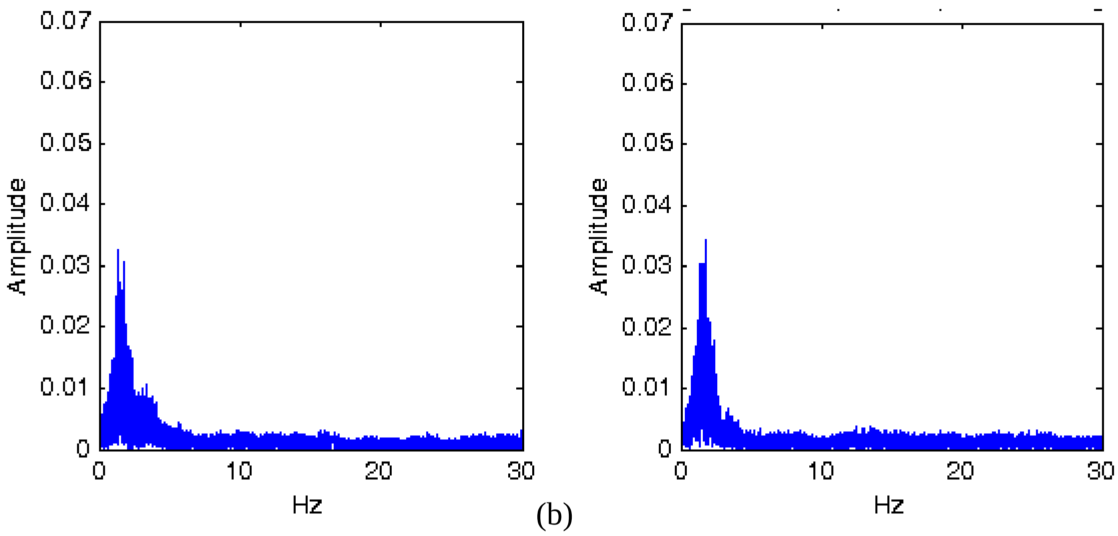
From the Figure 32 (a) and (b), it can be seen that the signal is in low frequencies and the noise is in the high frequencies. But analyzing the signal of only one character and using the frequency range of this character may not be enough for deciding about frequencies and filtering to remove the noise from the signal. We need to find not only the frequency of one character but also frequency of writing because the filter will be applied to all character signals. So we need to analyze the frequencies of different characters. Some other characters frequency components are shown in the Figure 33.



(a)

Figure 33. Single-Sided Amplitude Spectrum of acceleration signal of (a) lower case ‘b’ character (X axis, Y axis) (b) lower case ‘c’ character (X axis, Y axis)

(cont. on next page)



(b)

Figure 33. (cont.)

From the frequency component graphs, given in the Figure 33, it can be said that generally the accelerometer has high frequency inner noise and the measured acceleration of movement is in low frequency. To remove this high frequency noise a low pass filter is needed. In the literature, the most used filter to remove the high frequency noise in acceleration signal is Moving Average Filter (MAF).

The MAF is one of the most common filters used in Digital Signal Processing (DSP) because it is easy to understand and it is also easy to apply it to the discrete signals. This filter simply takes the average of the specified number of data points. The equation (4.1) shows the MAF in the equation form.

$$y[n] = \frac{1}{1 + M_1 + M_2} \sum_{k=-M_1}^{M_2} x[n-k] \tag{4.1}$$

$$y[n] = \frac{1}{1 + M_1 + M_2} (x[n + M_1] + x[n + M_1 - 1] + \cdots + x[n] + x[n-1] + \cdots + x[n - M_2]) \tag{4.2}$$

where $x[]$ is the input signal, $y[]$ is the output signal, n is the data point of the signal and M_1 and M_2 determine the window size. This equation simply computes the n^{th} sample of the output as the average of samples of the input sequence around the n^{th} sample (OPPENHEIM, 1998). As an example if we try to filter the 5^{th} point of the signal and if we choose $M_1 = 1$ and $M_2 = 3$ unsymmetrically the equation above will be

$$y[5] = \frac{1}{1+1+3}(x[5+1] + x[5+1-1] + x[5-1] + x[5-2]$$
$$+x[5-3])$$

(4.3)

$$y[5] \ = \frac{1}{1+1+3}(x[6] + x[5] + x[4] + x[3] + x[2])$$

If we choose the window border symmetrically like $M_1 = 2$ and $M_2 = 2$ the equation will be

$$y[5] = \frac{1}{1+2+2}(x[5+2] + x[5+2-1] + x[5+2-2] + x[5-1]$$
$$+x[5-2])$$

(4.4)

$$y[5] = \frac{1}{1+2+2}(x[7] + x[6] + x[5] + x[4] + x[3])$$

By averaging the data points symmetrically or unsymmetrically, this filter remove the sharp edges of the signal, so it is a smoothing filter and also by smoothing, the high frequency noise will be removed. The impulse response of the moving average filter is

$$h[n] = \begin{Bmatrix} \frac{1}{M_1 + M_2 + 1}, & -M_1 \leq n \leq M_2, \\ 0, & otherwise \end{Bmatrix} \quad (4.5)$$

And the frequency response of the moving average filter is

$$H\left(e^{jw}\right) = \frac{1}{M_1 + M_2 + 1} \sum_{n=-M_1}^{M_2} e^{-jwn} \quad (4.6)$$

The equation below can be used to express the sum in closed form

$$\sum_{k=N_1}^{N_2} a^k = \frac{a^{N_1} - a^{N_2+1}}{1-a}, \qquad N_2 \geq N_1 \tag{4.7}$$

The frequency response of the moving average filter can be written in closed form, using the equation (4.7) above, as

$$H(e^{jw}) = \frac{1}{M_1 + M_2 + 1} \frac{e^{jwM_1} - e^{-jw(M_2+1)}}{1 - e^{-jw}} \tag{4.8}$$

The frequency response of the filter can be seen in Figure 34.

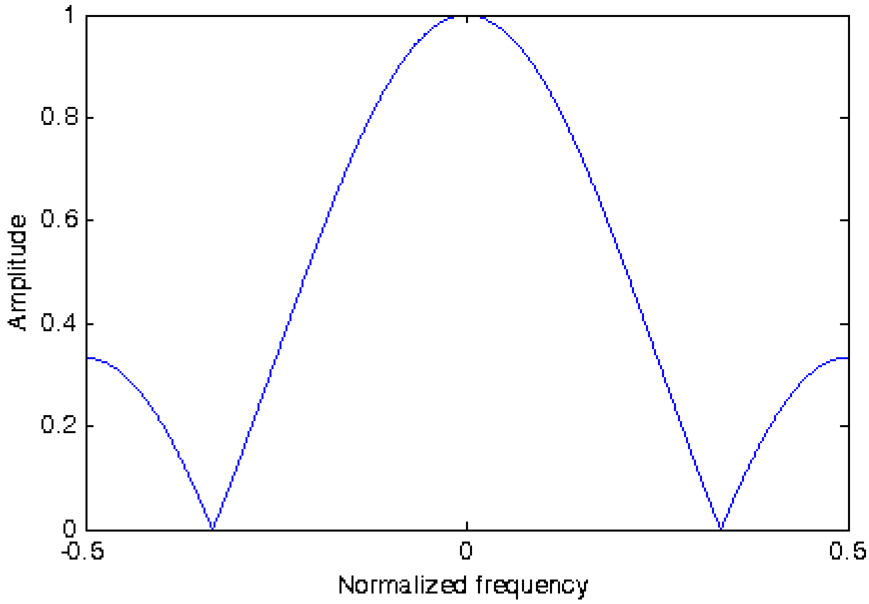


Figure 34. Frequency response of the MAF

When using the moving average filter the main problem is choosing the optimal window size. The window size determines that how many data points will be used for averaging and the points will be chosen symmetrically or unsymmetrically. If we want to compare the effects of the window size we can look at the frequency responses of the filters that have different window sizes. The Figure 35 shows the cutoff frequency difference between different window sized filters. $(M_1 + M_2 + 1 = L = window\ size)$

If the window size is chosen to have more data points, the cut off frequency of the low pass filter will be lower and the filter will suppress the frequency parts of the signals that wanted to be removed (the noisy part) much more. For the acceleration signals used in this thesis the window length is chosen to have 15 data points (red line in figure below) since the writing frequency is roughly between 0 Hz and 4 Hz for all of the samples as it can be seen from Figure 32 and 33.

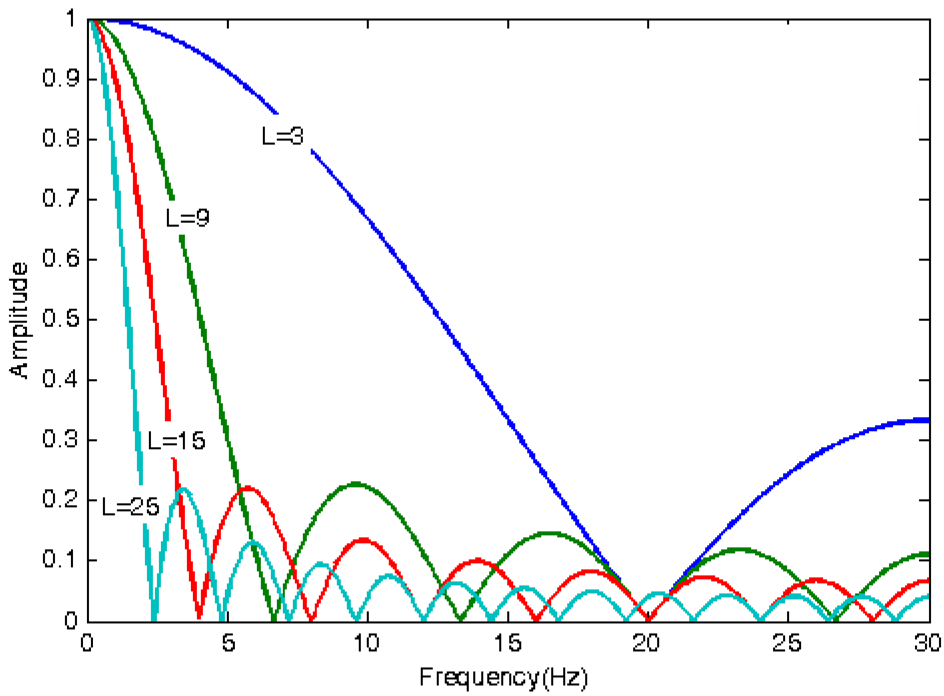


Figure 35. Comparing the frequency responses of the MAF with different window sizes

The filter can be applied to the signals with convolution. As an example if we choose the window size as 5, averaging for the 5th sample will be

$$y[5] = \frac{1}{5}(x[7] + x[6] + x[5] + x[4] + x[3])$$

$$y[5] = \frac{1}{5}(1 \times x[7] + 1 \times x[6] + 1 \times x[5] + 1 \times x[4] + 1 \times x[3])$$

$$y[5] = (\frac{1}{5} \times x[7] + \frac{1}{5} \times x[6] + \frac{1}{5} \times x[5] + \frac{1}{5} \times x[4] + \frac{1}{5} \times x[3]) \quad (4.9)$$

$$y[5] = \begin{bmatrix} \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \end{bmatrix} * [\ldots \quad x[7] \quad x[6] \quad x[5] \quad x[4] \quad x[3] \quad \ldots]$$

The "∗" symbol used above represents the convolution. The last equation in (4.9) shows how the convolution can be done. The first element of the convolution is the filter that all elements of it are divided by its window size and the second element of the convolution is the signal that wanted to be filtered. Convolution is a commutative operation so the equation below gives the same result as equation above.

$$y[5] = [\dots \quad x[7] \quad x[6] \quad x[5] \quad x[4] \quad x[3] \quad \dots] * \begin{bmatrix} \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} & \frac{1}{5} \end{bmatrix} \tag{4.10}$$

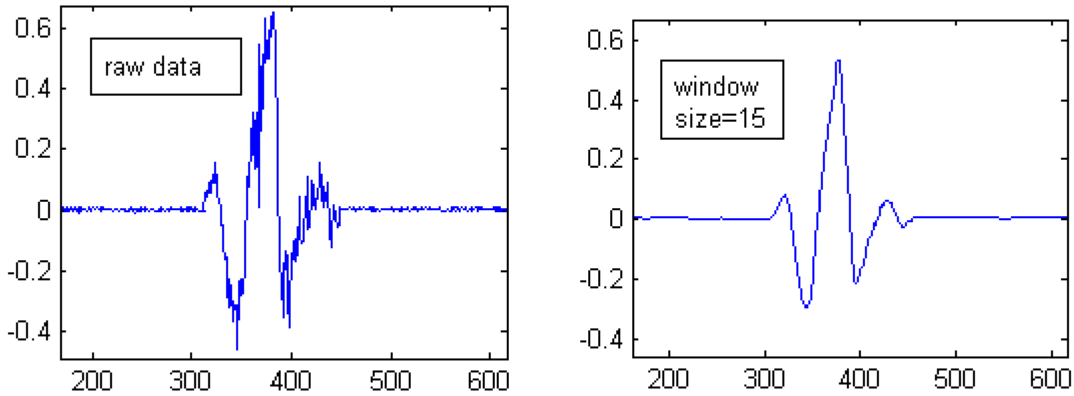


Figure 36. Raw data and filtered data (L=15)

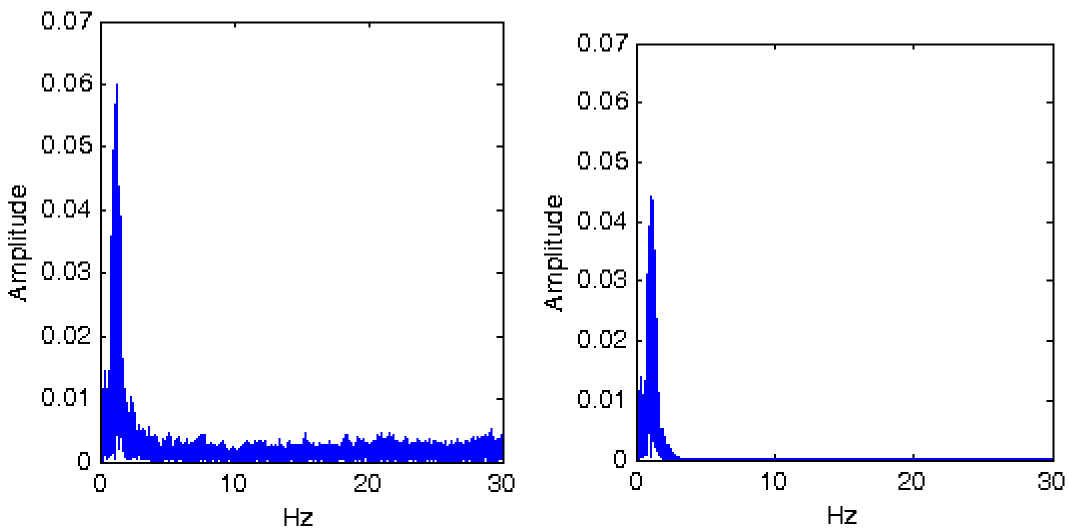


Figure 37. Single-Sided Amplitude Spectrum of raw data and filtered data

4.2.2. Separation

After filtering and obtaining the smoother signal the next step of the preprocessing is separating the signal parts that contain information of writing movement. In this project, two different separation methods are used. One of them is used for to separate the template signals that will be used in recognition step and the other separation method is used for separating the characters of the written words that wanted to be recognized.

Collecting the template signal is done continuously. To be more clear, repetitions of each character is collected and saved in the same signal sequence. For each character there are 20 repetitions and we need to use them separately for recognition step. As it can be seen from the Figure 38 below there are signal parts where there is no information about the writing. These parts are the places that the signals' amplitude values are almost zero. The zero valued parts mean that there is no acceleration and no movement. They occur because in writing process between every repetition of a character the users wait and don't move the acceleration sensor for a few seconds (almost 3 seconds).

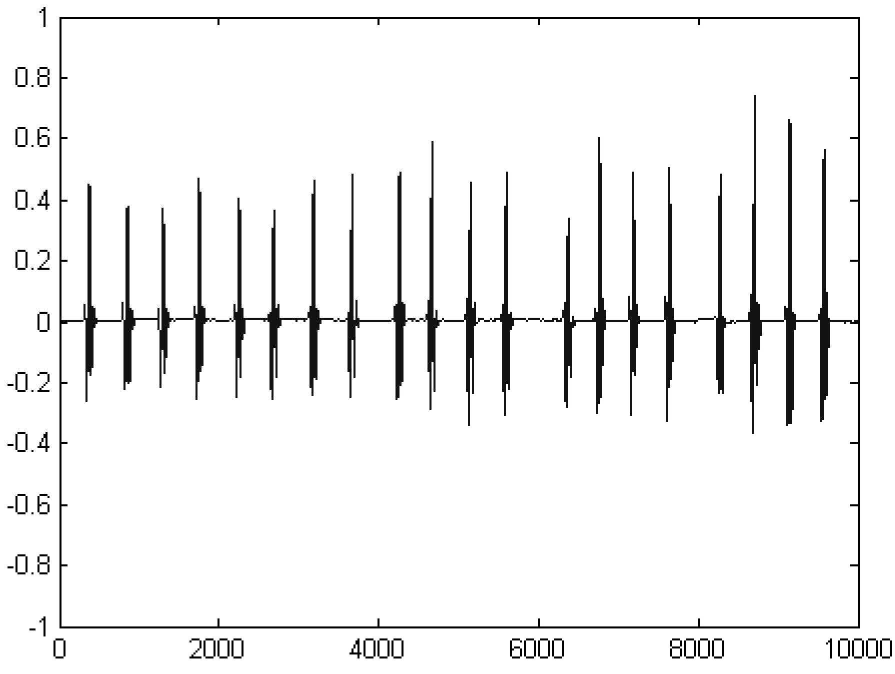


Figure 38. Filtered repetitions of the lower case 'a' character

To separate each repetition, firstly the first and the last points of each signal need to be found. The change in amplitude value is the only thing that we can use to find these points. In a perfect system, the zero valued parts could have been used and we could have said that, if the amplitude value of a point is not equals to zero, then these points are the signal parts that contain the movement. But it is not possible to obtain a perfect signal. So we should use the changes in amplitude values.

It is easy to explain this method in terms of algorithm. A variable (k) can be used to see the changes in amplitude. If there is a change between the present (i) and the next (i+1) data sample and if it is greater than a specified value (m), then add a specified value (t) to the variable (k) that represents the change. The difference value between the data samples could be negative and positive but in both case there is a change in amplitude, so during computing the absolute value of the difference is used. And if the difference is not greater than the specified value (m) then we can say that there is no change in amplitude.

The main problem is deciding for the value of *m*. If it is chosen so big then we can lose some important parts of the signal and if it is chosen so small then it will try to separate the vibrations in waiting parts that filter cannot remove. For the signals used in this study the threshold for the difference value of 0.002 is found to be enough to separate optimally.

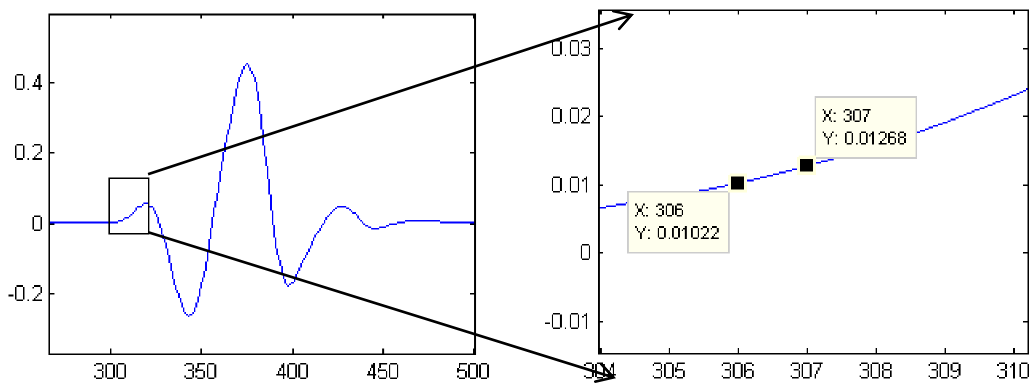


Figure 39. Amplitude change between samples of the signal

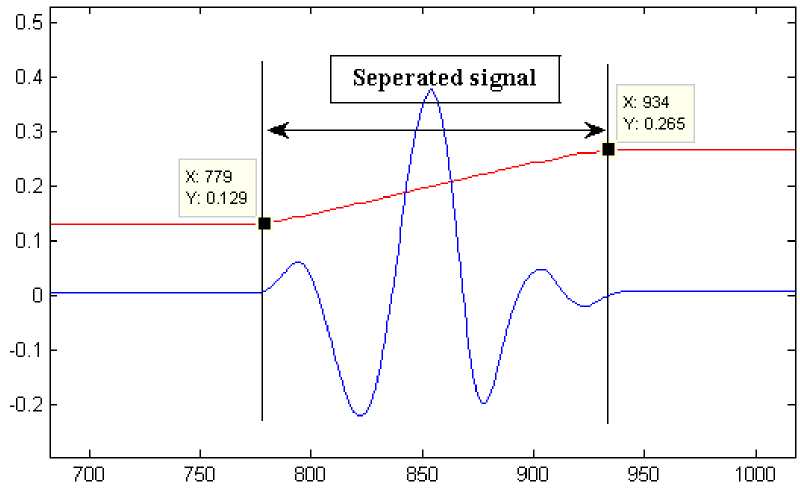


Figure 40. Detecting the first and last data points of the signal

The Figure 40 shows the result of the separation algorithm. The red line in figure shows the changes in the amplitude (*k* variable).

In the case that there is a vibration, that the changes in the amplitude value are greater than the *m*, then the predicted length of the signal could be used to achieve this problem. If we compare the lengths of the writing movement signals and the pure vibration, we can say that generally the signal's length will be longer. So the possible minimum length of the signal in this study is chosen to contain 50 data points.

3-axis accelerometer data are used in this study and we have three different data for a single character. Z axis is not used since there is no movement in this axis, so only X and Y-axis are used. In some cases, writing in one axis takes more space than the other axis, so it has more data points than the other one. If we consider only one axis to find the separation points, then we could lose important parts in the other axis. So firstly, the possible first and last data points of a single character repetition in both axes are found and if one of them contains more data points than the other one, then first and last data points are chosen as the longer ones.

As mentioned at the beginning of the separation topic there are two different separation methods used in this study. Mostly they are almost same. The main difference between these methods is the axes used. In case of written words, users write a single character in 2-axis (X axis, Y axis) but after writing a character there will be a movement out of the writing plane to write the other character. This movement will be out of the writing plane which means there will be movement acceleration in Z axis.

This movement in Z axis can be used for separating the characters of the words, because this acceleration belongs to the movement between each character. However there will also be accelerations in X and Y-axes during this movement. Since the X and Y-axis will be used for recognition, the X and Y-axis movements related to the movement in Z axis and also the duration between writing each character shouldn't be used and they have to be removed. To remove these parts of the signals we need to find the first and last data points of the movement between characters by using the Z axis. If there is a movement in Z axis then we find its first and last data points and use these points to make the amplitude of the signals between these points in X and Y axes zero. Other problem to solve in separating the characters of a word is that we cannot limit the number of characters for a word and we cannot know how many characters the user will write. The only thing we know is that, generally a word cannot contain more than 20 characters.

4.2.3. Normalization

After filtering and separating every repetition of a single character, we have 20 different signals belong to that character. In an ideal system, all repetitions of a character should be in same length and the amplitude value of a data point should be the same for each repetition, but our system is not ideal and it is not possible to have a system like this. Even for writing the same character, there will be different accelerations and decelerations between repetitions. We have signals that their overall shapes are same but their amplitude values and lengths are different. The recognition algorithm (DTW) used in this study can achieve the problem of inequalities in lengths, but this algorithm is also used the amplitude values of the signals to match them. The difference in amplitude may cause undesired recognition results or mismatching between data samples. So the signals are needed to be scaled in amplitude with same limits. There are both negative and positive acceleration values in raw data so the signals need to be normalized between a positive and negative amplitude value. In this study, normalization limits are chosen +1 and -1. To normalize the signals between -1 and 1 the normalization formula below can be used.

$$y = m + (x - A) * (\mathrm{n} - \mathrm{m})/(B - A) \tag{4.11}$$

where x is the signal that wanted to be normalized, m is the minimum limit, n is the maximum limit, A is the maximum amplitude value of the signal and B is the minimum amplitude value of the signal. To normalize between -1 and 1 the equation 4.11 can be in the form of

$$y = -1 + (x - A) * (1 - (-1))/(B - A) \tag{4.12}$$

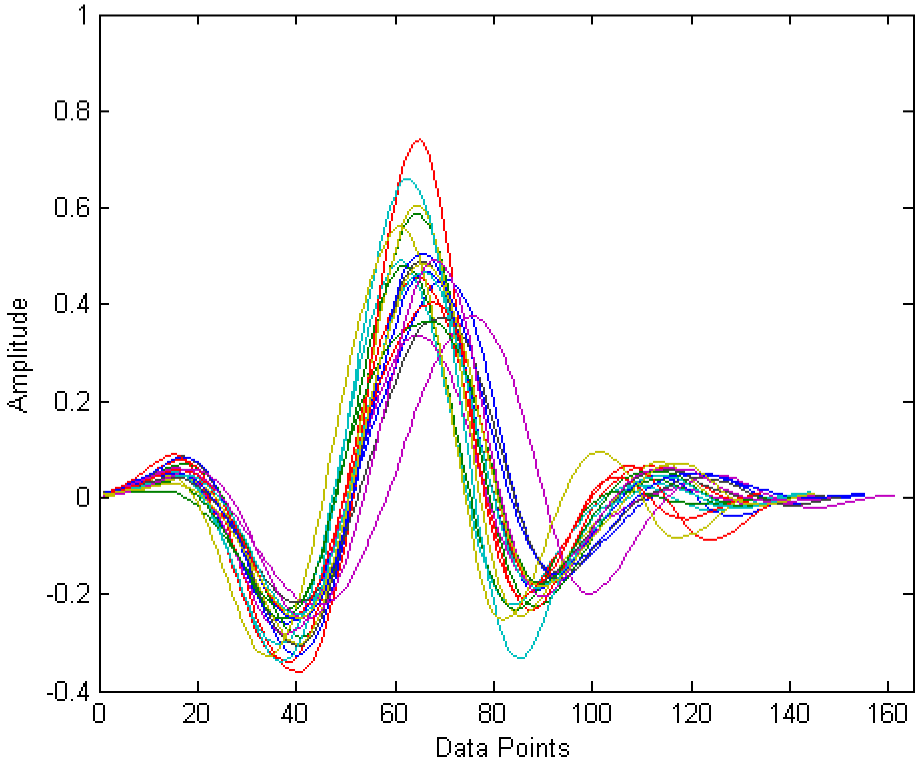


Figure 41. Repetition signals before normalization

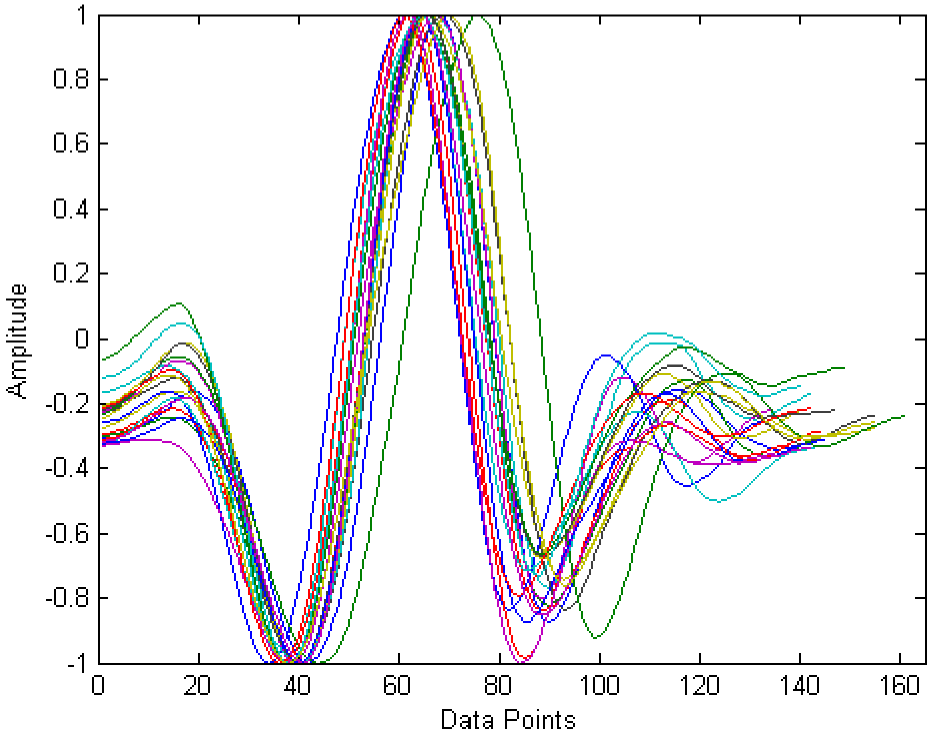


Figure 42. Repetition signals after normalization

4.3. Results

In this part of the thesis, the results obtained from experimental works are presented. The results are divided into three parts: raw data, derivative data and quantized data. There is an important point to consider about the durations of algorithms, the given durations are the durations of comparing the all database repetitions with each other. So it is normal to take so much time, especially for DTW. The lengths of the signals are not same, but if we consider them as same then the duration of DTW is approximately the duration value divided by 520, since we have 520 signals in our database.

4.3.1. Raw Data

User Dependent: In this part the results obtained from the acceleration data of a single user that knows how to use the ring like device very well are shown. The algorithms applied to the data with leave one out method, which means a sample is chosen and comparing with all of the other samples.

Linear Alignment with Euclidean:

- The accurate recognition rate : 96.1538%
- Duration : 63.44 seconds

Table 4. Recognition test for characters (linear alignment with Euclidean)

	Recognized As																									
Test	**a**	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**	**m**	**n**	**o**	**p**	**q**	**r**	**s**	**t**	**u**	**v**	**w**	**x**	**y**	**z**
a	20																									
b		19														1										
c			19																1							
d				20																						
e					19																					1
f						19														1						
g	1						19																			
h								17										2								1
i									20																	
j										20																
k		1						1			18															
l												20														
m													20													
n														20												
o															20											
p		3														17										
q				1													18					1				
r								1										19								
s																			20							
t						1														19						
u																					20					
v																						20				
w												1											19			
x																								20		
y																					1				19	
z														1												19

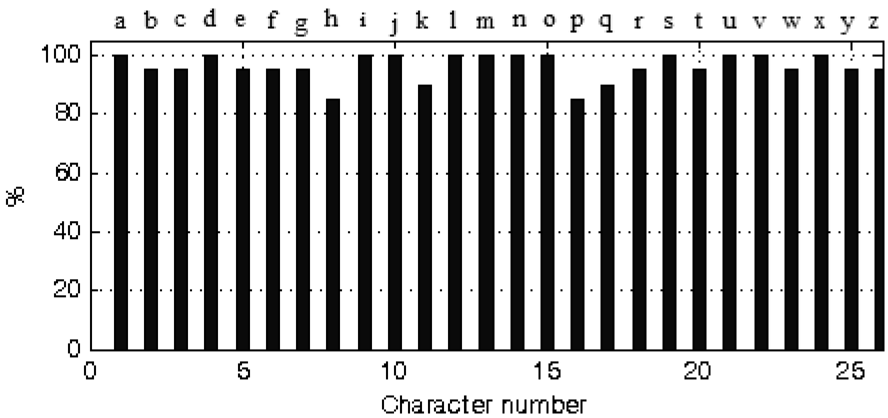


Figure 43. Accuracy of each character (linear alignment with Euclidean)

Linear Alignment with Manhattan:

- The accurate recognition rate : 96.1538%
- Duration : 57.26 seconds

Table 5. Recognition test for characters (linear alignment with Manhattan)

Test \ Recognized As	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	20																									
b		19														1										
c			19																1							
d				20																						
e					19																					1
f						19														1						
g	1						19																			
h								17										2								1
i									20																	
j										20																
k		1						1			18															
l												20														
m													20													
n														20												
o															20											
p		2														18										
q																	19					1				
r								1										19								
s																			20							
t				1		1												1		17						
u																					20					
v																						20				
w												1											19			
x																								20		
y																					1				19	
z														1												19

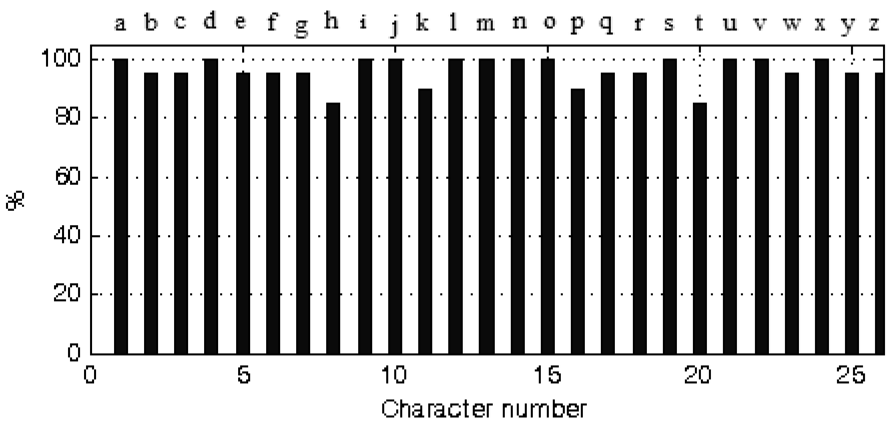


Figure 44. Accuracy of each character (linear alignment with Manhattan)

Linear Alignment with Chessboard:

- The accurate recognition rate : 95.7692%
- Duration : 58.10 seconds

Table 6. Recognition test for characters (linear alignment with Chessboard)

Test \ Recognized As	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	20																									
b		19														1										
c			19																1							
d				20																						
e					19																					1
f						19														1						
g							20																			
h								18										2								
i									20																	
j										20																
k		2						1			17															
l												20														
m													20													
n														20												
o															20											
p		5														15										
q				1													18					1				
r								1										19								
s			1																19							
t						1														19						
u																					20					
v																						20				
w												1											19			
x																								20		
y																					1				19	
z														1												19

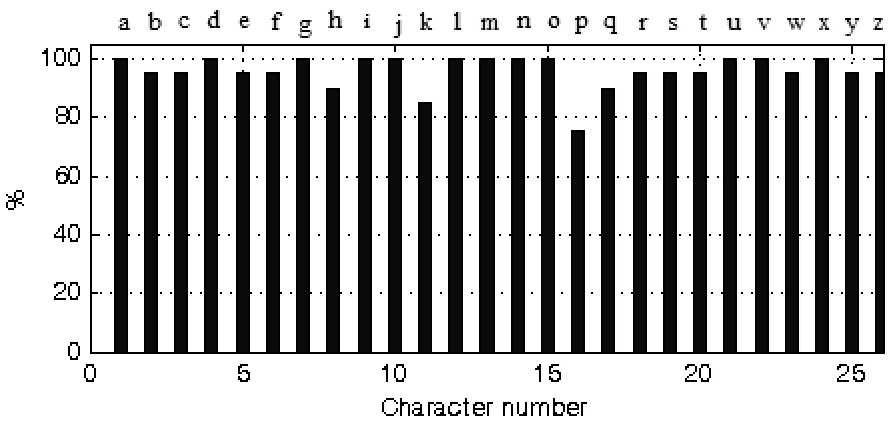


Figure 45. Accuracy of each character (linear alignment with Chessboard)

<u>Nonlinear Alignment-DTW with Euclidean:</u>

- The accurate recognition rate : 97.8846%
- Duration : 8066.30 seconds

Table 7. Recognition test for characters (DTW with Euclidean)

Recognized As

Test	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	20																									
b		20																								
c			19																				1			
d				20																						
e					19																					1
f						20																				
g							19												1							
h		1						19																		
i									19					1												
j										20																
k											20															
l												20														
m													20													
n														20												
o															20											
p		1														19										
q																	19					1				
r																		20								
s																			20							
t																	1	2		17						
u																					20					
v																						20				
w																							20			
x																								20		
y																					1				19	
z																										20

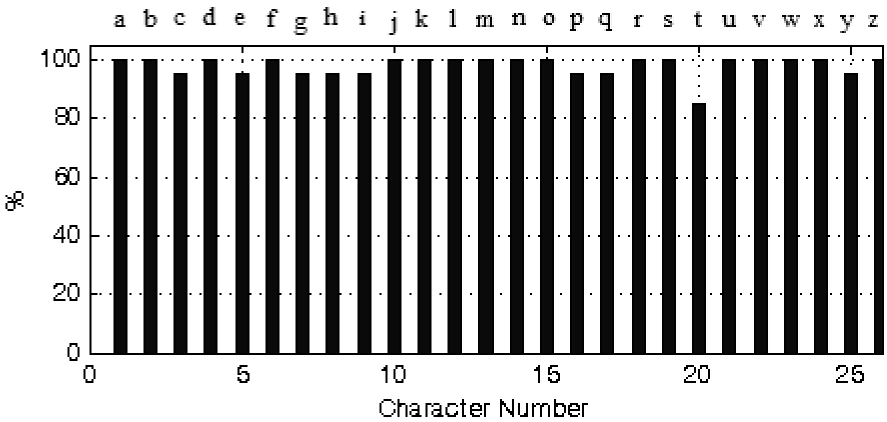


Figure 46. Accuracy of each character (DTW with Euclidean)

<u>Nonlinear Alignment-DTW with Manhattan:</u>

- The accurate recognition rate : 97.8846%
- Duration : 2986.86 seconds

Table 8. Recognition test for characters (DTW with Manhattan)

		Recognized As																									
		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Test	a	20																									
	b		20																								
	c			19											1												
	d				20																						
	e					19																					1
	f						20																				
	g	1						18												1							
	h								20																		
	i									19																1	
	j										20																
	k											20															
	l												20														
	m													20													
	n														20												
	o															20											
	p		1														19										
	q																	19					1				
	r																		20								
	s																			20							
	t																	1	1		18						
	u																					20					
	v																						20				
	w																							20			
	x																								20		
	y																					2				18	
	z																										20

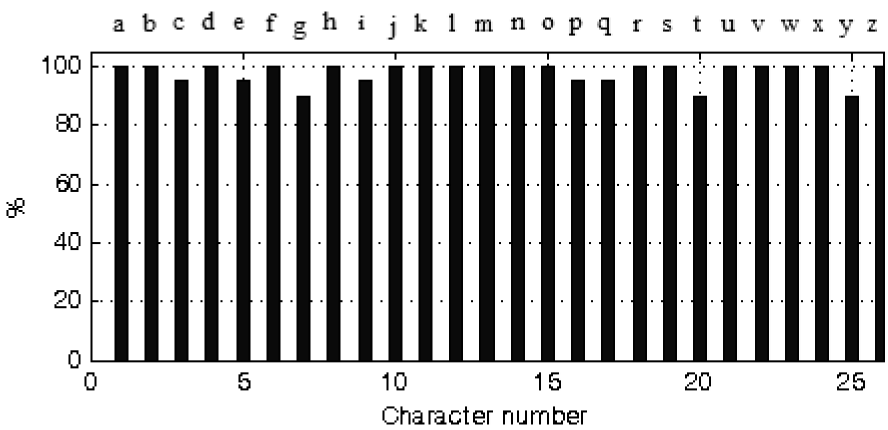


Figure 47. Accuracy of each character (DTW with Manhattan)

Nonlinear Alignment-DTW with Chessboard:

- The accurate recognition rate : 97.50%
- Duration : 2761.42 seconds

Table 9. Recognition test for characters (DTW with Chessboard)

Test	Recognized As																									
	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	20																									
b		20																								
c			19		1																					
d				20																						
e					19																					1
f						20																				
g							18								1				1							
h		1						19																		
i									19					1												
j										20																
k											20															
l												20														
m													20													
n														20												
o															20											
p		1														19										
q																	19					1				
r																		19		1						
s																			20							
t																	1	2		17						
u																					20					
v																						20				
w																							20			
x																								20		
y																					1				19	
z																										20

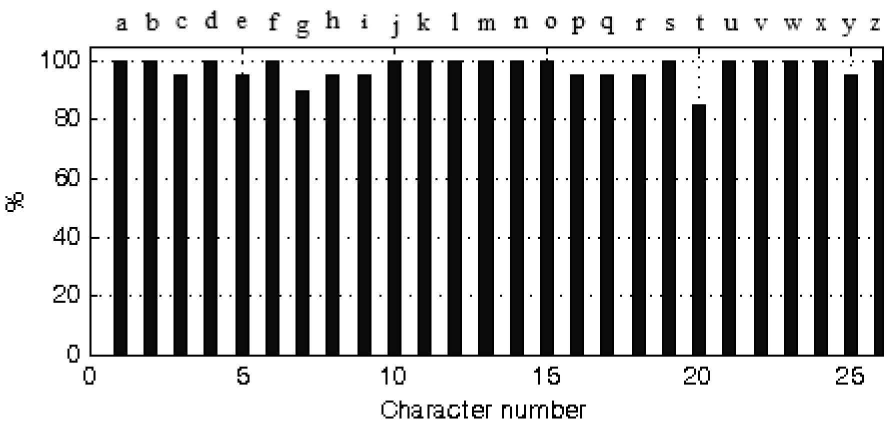


Figure 48. Accuracy of each character (DTW with Chessboard)

<u>Correlation Coefficient:</u>

- The accurate recognition rate : 95.1923%
- Duration : 39.22 seconds

Table 10. Recognition test for characters (Correlation Coefficient)

Test	Recognized As																									
	a	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**	**m**	**n**	**o**	**p**	**q**	**r**	**s**	**t**	**u**	**v**	**w**	**x**	**y**	**z**
a	20																									
b		20																								
c			18		2																					
d				20																						
e			2		17										1											1
f						19																			1	
g							19																1			
h								20																		
i									19			1														
j										20																
k											19								1							
l												20														
m													20													
n														20												
o			1		1										18											
p		1														19										
q															1		18					1				
r																		19				1				
s								1	2										17							
t																				18	1					1
u																					20					
v																						20				
w	1																1						19			
x																								20		
y																					2				18	
z					1																					19

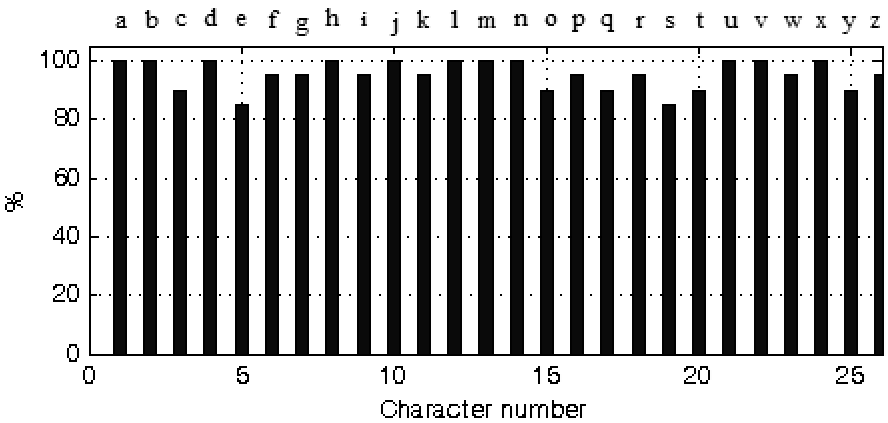


Figure 49. Accuracy of each character (Correlation Coefficient)

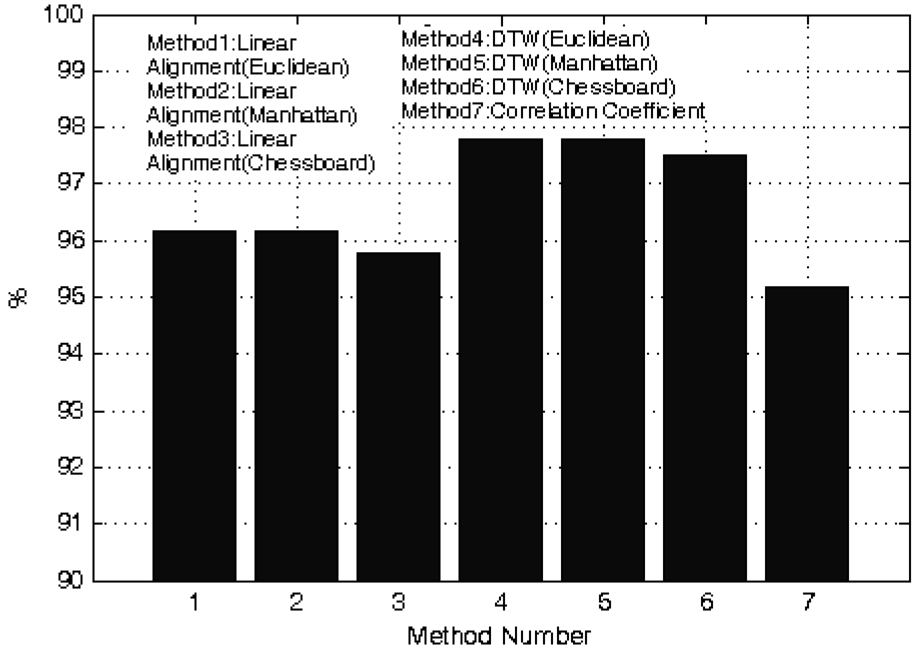


Figure 50. Accurate recognition rate of methods for raw data

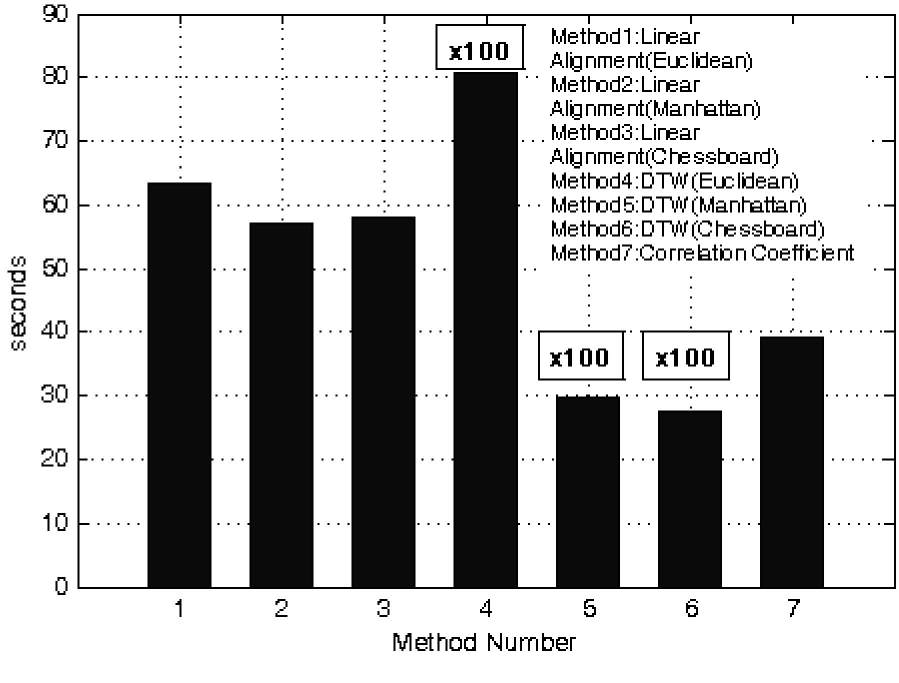


Figure 51. Time duration of each method for raw data

Our aim in this thesis is obtaining the highest recognition rate with the minimum duration. So, for dependent samples, DTW with Manhattan is the most efficient method according to the results presented above.

User Independent: The results explained in this part are obtained from the data comparison of the 5 different users that use the ring like device for the first time and the user that explained in the last part (4.3.2). To be clearer the new users' data are not used for training or template. The recognition algorithms explained in Chapter 3 are applied to the data, respectively.

Linear Alignment with Euclidean:

- The accurate recognition rate : 56.9231%
- Duration : 21.51 seconds

Linear Alignment with Manhattan:

- The accurate recognition rate : 55.3846%
- Duration : 19.92 seconds

Linear Alignment with Chessboard:

- The accurate recognition rate : 56.9231%
- Duration : 21.51 seconds

Nonlinear Alignment-DTW with Euclidean:

- The accurate recognition rate : 66.92%
- Duration : 5664.85 seconds

Nonlinear Alignment-DTW with Manhattan:

- The accurate recognition rate : 69.2308%
- Duration : 3033.04 seconds

Nonlinear Alignment-DTW with Chessboard:

- The accurate recognition rate : 69.2308%
- Duration : 2729.98 seconds

Correlation Coefficient:

- The accurate recognition rate : 58.6538%
- Duration : 12.93 seconds

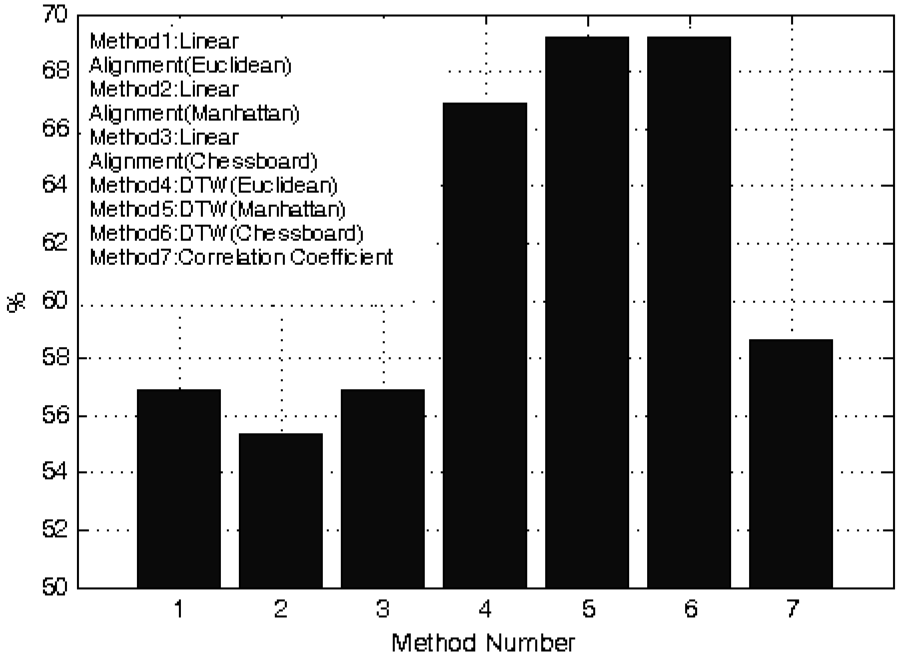


Figure 52. Accurate recognition rate of methods for raw data (user independent)

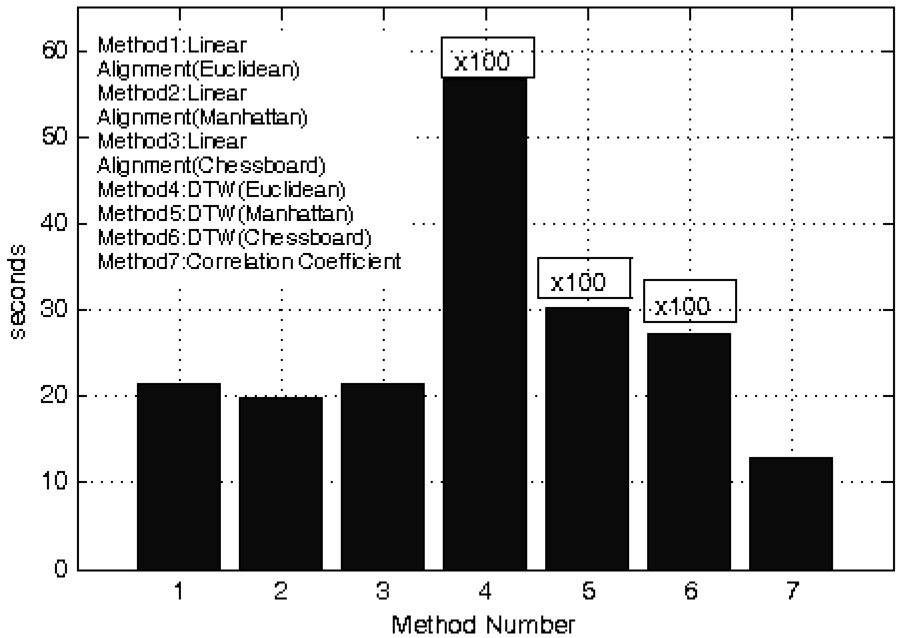


Figure 53. Time duration of each method for raw data (user independent)

The efficiency of the DTW can be seen easily with user independent data results. DTW increases the accuracy by 11% compared to results of linear alignment and correlation coefficient. The most efficient method is DTW with Chessboard.

4.3.2. Derivative Data

In this part the results obtained from the first derivatives of the acceleration signals are shown. The data used for derivation are not normalized.

<u>*Linear Alignment with Euclidean:*</u>

- The accurate recognition rate : 98.46%
- Duration : 65.4 seconds

Table 11. Recognition test for characters (Linear Alignment with Euclidean)

	Recognized As																									
Test	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	20																									
b		20																								
c			19		1																					
d				20																						
e					19									1												
f						20																				
g							19												1							
h								20																		
i									19												1					
j										20																
k											20															
l												20														
m													20													
n														20												
o															20											
p																20										
q																	19					1				
r																		20								
s																			20							
t																		1		19						
u																					20					
v																						20				
w	1											1											19			
x																								20		
y																					1				19	
z																										20

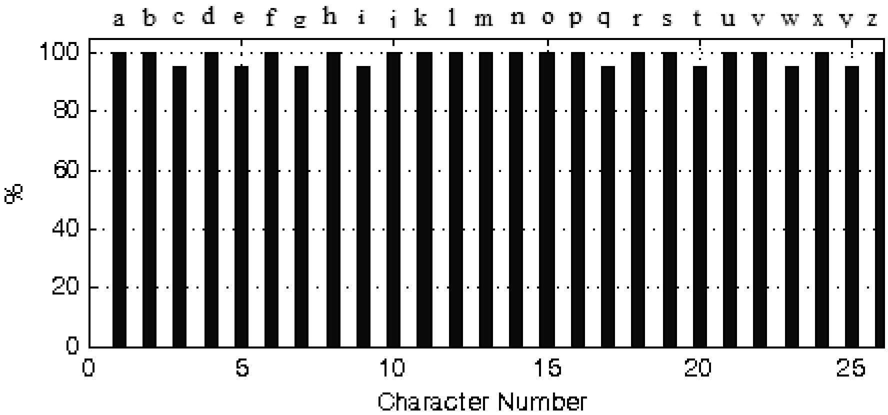


Figure 54. Accuracy of each character (Linear Alignment with Euclidean)

<u>Linear Alignment with Manhattan:</u>

- The accurate recognition rate : 96.15%
- Duration : 58 seconds

Table 12. Recognition test for characters (Linear Alignment with Manhattan)

		Recognized As																									
		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Test	a	20																									
	b		19														1										
	c			19																1							
	d				20																						
	e					19																					1
	f						19														1						
	g	1						19																			
	h								17										2								1
	i									20																	
	j										20																
	k		1						1			18															
	l												20														
	m													20													
	n														20												
	o															20											
	p		2														18										
	q																	19					1				
	r								1										19								
	s																			20							
	t				1		1												1		17						
	u																					20					
	v																						20				
	w	1											1											19			
	x																								20		
	y																					1				19	
	z														1												19

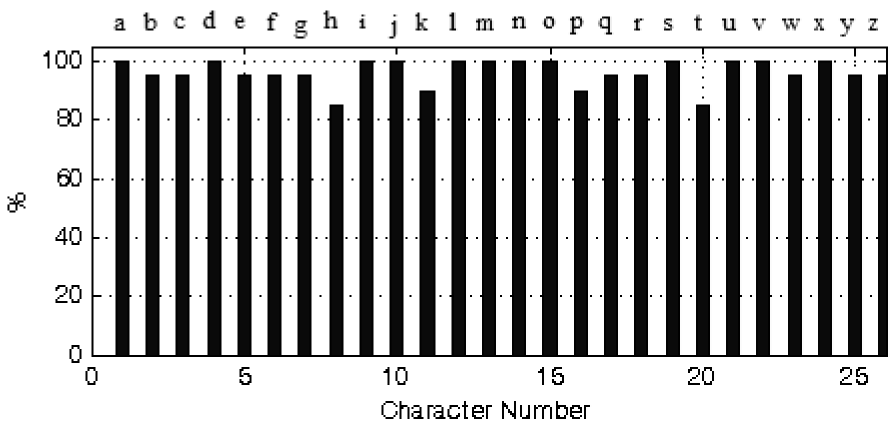


Figure 55. Accuracy of each character (Linear Alignment with Manhattan)

Linear Alignment with Chessboard:

- The accurate recognition rate : 97.88%
- Duration : 59.4 seconds

Table 13. Recognition test for characters (Linear Alignment with Chessboard)

Test \ Recognized As	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	20																									
b		20																								
c			19		1																					
d				20																						
e					19									1												
f						20																				
g							19												1							
h								17								1										
i									19												1					
j										20																
k											20															
l												20														
m													20													
n														20												
o															20											
p																20										
q																	19					1				
r																1		19								
s																			20							
t																	1	1		18						
u																					20					
v																						20				
w	1											1											19			
x																								20		
y																					1				19	
z																										20

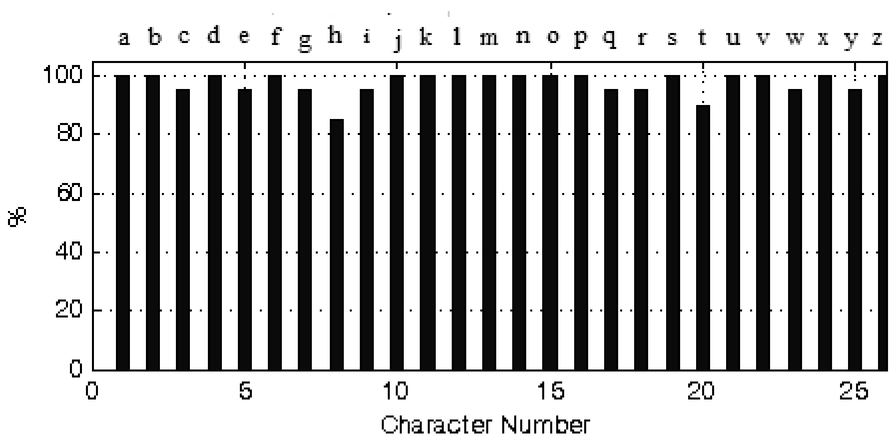


Figure 56. Accuracy of each character (Linear Alignment with Chessboard)

<u>Nonlinear Alignment-DTW with Euclidean:</u>

- The accurate recognition rate : 98.65%
- Duration : 7057 seconds

Table 14. Recognition test for characters (DTW with Euclidean)

	Recognized As																									
Test	**a**	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**	**m**	**n**	**o**	**p**	**q**	**r**	**s**	**t**	**u**	**v**	**w**	**x**	**y**	**z**
a	20																									
b		20																								
c			19																				1			
d				20																						
e					19																				1	
f						20																				
g	1						19																			
h								20																		
i									20																	
j										20																
k											20															
l												20														
m													20													
n														20												
o															20											
p																20										
q				1													18					1				
r																		20								
s																			20							
t																				20						
u																					20					
v																						20				
w	1											1											19			
x																								20		
y																					1				19	
z																										20

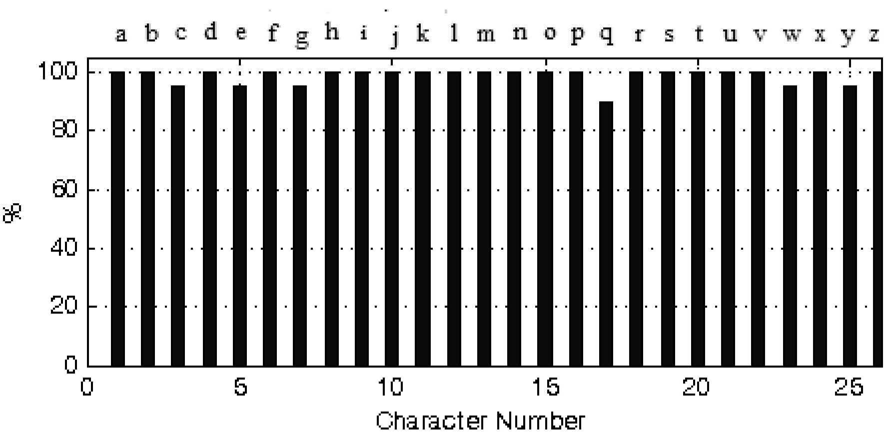


Figure 57. Accuracy of each character (DTW with Euclidean)

Nonlinear Alignment-DTW with Manhattan:

- The accurate recognition rate : 97.8%
- Duration : 5823 seconds

Table 15. Recognition test for characters (DTW with Manhattan)

		Recognized As																									
		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Test	a	20																									
	b		19														1										
	c			19																1							
	d				20																						
	e					19																					1
	f						19														1						
	g	1						19																			
	h								17										2								1
	i									20																	
	j										20																
	k		1						1			18															
	l												20														
	m													20													
	n														20												
	o															20											
	p		2														18										
	q																	19					1				
	r								1										19								
	s																			20							
	t				1		1												1		17						
	u																					20					
	v																						20				
	w												1											19			
	x																								20		
	y																					1				19	
	z														1												19

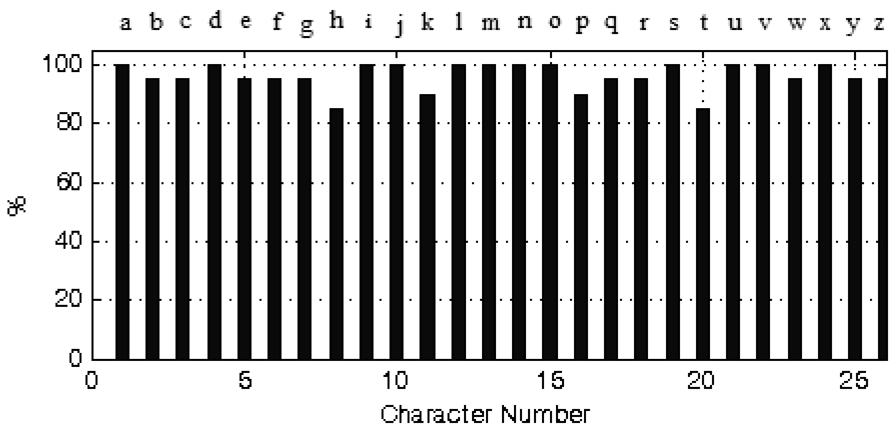


Figure 58. Accuracy of each character (DTW with Manhattan)

Nonlinear Alignment-DTW with Chessboard:

- The accurate recognition rate : 98.65%
- Duration : 5948 seconds

Table 16. Recognition test for characters (DTW with Chessboard)

Recognized As

Test	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	20																									
b		20																								
c			19																				1			
d				20																						
e					19																				1	
f						20																				
g	1						19																			
h								20																		
i									20																	
j										20																
k											20															
l												20														
m													20													
n														20												
o															20											
p																20										
q				1													18					1				
r																		20								
s																			20							
t																				20						
u																					20					
v																						20				
w	1											1											19			
x																								20		
y																					1				19	
z																										20

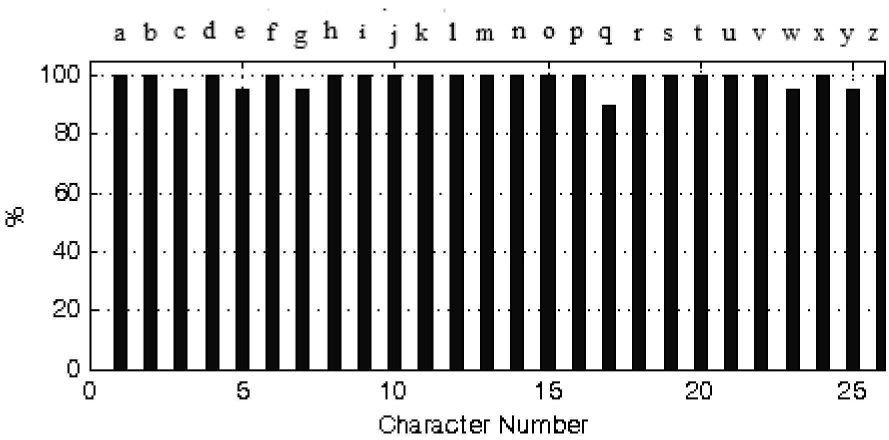


Figure 59. Accuracy of each character (DTW with Chessboard)

Correlation Coefficient:

- The accurate recognition rate : 95.57%
- Duration : 44.6 seconds

Table 17. Recognition test for characters (Correlation Coefficient)

		Recognized As																									
		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Test	a	20																									
	b		20																								
	c			16		2										2											
	d				20																						
	e			3		16										1											
	f						20																				
	g							19																1			
	h								20																		
	i									19			1														
	j										20																
	k								1			19															
	l												20														
	m													20													
	n														19								1				
	o			2		2										16											
	p																20										
	q									1						1		18									
	r																		20								
	s																			20							
	t																				18	2					
	u																					20					
	v																						20				
	w				1																			19			
	x																								20		
	y									1												1				18	
	z																										20

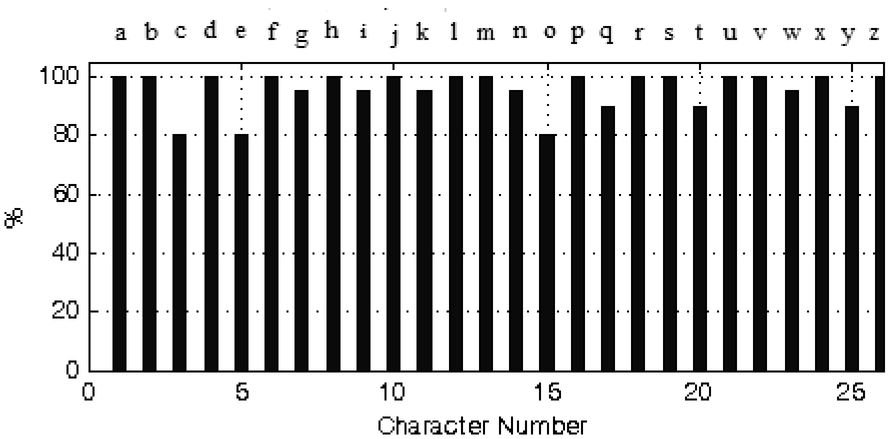


Figure 60. Accuracy of each character (Correlation Coefficient)

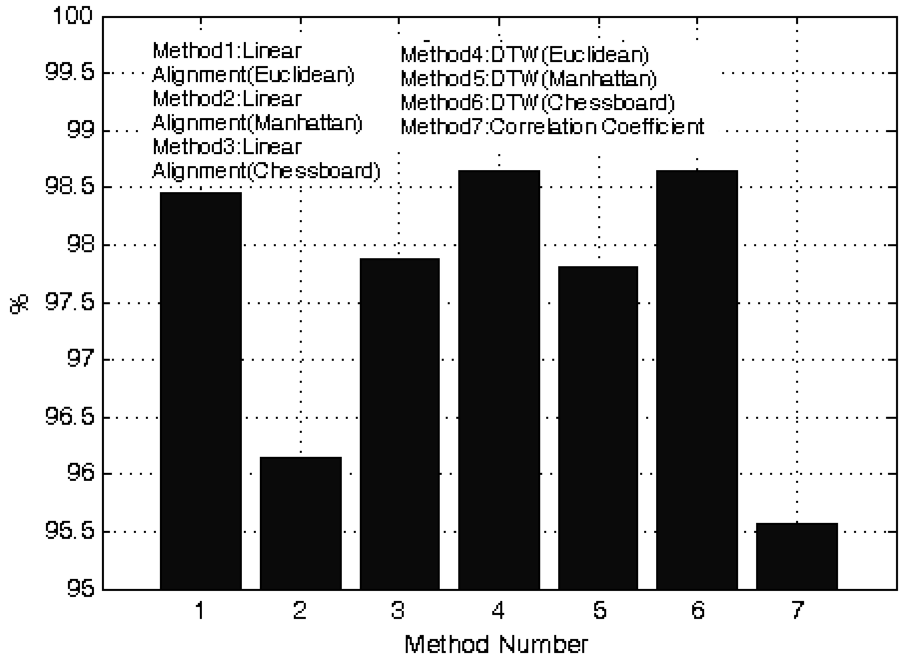


Figure 61. Accurate recognition rate of methods for derivative data

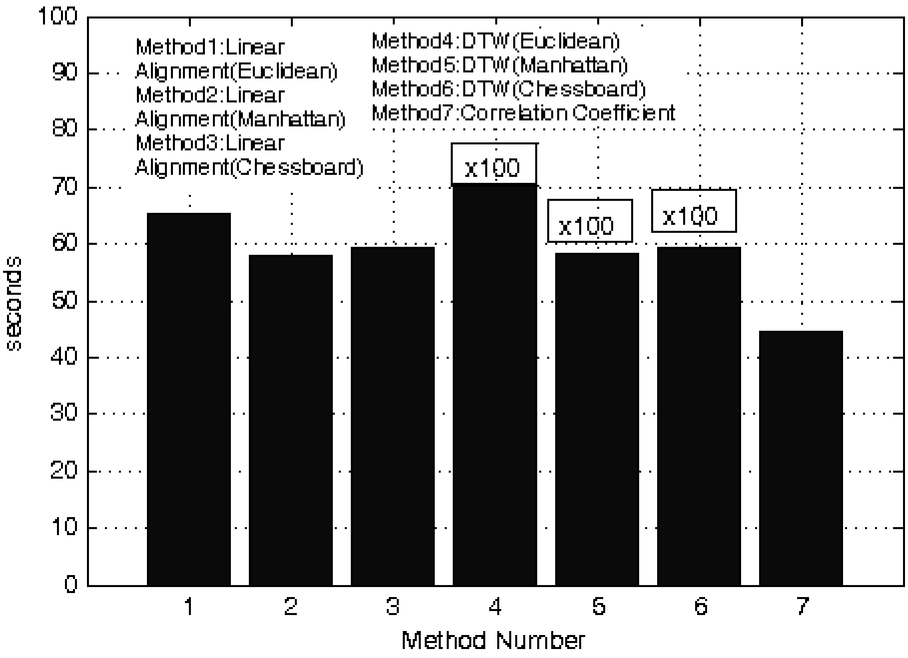


Figure 62. Time duration of each method for derivative data

Taking the derivatives of the signals increases the recognition rate accuracy compared to the raw data results. The most effective method is DTW with Chessboard.

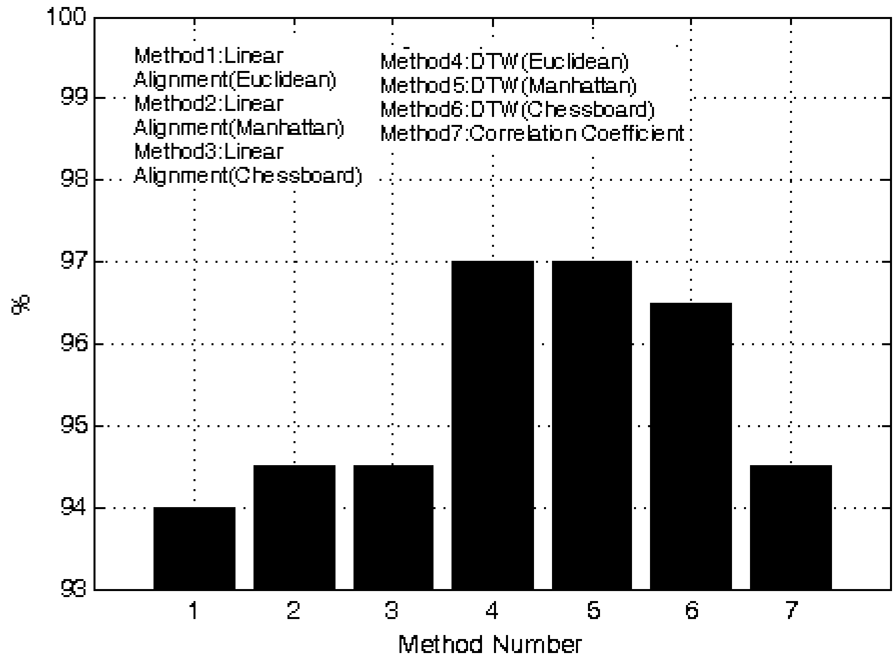


Figure 63. Accurate recognition rate of methods for raw data (digits)

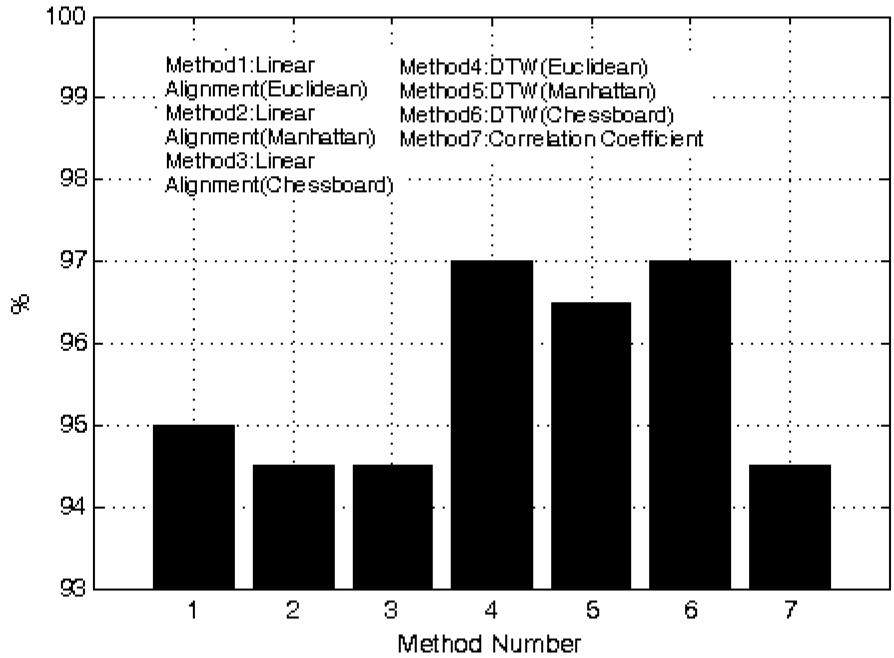


Figure 64. Accurate recognition rate of methods for derivative data (digits)

For comparison with some of the related studies in the literature ((Sung-Do Choi, 2006), (Jeen-Shing Wang, 2013), (Shashidhar Patil, 2015)), recognition rates of each method for digits are shown in the figures. DTW gives higher recognition rate than linear alignment and correlation.

4.3.3. Data after Quantization

In this part the results obtained from the quantization are presented.

The results of linear alignment with Euclidean, Manhattan and Chessboard and also Correlation Coefficient are the same with very short duration differences. Their accurate recognition rate is 95.96%.

Table 18. Recognition test for characters (linear alignment)

		Recognized As																									
		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Test	a	20																									
	b		20																								
	c			19																1							
	d				20																						
	e					19									1												
	f					1	19																				
	g	1						19																			
	h								18										2								
	i									19												1					
	j										20																
	k		1						1			18															
	l												20														
	m													20													
	n														20												
	o															20											
	p		1														19										
	q				3													16					1				
	r														1				19								
	s			1																19							
	t				1																19						
	u																					20					
	v																						20				
	w															1								19			
	x																								20		
	y																					1				19	
	z					1									1												18

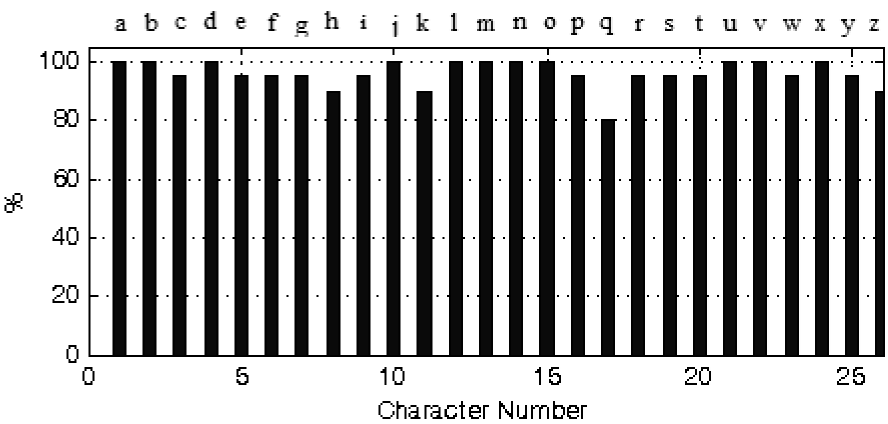


Figure 65. Accuracy of each character (linear alignment)

Nonlinear Alignment-DTW with Euclidean:

- The accurate recognition rate : 96.73%
- Duration : 6903 seconds

Table 19. Recognition test for characters (DTW with Euclidean)

		Recognized As																									
		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
Test	a	20																									
	b		19														1										
	c			19		1																					
	d				20																						
	e					19																					1
	f						20																				
	g	1						19																			
	h								20																		
	i									19												1					
	j										20																
	k								1			19															
	l			1									19														
	m													20													
	n														20												
	o															20											
	p		1														19										
	q				1													18					1				
	r								1										19								
	s																			20							
	t				2														1		17						
	u																					20					
	v																						20				
	w																			1				19			
	x																								20		
	y																					1				19	
	z					1																					19

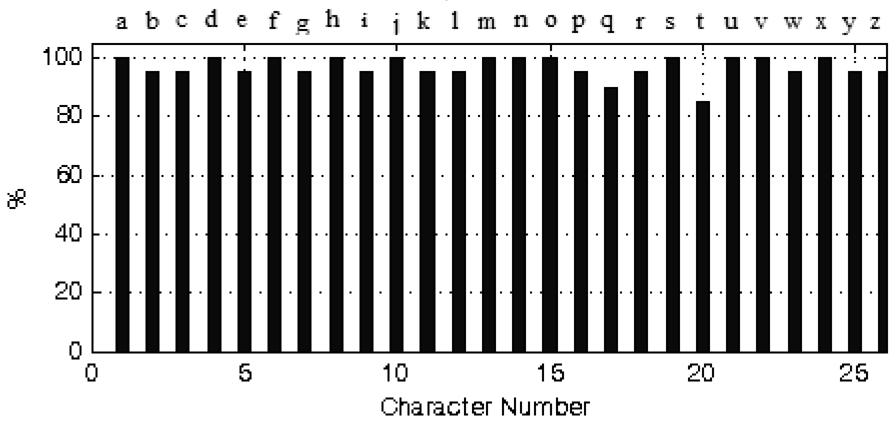


Figure 66. Accuracy of each character (DTW with Euclidean)

Nonlinear Alignment-DTW with Manhattan:

- The accurate recognition rate : 96.53%
- Duration : 3631 seconds

Table 20. Recognition test for characters (DTW with Manhattan)

Test \ Recognized As	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	20																									
b		19														1										
c			18		1							1														
d				20																						
e					19																					1
f						20																				
g	1						19																			
h								20																		
i									19												1					
j										20																
k								1			19															
l			1									19														
m													20													
n														20												
o															20											
p		1														19										
q				1													18					1				
r								1										19								
s																			20							
t				2														1		17						
u																					20					
v																						20				
w																			1				19			
x																								20		
y																					1				19	
z					1																					19

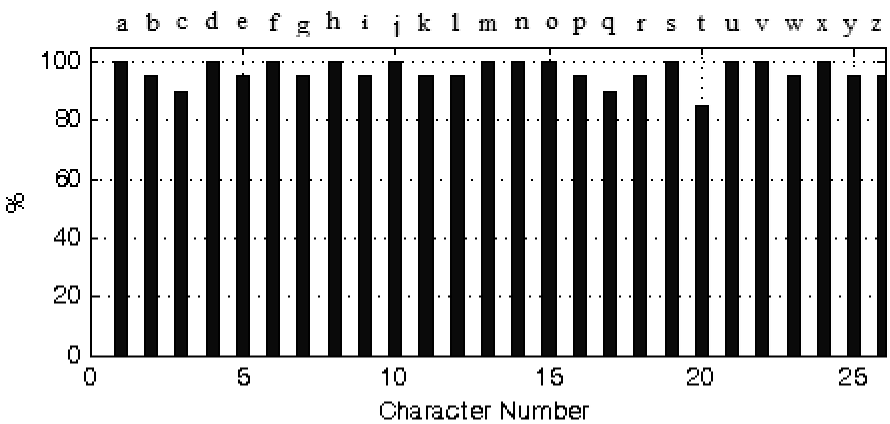


Figure 67. Accuracy of each character (DTW with Manhattan)

Nonlinear Alignment-DTW with Chessboard:

- The accurate recognition rate : 96.9%
- Duration : 3853 seconds

Table 21. Recognition test for characters (DTW with Chessboard)

	Recognized As																									
Test	**a**	**b**	**c**	**d**	**e**	**f**	**g**	**h**	**i**	**j**	**k**	**l**	**m**	**n**	**o**	**p**	**q**	**r**	**s**	**t**	**u**	**v**	**w**	**x**	**y**	**z**
a	20																									
b		19														1										
c			19		1																					
d				20																						
e					19																					1
f						20																				
g							20																			
h								19	1																	
i									19												1					
j										20																
k								1			19															
l			1									19														
m													20													
n														20												
o															20											
p		2														18										
q				1													18					1				
r								1										19								
s																			20							
t				2																18						
u																					20					
v																						20				
w																			1				19			
x																								20		
y																									20	
z					1																					19

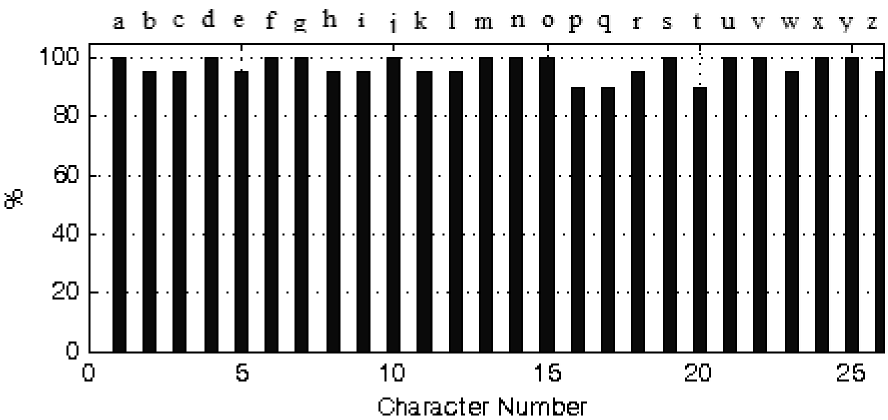


Figure 68. Accuracy of each character (DTW with Chessboard)

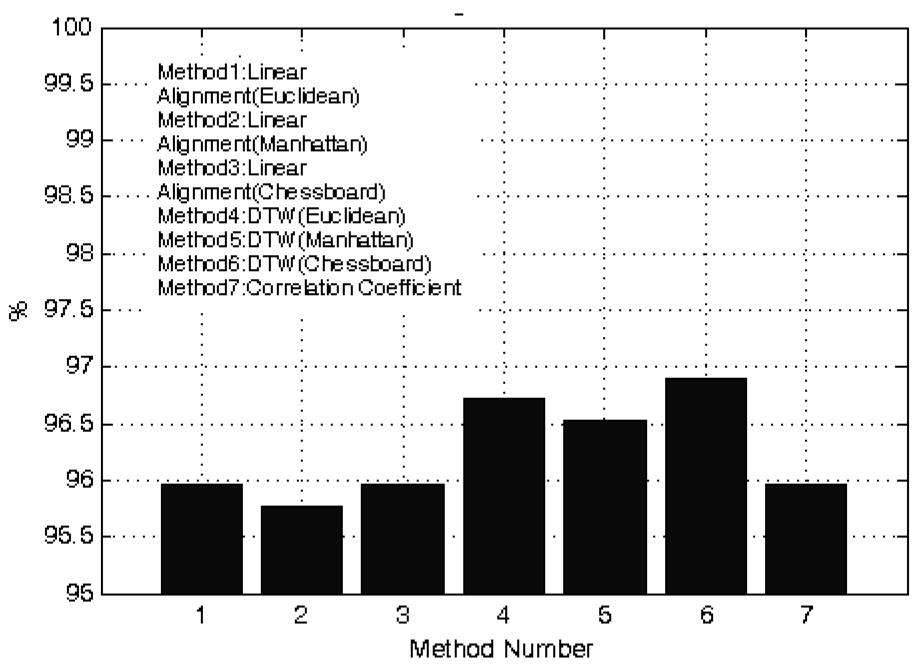


Figure 69. Accurate recognition rate of methods for data after quantization

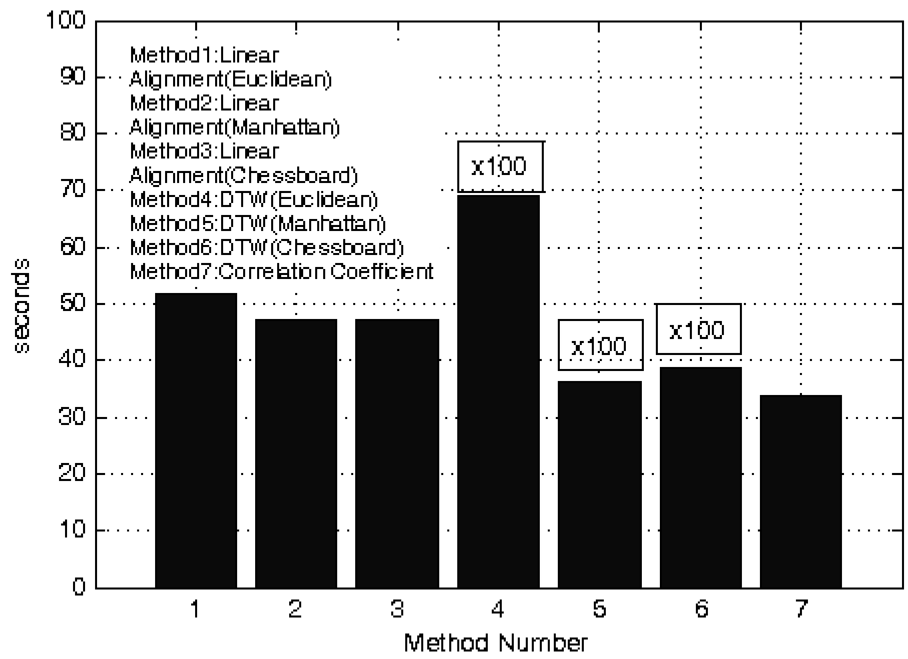


Figure 70. Time duration of each method for data after quantization

The most effective method is DTW with Chessboard.

The graph below shows the overall recognition rates of DTW with 3 different data.

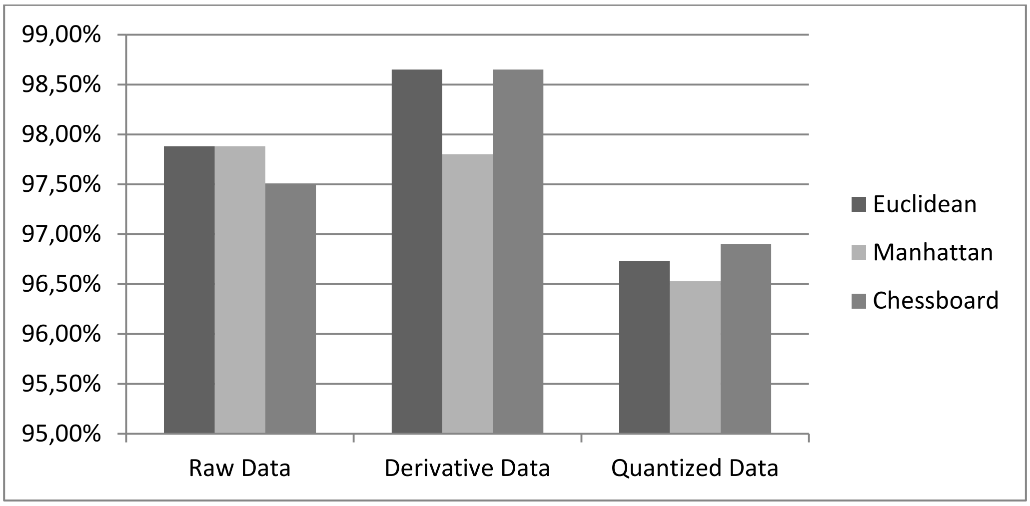


Figure 71. Overall recognition rates of DTW

The graph below shows the overall duration of DTW with 3 different data in seconds.

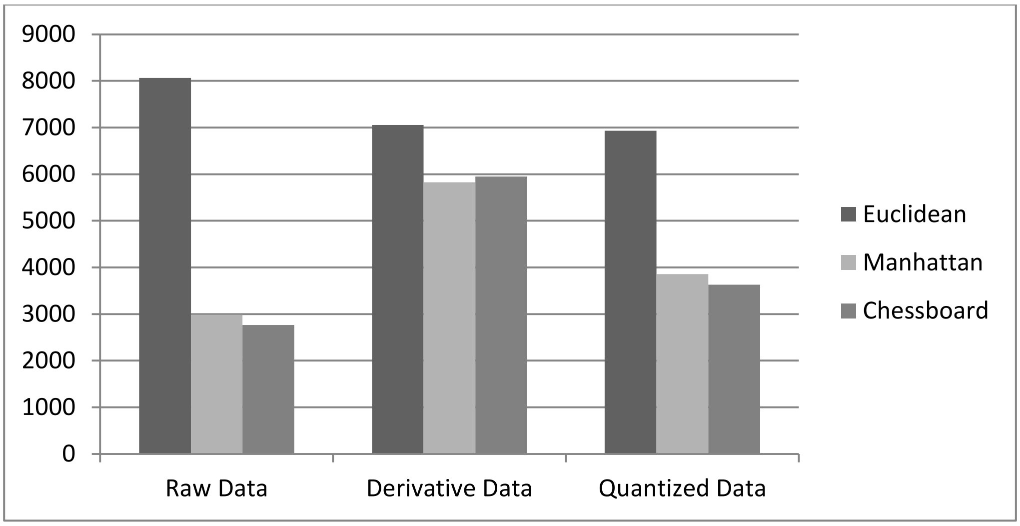


Figure 72. Overall duration of DTW

The graph in Figure 69 shows that DTW gives the best recognition rate for both Euclidean and Chessboard. Quantized data decreases the recognition rates on all three different versions of DTW. Derivation gives the best rate when Euclidean and Chessboard distances are used.

Both quantization and derivation reduce the duration on Euclidean as shown in Figure 70, however derivative version of DTW (DDTW) gives higher recognition rate. So, comparing with DDTW, quantization is not an effective method for our dataset.

CHAPTER 5

CONCLUSION

In this thesis, firstly we have reviewed the character recognition studies. The existing studies in the literature are mostly related to the optical character recognition (OCR). With printed or typed texts and characters, OCR gives satisfied results. However, with handwritten characters it's not easy to obtain high recognition rates since the writing movement dynamics of the characters are not considered in OCR. Some studies in the literature measure the writing movement dynamics with sensors. We also have found that gesture recognition is very likely with our aim and there are lots of gesture recognition studies than character recognition using sensors.

Gesture recognition is simply making the computers and machines understand the human body movements, so controlling these machines without any physical interaction can be possible. Generally, in these studies, acceleration sensors are used and in some of those, the acceleration sensors also contain angular velocity sensors (gyro sensors). But gyro data needs lots of complex calculations and it takes more time. One of the aims of this thesis is having the maximum recognition rate with minimum duration. So we haven't used the gyro data, instead of that we have used a 3-axis accelerometer in a ring-like device that the users wear the ring part on their middle finger and write the characters on the surface parallel to ground.

Generally, gesture recognition studies contain very simple and easy-to-write gestures for any user. And also the number of the gestures is very limited in these studies. For our database, we have used lowercase English alphabet that contains 26 characters and these characters are more complex than most of the gesture recognition studies.

Like most of the other sensors, acceleration sensors have high frequency inner noise. To obtain better recognition results, this high frequency noise is needed to be removed. For the preprocessing part of our study, firstly, the raw acceleration data coming from the sensor have been filtered with the MAF, which is the most effective and the most used filter in these studies related with acceleration sensor. The window size of the filter is chosen to have 15 samples, since an MAF with window size of 15 removes the frequencies higher than 4Hz roughly. The 0-4Hz is also the frequency range of writing so the noise in our data could be removed. After filtering, we have applied segmentation process to the signals. When the acceleration signals of the characters are collected, all repetitions have been written continuously, but between each repetition there are signal parts with no movement. We have used these parts to find the starting and the end points of the signals. After segmentation, we have noticed that the amplitude values of repetitions are very different, so we have normalized the signals between range of +1 and -1 to obtain more likely signals.

In second part, we have noticed two important things, the first one is that the lengths of the signals are not equal and the second one is that the acceleration and deceleration of the repetitions of the same character have different amplitude values and they are shifted in time. Normalization in preprocessing reduced the amplitude difference, but it could not solve the time shifting problem. So, we have decided to use DTW, which is an algorithm used for alignment of the two signals nonlinearly. Briefly, it is finding the optimal warping path that gives the minimum valued warping cost and this path needs to be related to the best matching of the two signals.

When linear alignment and DTW are compared nonlinear alignment gives higher accurate recognition rate than linear alignment, since the acceleration signals are shifted in time. For user dependent data, the average recognition rates of the linear alignment and DTW are 96.02% and 97.75%, respectively. The effectiveness of the DTW can be seen easily from the results of the user independent data collected from five users that wear and write with the ring-like device for the first time. For user independent data, the average recognition rates of the linear alignment and DTW are 56.4% and 68.46%, respectively. The recognition rate of the user independent data is lower than the user dependent data, because the new user signals are compared with the signals in the database.

Besides all the positive aspects of the DTW, there is a problem called as singularities. It is the matching a wide area of the first signal with only one point of the other signal, just because their amplitude values are very close. A study has been found in literature says that by taking the derivative of the signals, the shape information of them can be obtained. The shape information is independent from the amplitude values, although it uses the difference of the amplitude values to find the derivatives. This method is called as DDTW (Eamonn J. Keogh, 2001). Our thesis is the first study that uses this method in accelerometer based character recognition. The average recognition rate that obtained with DDTW method is 98.4%, 97.4% and 95.57% respectively for DTW, linear alignment and correlation coefficient. These recognition rate results are higher than the results of the raw data. The most surprising part of the DDTW is that the signals used for derivation were filtered and segmented signals. The normalization step has not been applied to these signals. This method also speeds up the duration of DTW process for Euclidean distance matrix but speeds down for Manhattan and Chessboard distance matrices.

To speed up the algorithm we have applied quantization process to the signals and then tested the DTW results. For Euclidean distance matrix it has actually speed up the algorithm approximately 12.5% with reducing the recognition rate by 1%. But this duration is almost same with the DDTW.

For DTW, DDTW and quantization we have tested two extra distance metrics; Manhattan distance and Chessboard distance. The results show that the duration of these two methods are shorter than Euclidean. For raw data Euclidean and Manhattan give almost same rate and give higher recognition rates than Chessboard. For derivative data the highest rate is on Euclidean and Chessboard and for quantization Chessboard gives the higher recognition rate.

REFERENCES

A. Ratanamahatana, E. K. (2004). *Everything you know about dynamic time warping is wrong*. Paper presented at the 3rd Workshop on Mining Temporal and Sequential Data,in conjunction with 10th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining Seattle.

Ahmad Akl , S. V. (2010). *Accelerometer-based gesture recognition via dynamic-time warping, affinity propagation, & compressive sensing*. Paper presented at the 2010 IEEE International Conference on Acoustics, Speech and Signal Processing, Dallas, TX.

ARM mbed NXP LPC1768 Development Board. (2016, 28.07.2016). Retrieved from https://www.pololu.com/product/2150

Cassisi, C. (2012). Similarity Measures and Dimensionality Reduction Techniques for Time Series Data Mining. In A. Karahoca (Ed.), *Advances in Data Mining Knowledge Discovery and Applications*: InTech.

Chebyshev distance. (2016, 28.05.2016). Retrieved from https://en.wikipedia.org/wiki/Chebyshev_distance

Culp, S. L. (2008). *Warping Methods for Means and Variances in Functional Data*. (PhD), The University of Michigan.

Çivi yazısı nedir? (2011, 28.07.2016). Retrieved from http://www.webmastersitesi.com/webmaster-sozlugu/226436-civi-yazisi-nedir.htm

Davide Figo, P. C. D., Diogo R. Ferreira, Joao M. P. Cardoso. (2010). Preprocessing Techniques for Context Recognition from Accelerometer Data. *Personal and Ubiquitous Computing, 14*(7), 645-662.

Distance based models (2013, 28.07.2016). Retrieved from http://www.cs.colostate.edu/~cs545/fall13/dokuwiki/lib/exe/fetch.php?media=wiki%3A11_distances.pdf

Eamonn J. Keogh, M. J. P. (2001). *Derivative Dynamic Time Warping*. Paper presented at the First SIAM International Conference on Data Mining

Google Checks Out Library Books. (2004). [Press release]. Retrieved from http://googlepress.blogspot.com.tr/2004/12/google-checks-out-library-books.html

Hieroglyph. (2016). Retrieved from https://www.flickr.com/photos/neiljs/sets/72157602367245779/

Jeen-Shing Wang, Y.-L. H., Cheng-Ling Chu. (2013). *Online Handwriting Recognition Using an Accelerometer-Based Pen Device*. Paper presented at the 2nd

International Conference on Advances in Computer Science and Engineering (CSE 2013).

Jiayang Liu, Z. W., and Lin Zhong. (2009). *uWave: Accelerometer-based personalized gesture recognition and its applications*. Paper presented at the Pervasive Computing and Communications, 2009. PerCom 2009. IEEE International Conference on.

Joseph Lee Rodgers, W. A. N. (1988). Thirteen Ways to Look at the Correlation Coefficient. *The American Statistician, 42*(1), 59-66.

Kim, Y. S. (2005). A new wearable input device: SCURRY. *IEEE Transactions on Industrial Electronics*, *52*(6), 1490 - 1499. doi:10.1109/TIE.2005.858736

Klingmann, M. (2009). *Accelerometer-Based Gesture Recognition with the iPhone.* (MSc.), Goldsmiths University of London, London.

Leskovec, J. (2014). Cluster *Mining of Massive Datasets* (2 ed.).

Liu, J. (2010). An Accelerometer-Based Gesture Recognition Algorithm and its Application for 3D Interaction. *Computer Science and Information Systems, 7*(1).

Mace, D. (2013). *Accelerometer-Based Hand Gesture Recognition using Feature Weighted Naïve Bayesian Classifiers and Dynamic Time Warping.* Paper presented at the IUI '13 Companion Proceedings of the companion publication of the 2013 international conference on Intelligent user interfaces companion, USA.

MATLAB2008-Help. (2008). Discrete Fourier transform: MATLAB.

Meenaakumari, M. (2013). MEMS ACCELEROMETER BASED HAND GESTURE RECOGNITION. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET), 2*(5), 1886-1892.

MPU-6000 and MPU-6050 Product Specification Revision 3.4. (2013). (Vol. PS-MPU-6000A-00). USA: InvenSense Inc.

Online OCR. (2016). Retrieved from http://www.onlineocr.net/service/about

OPPENHEIM, A. V. (1998). *Discrete Time Signal Processing* (2 ed.): Prentice-Hall, Inc.

Patil, M. K. J. (2014). GESTURE RECOGNITION OF HANDWRITTEN DIGIT USING ACCELEROMETER BASED DIGITAL PEN. *International Journal of Application or Innovation in Engineering & Management, 3*(4), 353-357.

Perng, J. K. (1999). *Acceleration sensing glove (ASG)*. Paper presented at the Wearable Computers, 1999. Digest of Papers. The Third International Symposium on, San Francisco, CA, USA.

Qian Chen, G. H., Fanglin Gu, Peng Xiang (2012). *Learning Optimal Warping Window Size of DTW for Time Series Classification* Paper presented at the The 11th International Conference on Information Sciences, Signal Processing and their Applications: Special Sessions.

Schumm, L. (2014). Who created the first alphabet? Retrieved from http://www.history.com/news/ask-history/who-created-the-first-alphabet

Selina Chu, E. K., David Hart, Michael Pazzani. (2002). *Iterative Deepening Dynamic Time Warping for Time Series*. Paper presented at the In Proc 2 nd SIAM International Conference on Data Mining.

Serway, R. A. (2008). *PHYSICS for Scientists and Engineers with Modern Physics* (7 ed.). USA: Thomson Learning, Inc.

Shashidhar Patil, D. K., Seongsill Park, Youngho Chai. (2015). Handwriting Recognition in Free Space Using WIMU-based Hand Motion Analysis. *Journal of Sensors*.

Shiqi Zhang, C. Y., Yan Zhang. (2008, 2008-07-21). *Handwritten character recognition using orientation quantization based on 3D accelerometer*. Paper presented at the Mobiquitous '08 Proceedings of the 5th Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services, Belgium.

Sung-Do Choi, A. S. L., Soo-Young Lee. (2006). *On-Line Handwritten Character Recognition with 3D Accelerometer*. Paper presented at the International Conference on Information Acquisition, China.

Sung-Jung Cho, J. K. O., Won-Chul Bang, Wook Chang, Eunseok Choi, Yang Jing, Joonkee Cho, Dong Yoon Kim. (2004). *Magic Wand: A Hand-Drawn Gesture Input Device in 3-D Space with Inertial Sensors.* Paper presented at the 9th Int’l Workshop on Frontiers in Handwriting Recognition (IWFHR-9 2004).

Taxicab geometry. (2016, 28 April 2016). Retrieved from https://en.wikipedia.org/wiki/Taxicab_geometry

Texas-Instruments. (2005). Accelerometers and How they Work? Retrieved from http://www2.usfirst.org/2005comp/Manuals/Acceler1.pdf

Texas-Instruments. (2015). eZ430-Chronos Development Tool. In U. s. Guide (Ed.).

Xu, R. (2012). MEMS Accelerometer Based Nonspecific-User Hand Gesture Recognition. *IEEE Sensors Journal, 12*(5), 1166 - 1173. doi:10.1109/JSEN.2011.2166953