# Abstract

Automated recognition of text has been an actively researched since the early days of computers. A 1972 survey cites nearly 130 works on the subject. [1]

Handwriting recognition is a task of transforming a language represented in its spatial form of graphical marks into its symbolic representation. Online Hand Writing Recognition refers to the situation where the recognition is performed concurrently to the writing process.

After a long period of focus on western and East Asian scripts there is now a general trend in the on-line handwriting recognition community to explore recognition of other scripts such as Arabic and various Indic scripts. One difficulty with the Arabic script is the number and position of diacritic marks associated to Arabic characters.

[Our Approach]

Contents

[Abstract 1](#_Toc345197303)

[List of figures 4](#_Toc345197304)

[Abbreviation 5](#_Toc345197305)

[1. Introduction 6](#_Toc345197306)

[2. Background 7](#_Toc345197307)

[2.1. Hand Writing Recognition 7](#_Toc345197308)

[2.1.1. Online vs. offline Handwriting Recognition 7](#_Toc345197309)

[2.1.2. Writer dependent vs. Writer Independent 8](#_Toc345197310)

[2.1.3. Closed dictionary vs. open Dictionary 8](#_Toc345197311)

[2.2. Characteristics of Arabic Script 8](#_Toc345197312)

[2.3 The ADAB Database 9](#_Toc345197313)

[3. Literature Review 10](#_Toc345197314)

[4. Approach 11](#_Toc345197315)

[4.1. Overview 11](#_Toc345197316)

[4.1.1. Demarcation Points Nomination 11](#_Toc345197317)

[4.1.2. Segmentation selection and Letter classification 11](#_Toc345197318)

[4.1.3. Candidate Word-Part generation and embedding 12](#_Toc345197319)

[4.1.4. Word-Part classification 12](#_Toc345197320)

[4.2. Notes and Limitations 13](#_Toc345197321)

[4.3. Letters Collection 13](#_Toc345197322)

[4.3.1. Letters Extraction from the ADAB Database 14](#_Toc345197323)

[4.4. Samples Preprocessing 14](#_Toc345197324)

[4.4.1. Normalization 14](#_Toc345197325)

[4.4.2. Simplification 15](#_Toc345197326)

[4.4.3. Resampling 15](#_Toc345197327)

[4.5. Feature Extraction 16](#_Toc345197328)

[4.5.1. Shape Context 16](#_Toc345197329)

[4.5.2. Multi Angular Descriptor 16](#_Toc345197330)

[4.6. Metric Embedding 18](#_Toc345197331)

[4.6.1. Earth Movers Distance 18](#_Toc345197332)

[4.7. Features Transformation and Dimensionality Reduction 20](#_Toc345197333)

[4.8. Letters Classification 22](#_Toc345197334)

[4.8.2. Support Vector machine 22](#_Toc345197335)

[4.8.3. MultiClass SVM 24](#_Toc345197336)

[4.9. Segmentation 25](#_Toc345197337)

[4.9.1. Segmentation in progress 25](#_Toc345197338)

[4.9.2. Segmentation Selection 25](#_Toc345197339)

[4.10. Word Part Recognition 25](#_Toc345197340)

[4.10.1. KDTree/LSH 25](#_Toc345197341)

[5. Experimental Details and Results 26](#_Toc345197342)

[6. Summary and Conclusions 26](#_Toc345197343)

[6.1. Contribution to the field 26](#_Toc345197344)

[6.2. Future Directions 26](#_Toc345197345)

[References 27](#_Toc345197346)

[4.7. Appendix 29](#_Toc345197347)

[4.7.1. Wavelets 29](#_Toc345197348)

# List of figures

[Figure 1 – The main stages of the recognition system. 12](#_Toc352684055)

[Figure 2 – The set of all distinctive main body Arabic letters 16](#_Toc352684056)

[Figure 3 – Different writing styles of the isolated form of the letter ح (Ha) 16](#_Toc352684057)

[Figure 4 – Non Simplified representation of the letter ح (Ha) is shown on the right. A simplified version is shown on the left. 17](#_Toc352684058)

[Figure 5 – Representation of a non-resamples sequence of the letter ى (Y) is shown in the right. The resampled version is shown on the left. 18](#_Toc352684059)

[Figure 6 - diagram of the log-polar bins used to compute the shape context. 19](#_Toc352684060)

[Figure 7 - In this figure we can see an example of three line segments drawn from the same viewpoint, generating the three angles  with the plane of the shape. When the parameter j goes over all contour points we get the vector  describing the shape from the view point with the parameter  goes over all viewpoints. 20](#_Toc352684061)

# Abbreviation

WP – Word Part - a single stroke connected component

DTW – Data Time Warping

EMD – Earth Movers Distance

PCA – Principle Component Analysis

LDA – Linear Discrimination Analysis

SVM – Support Vector Machine

# Introduction

The growing use of keyboard-less handheld devices accelerates a transition in the Human-Computer data input interface. While in the past, keyboards and mice were the primary mode of data entry, nowadays, hand and finger gestures are increasingly used for this task. Consequently, a growing interest in the online character recognition field has taken place. Automated script recognition of Latin, Chinese and Kanji has been a focus of study in the last decade and impressive recognition rates were achieved. However, Arabic text recognition is at an early stage. The reason for this is lack of funds and other utilities such as text database, dictionaries, etc. [2]

The Arabic language is one of the most structured and served languages. It comes as the fifth of the most used languages (as a first language) after Chinese, Hindi, Spanish and English. It is spoken as a first language by nearly 350 million people around the globe, mainly in the Arab countries, which is about 5.5% of the world population (the world population is estimated at 6.44 billion in July 2005) (CIA, 2005). However, almost all Muslims (close to ¼ of the world population) can read Arabic script as it is the language of the Holy Qur’an. The Arabic script evolved from a type of Aramaic, with the earliest known document dating from 512 AD. The Aramaic language has fewer consonants than Arabic (Burrow, 2004). The old Arabic was written without dots or diacritics. The dots were first introduced by Yahya bin Ya’mur (died around 746 AD) and Nasr bin Asim (died around 707 AD), students of Abu Al-Aswad Al-Du’ali (died around 688 AD) who introduced the diacritics to prevent the Qur’an from being misread by Muslims (Al-Fakhri, 1997). Figure 1 shows a sample of an old manuscript of a sentence written without dots or diacritics. Due to the Islamic conquests, the use of Arabic language extended in the 7th and 8th centuries from India to the Atlantic Ocean (Al- Fakhri, 1997). Consequently, many other languages adopted the Arabic alphabet with some changes. Among those languages are Jawi, Urdu, Persian, Ottoman, Kashmiri, Punjabi, Dari, Pashto, Adighe, Baluchi, Ingush, Kazakh, Uzbek, Kyrgyz, Uygur, Sindhi, Lahnda, Hausa, Berber, Comorian, Mandinka, Wolof, Dargwa, and few others. Figure 2 shows samples of some of the above mentioned languages. However, it must be mentioned that some of those languages are currently using Latin characters, but in general, people can still read the Arabic script. It is also worth mentioning that the United Nation adopted Arabic in 1974 as its sixth official language (Strange, 1993). Despite the fact that Arabic alphabets are used in many languages, Arabic Character Recognition (ACR) has not received enough interests from researchers. Little research progress has been achieved as compared to the one done on Latin or Chinese. It has almost only started in 1975 by Nazif (1975), while the earlier research efforts in Latin may be traced back to the middle of the 1940s. However, due to a lack of computing power, no significant work was performed until the 1980s. Recent years have shown a considerable increase in the number of research papers related to ACR. (**Segmentation of Arabic Characters: A Comprehensive Survey**)

Text recognition can be classified into two main fields: online and offline recognition. Online recognition aims recognizing the text as it is being written. However, in the offline script recognition field, a digital image containing text is fed to the computers and the system attempt to recognise the written text [3]. The main existing approaches for script recognition are the holistic approach and the analytic approach. The holistic approach considers the global properties of the written text while the analytic approach involves segmentation and classification of each part of the text.

Arabic text, both handwritten and printed is cursive. An Arabic word consists of one or more sub-words (which will be referred to as word parts). Each word part consists of one character or more. Thus, most of the online Arabic recognition systems perform the recognition soon after the word part scribing is completed using the holistic or the analytic approach. The analysis of Arabic script is further due to obligatory dots and stokes that are placed above or below most letters in addition to the wide range variety of writing fashions.

In this thesis we propose a novel approach which combines both holistic and analytic techniques for recognizing open dictionary Arabic online script. The recognition process, in our approach, is performed whilst the word part is being written. At the first stage, demarcation points are identified using SVM. At the second stage we employ an agile but loose classifier to select letters candidates for each segment. Finally, we holistically recognize the word part by applying a costly yet accurate classifier.

# Background

## Hand Writing Recognition

Handwriting recognition is a task of transforming a language represented in its spatial form of graphical marks into its symbolic representation.

At the highest level, handwriting recognition can be broken into two categories: offline and online.

### Online vs. offline Handwriting Recognition

While much of the today’s data is directly entered into computers using the keyboard, many tasks still exist in which people tend to prefer handwriting over keyboard entry. Note taking (e.g. in classrooms) is a task that can still be done more efficiently by hand for most users. In addition, while people can produce annotated hand sketches very quickly, data entry into a computer using a combination of the mouse and keyboard is relatively time consuming. [4]

Smartphones and tablets are pocket sized consumer devices that can store calendars and address books, provide access to emails, the web, and contain other productivity tools. These devices are too small to have full sized keyboards, or sometimes may be too small for any keyboard at all, requiring pen, hand gestures, figure gestures or voice interface to enter data. [4]

The problem of handwriting recognition has now been a topic of research for over four decades. There are many types of problems (with varying complexity) within handwriting recognition, based on how the data is presented to the recognition system, at what level the data can be unambiguously broke n into pieces (e.g. individual characters or words), and the transcription complexity of the language used. [4]

Offline handwriting recognition focuses on documents that have been written on paper at some previous point of time. Information is presented to the system in the form of scanned image of the paper document. In contrast, online Hand Writing Recognition refers to the situation where the recognition is performed concurrently to the writing process. This requires the use of special equipment, such touch screen or digitizing tablet, to capture the strokes of the pen as that are being written. The trace of a writer’s pen is stored as a sequence of points sampled at equally spaced time intervals. The information captured for each sample is the  coordinates. While this sequence can be used to construct a static image of the writing, thus allowing offline character recognition techniques to be applied, it has been shown [63] that the information about the pen dynamics can be used to obtain a better recognition accuracies than the static data alone. Therefore, it is beneficial to capture the data in an online form, even if the real-time processing requirements can be relaxed. [4]

Another advantage of online handwritten data over offline data is the availability of the stroke segmentation and the order of writing. Ink in static images must first be separated from the image background, creating a potential source of error. The ability to detect the states of “pen-down” (when the pen touches the tablet or the finger touches the touch screen) and “pen-up” can also be used. A single stroke is defined as the sequence of sample points occurring between consecutive pen-down and pen-up transitions. However, a complication occurs when a stroke is added to a character in a word after the rest of the word has already been written, such as the cross of a ‘t’ or an ‘x’, or the dot of an ‘i’ or a ‘j’. These types are called delayed strokes. [4]

### Writer dependent vs. Writer Independent

### Closed dictionary vs. open Dictionary

## Characteristics of Arabic Script

[See Online handwriting recognition for the Arabic Letter Set]

[See [1] G. a. Abandah and M. Z. Khedher, “Analysis of Handwritten Arabic Letters Using Selected Feature Extraction Techniques,” *International Journal of Computer Processing of Languages*, vol. 22, no. 01, pp. 49–73, Mar. 2009.]

The Arabic Aleph bet is widely used for more than twenty different languages such as Farsi, Urdu, Malay, Housa and Ottoman Turkish. Arabic is used in over 20 different countries, written by more than 100 million people and spoken by 234 million people. Although the spoken Arabic is slightly different from country to country, the written Arabic is standard system used all over the Arab world. [5] [6]

Arabic Scripts consists of 28 basic letters, 12 additional special letters, and 8 diacritics. Arabic script is written from right to left in a semi-cursive manner in both printed and handwritten. Most letters are written in four different letter shapes depending on their position in a word, e.g., the letter ع (Ain) appears as ع (isolated), عـ(initial), ـعـ (medial) and ـع (final). Among the basic letters, six are Disconnective – ا (Alef), د (Dal), ذ (Thal), ر (Reh), ز (Zain) and و (Waw). Disconnective letters do not connect to the following letter and have only two shapes each. The presence of this letters interrupts the continuity of the graphic form of a word. We denote connected letters in a word, as word-part. If the word-part is composed of only one letter, this letter will be in its isolated shape. [7]

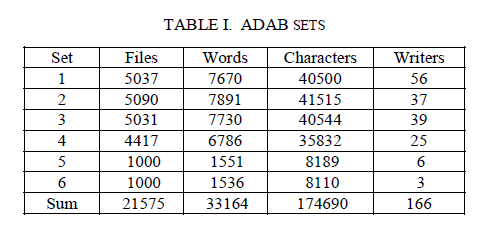
Certain characteristics relating to the obligatory dots and strokes of the Arabic script distinguish it from Roman script, making the recognition of words in Arabic more difficult than in Roman script. First, Most Arabic letters contain dots in addition to the letter body, such as ش (Sheen) which consists of س (Seen) body and three dots above it. In addition to dots, there are stroke that can attach to a letter body creating new letter such as ك, ط and لا. These dots and strokes are called *delayed strokes* since they are usually drawn last in the in handwritten word-part/word. Second, eliminating, adding or moving a dot or stroke could produce a completely different letter and, as a result, produce a word other than the one that was intended (see Table 1). Third, the number of possible variations of delayed strokes is greater than those in Roman scripts, as shown in Figure 2. There are only three such strokes used for English: the cross in the letter *t*, the slash in *x*, and the dots in *i* and *j*.

The Arabic language has some diacritics that are used in the holy book Qur’an and sometimes in teaching material and poetry. These diacritics are small markings used above or below the letters of a word to specify the exact pronunciation of the word. They are not commonly used in the daily, scientific, and business uses, and are not discussed further in this work.(Abandah & Khedher, 2009)

Finally, in Arabic script a top-down writing style called *vertical ligatures* is very common – letters in a word may be written above their consequent letters. In this style, the position of letters cannot be predefined relative to the baseline of the word. [7]

Saabni and El-sana have explored a large collection of Arabic texts and extracted 300,000 different word combinations of 82,000 different word-parts. Ignoring the additional strokes reduced the number of different word-parts to 40,000. [5]

## The ADAB Database

The database ADAB was developed to advance the research and development of Arabic on-line handwritten text recognition systems. This database is developed in cooperation between the Institutfuer Nachrichtentechnik (IfN) and the Research group on intelligent Machines (REGIM). The database consists of 20575 Arabic words handwritten by more than 170 different writers, most of them selected from the narrower range of the National school of Engineering of Sfax (ENIS). The ADAB database is freely available for non-commercial research (www.regim.org). Our aim was to collect a database of handwritten town names written in a similar quality as on a Mobile Phone with a digital input device. The collection process starts when the writer clicks on start bottom. The collection tool generates a town name randomly from 937 Tunisian town/village names, the writer must write the displayed word as it is shown in Fig.3. A pre-label will be automatically assigned to each file. It consists of the postcode in a sequence of Numeric Character References which will be stored in the UPX file format. An InkML file including trajectory information and a plot image of the word trajectory are also generated. Additional information about the writer can also be provided. The ADAB-database is divided to 6 sets. Details about the number of files, words, characters, and writers for each set 1 to 6 are shown in the table below.(Kherallah, Tagougui, Alimi, Abed, & Margner, 2011)

# Literature Review

Automated recognition of text has been an active subject of research since the early days of computers. A 1972 survey cites nearly 130 works on the subject [8]. Despite the age of the subject, it remains one of the most challenging and exciting areas of research in computer science. In recent years it has grown into a mature discipline, producing a huge body of work.

Despite long standing predictions that handwriting, and even paper itself, would become obsolete in the age of the digital computer, both persist. Whilst the computer has hugely simplified the process of producing printed documents, the convenience of a pen and paper still makes it the natural medium for many important tasks.

A brief survey of students in any lecture theatre will confirm the dominance of handwritten notes over those typing on laptops. However, the ease and convenience of having information in digital form provides a powerful incentive to find a way of quickly converting handwritten text into its digital equivalent.

The field of handwriting recognition can be split into two different approaches. The first of these, on-line, deals with the recognition of handwriting captured by a tablet or similar touch-sensitive device, and uses the digitized trace of the pen to recognize the symbol. In this instance the recognizer will have access to the x and y coordinates as a function of time, and thus has temporal information about how the symbol was formed. The second approach concentrates on the recognition of handwriting in the form of an image, and is termed off-line. In this instance only the completed character or word is available. It is this off-line approach that will be taken in this report.

* Arabic printed and handwritten text recognition literature review.
* Segmentation literature review.

Printed Arabic text is like handwritten Latin text, such that connection of characters is an inherent property for Arabic script whether it is typed, printed or handwritten. Most of errors and deficiencies of Arabic recognition systems comes from the segmentation stages Various segmentation algorithms have been proposed in the literature. Given the vast number of papers published on OCR, it is impossible to include all the segmentation methods in this survey. (Zidouri, Sarfraz, Shahab, & Jafri, 2005)

Need to decide about what to write in this section. Should it contain literature review about every stage in the system? i.e. Pre-processing, segmentation, letters recognition, and post-processing.

# Approach

While the holistic approach is a common technique used for Arabic handwriting recognition, this approach is not practical for recognizing words from the unrestricted dictionary (contains all the words in the Arabic dictionary), since it needs to train the system for each word in the dictionary. As mentioned in section ‎2.2, there are about 40,000 valid word parts, with different main strokes. There are much more classes if additional strokes are taken into consideration, practically, a very large number of classes. To overcome this problem, we propose an Arabic word-parts recognizer which breaks up the whole recognition process to 2 restricted recognition tasks. The first is analytical letters classification and the other is holistic word-part recognition from the limited dictionary that contains veritable combinations of letters candidates found in the previous stage. The most significant benefit of this approach is that in both stages the recognition is performed on spaces which contain a very limited class number.

## Overview

In this section we draw the skeleton of our recognition process. Figure 1 presents the main flow of our approach followed by a brief description on each stage.



Figure 1 – The main stages of the recognition system.

### Demarcation Points Nomination

The goal of this stage is to decide coarsely but effectively if a certain point is a demarcation point. This stage is performed while the word part is being written. To identify demarcation point we have used the following 2 Arabic demarcation points characteristic: 1. Demarcation point lives in a *horizontal region*. 2. Demarcation point is usually contained in a *Forward region*. More details and definitions will follow in a section ‎4.9. We present a novel algorithm online segmentation using 2-class SVM. Over segmentation and under-segmentation are the main problems segmentation algorithms encounter. Over segmentation is handled in the next stage. Under-segmentation is handled by a defining a new notion named hyper-letter, which represents combinations of letters that the demarcation points between them is sometimes concealed. More details will be given in section ‎4.9.

### Segmentation selection and Letter classification

Good segmentation outcomes a better letter recognition. The goal of this sub-process is to overcome the over-segmentation problem by selecting the set of demarcation points which gives the best letter recognition results. This is done after the writer has completed scribing the main body of the word-part. Segmentation induces partition to subsequences, where each subsequent is a geometric representation of a letter. The subsequence is classified to a set of 3 candidate letters using RBF-kernel based Multiclass SVM. The quality scoring of the segmentation is based on the recognition score of each letter which is calculated using dynamic programming technique.

### Candidate Word-Part generation and embedding

After the segmentation is determined and each sub-sequent was classified to a small set of hyper-letters candidates, the system generate all possible word-parts sequences using the samples of hyper-letters in our letters database using the most common ways to write the hyper-letter. Here we use the Holistic approach to classify the word-part. However, instead of taking the whole dictionary of possible word-parts, our space of candidate word part only samples of word parts combinations of letters nominated in the previous stage. Subsequently, we extract the features of the generated word-parts and embed them using approximate Earth Movers Distance (EMD) approach described by Shirdhonkar and Jacobs in [9].

### Word-Part classification

The number of possible different word-parts is relatively small. Assume that the written word part composed of 4 letters. For every letter, the second stage has nominated 3 different letters. In the worst case when all combinations are legal, we get at mostdifferent possible word-parts. However, the number of different samples created is very large. Thus, we need a fast classification approach to overcome this obstacle.

EMD is a true metric, thus we can utilize a recently developed tool which allow fast (sub linear time) approximate nearest neighbor (NN) named Linear Sensitivity Hashing (LSH) presented by Gionis et al in [10]. LSH indexes a set of training examples living in a normed space by a number of hash tables, such that the probability of collision is high for similar examples and low for dissimilar ones. In this stage, the system initializes an LSH data structure with the set of word parts generated in the previous stage and use it to efficiently find the most similar word-parts.

## Notes and Limitations

As mentioned before, Arabic is a cursive written language and it contains about 40k possible word parts. By “*possible”*, we mean that there is an Arabic word which contains the word part. Arabic letters may differ by additional stroke above or beneath the main stroke. For example, the Arabic letter ف (Fa) contains a single dot above the main stroke, however the letter ق (Qa), contains double dots, both having identical main body. In our work we recognize and classify the main body of the letter and ignore the additional stroke entirely. As a result, the number of different letters drops from 29 to18 and the number of different possible word parts decreases to []. It is important to comment that taking the additional strokes into consideration may be exploited to boost the classification rate.

The Arabic letters system contain 7 sets of letters that have the same body, and are differentiated only by the additional strokes above or under it. The following table describes the sets of similarity:

|  |  |
| --- | --- |
| Letter Set | Positions of similarity |
| ب,ت,ث | All Positions |
| ع,غ | All Positions |
| ح,ج,خ | All Positions |
| ف,ق | All Positions (very slight differences in Isolated mode, the valley of the letter ق is deeper) |
| ر,ز | All Positions |
| ه,ة | Isolated |
| ى,ي | Isolated |

* In most writing techniques Arabic letters are connected on the baseline.
* We assume that the main body of the letters is written in a single stroke.

The main body of most Arabic letters is written by a single stroke. However, there are some letters that usually written using 2 strokes, such as the letter ـكـ which is the middle \_\_ of the letter ك (Ka). The writer usually writes ـلـ and adds the final upper slanted line when the main body is completed, as if he writes an additional stroke.

Another problem arises when trying to recognize Arabic transcript, is that, different writers may write the main body of the same word part in a different number of strokes. For instance, the main body of the word part بىت (Bayt - Home), is usually written in a single stroke however, sometimes it may be written by some writers using 3 strokes.

We have also considered the common combination of the letter ل followed by the vowel ا as a single letter which is commonly drawn as لا or لا.

Also, three consequent appearances of the letter ب (Ba) in the middle of the word-part looks as follows: ـببنـ (in handwritten font it looks more like ـببنـ). As can be seen very similar to the س (Sa) letter in its medial position ـسـ (which looks more as follows in handwritten text ـسـ), the only to distinguish between the two options is by looking at the additional strokes.

For these mentioned complexities, when recognizing Arabic scripts, most researches have preferred the holistic approach.

## Letters Collection

The data is the most important part of any supervised learning technique. Thus, the data collection stage is an important stage in the development of our system. The Arabic alphabet consists of 29 isolated characters. If we ignore all diacritical marks, which do not carry important information about shape, we obtain 18 shapes.

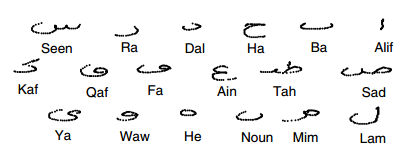


Figure 2 – The set of all distinctive main body Arabic letters

The writing of these characters was not constrained, leading to a wide variety of size, and orientation. Figure (5) shows some samples of the “Ha” letter written by several writers.

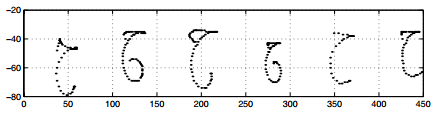


Figure 3 – Different writing styles of the isolated form of the letter ح (Ha)

We utilize two sources of letters samples written in different positions and different types of writing. The first is a self-made database of letters that were collected from different age Arabic writers using digitizer tablet. We discuss these two methods in more details below (‎4.3.1 and ‎4.3.2). In both cases the information is a sequence where the vector denotes the horizontal and vertical coordinates, sampled from the writer’s pen movement.

### A self-collected letters database

We have developed an application, using Matlab, which asks the user to draw a letter on an electronic pad, the sequence data of the letter shape was saved as a file on the file system. The rectangle in which the user was asked to draw the letter is 1X1. See image of the GUI below. [Add an Image of the Matlab data collection UI].

### Letters Extraction from the ADAB Database

The other source was the ADAB database. The ADAB contains Arabic script samples. The information saved is the strokes information; each stroke contains the sequence of  coordinates of the pen down to pen up. The challenge was to match each word part to each stroke, and then manually segment each sequence which describes a word-part to letters (subsequences). Another problem we faced was that automatically corresponding each stroke o each word-part was not easy. [talk more abour it - Wajdi]

## Samples Preprocessing

The data variation of the letters data, which is caused by the uniform temporal sampling of the digitizer, should be reduced as much as possible. Digitizers often oversample slow pen motion regions and under-sample fast motion regions. [3]

This phase contains 3 parts: *Normalization*, *Simplification* and *Resampling*.

### Normalization

Given the written sequence, normalized sequence  is calculated by:

|  |  |
| --- | --- |
|  | Equation 1 |

Where

|  |  |
| --- | --- |
|  | Equation 2 |
|  | Equation 3 |
|  | Equation 4 |
|  | Equation 5 |

The same for the y coordinate.

### Simplification

In order to eliminate redundant points irrelevant for pattern classification and screening out unwanted noise and vibrations in the letter inscription we have used the Douglas-Peucker Polyline Simplification algorithm described in [11]. Briefly, it is a line simplification algorithm to reduce the number of vertices in a piecewise linear curve according to a specified tolerance. The algorithm is also known as Iterative Endpoint Fit.

Assuming the stroke presentation is a sequence of points, the sensitivity parameter that was used in our work is :

|  |  |
| --- | --- |
|  | Equation 6 |



Figure 4 – Non Simplified representation of the letter ح (Ha) is shown on the right. A simplified version is shown on the left.

### Resampling

The resampling is needed to prevent the unbalanced sampling density, which may be influenced by the sampling rate and the user non-uniform letter drawing. The resampling was done using linear piecewise splines. Given a number of points N, this technique returns N equidistant points on given curve.



Figure 5 – Representation of a non-resamples sequence of the letter ى (Y) is shown in the right. The resampled version is shown on the left.

## Feature Extraction

The selection of valuable features is crucial in pattern recognition. In this work we have chosen to work with 2 features, the Shape Context and the Multi Angular Descriptor. Generally, when considering shapes, the contour of the shape is taken into account, thus the following 2 shape descriptors is defined using the contour of the shape. However in the online writing recognition case, the features are applied upon the 2-D sequence.

### Shape Context

Belongie and Malik have presented a point matching approach named Shape Context. [12] The Shape context is a shape matching approach that intended to be a way of describing shapes that allows for measuring shape similarity and the recovering of point correspondences. This approach is based on the following descriptor: Pick n points on the shape’s contour, for each point  on the shape, consider the  other points and calculate the coarse histogram of the relative coordinates. Equation 4 is defined to be the shape context of.

|  |  |
| --- | --- |
|  | Equation 4 |
|  |  |

The bins are normally taken to be uniform log-polar space making the descriptor more sensitive to positions of nearby sample points than to those of points farther away. This distribution over relative positions is robust and compact, yet highly discriminative descriptor. The basic Idea of the Shape Context Descriptor is illustrated in Figure 6. This can be calculated in  time using the Hungarian method.

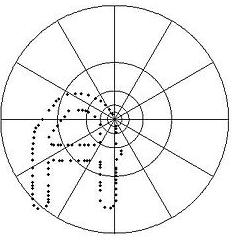


Figure 6 - diagram of the log-polar bins used to compute the shape context.

### Multi Angular Descriptor

The Multi Angular Descriptor (MAD) is a shape recognition method described in [13], which captures the angular view to multi resolution rings in different heights. The shape is treated as a two dimensional set of points and the different rings are upper view points from rings around the shape centroid with different sizes and heights. To enables scale and translation invariance, the sizes and heights of these rings are calculated using the diameter and centroid of the shape.

Formally, let  be a shape and Let  and  be the centroid and the diameter of the shape respectively. Let a set of  point taken uniformly from the extracted contour of. Given a view point from a given ring with height over the shape, the angle, obtained by connecting the point  with each point  and the plain of the shape is a rich description of the shape from this view point. Let R be a ring with the radius  and the center  positioned above the shape  with the height. Let be a set of  viewpoints lying uniformly on the ring R and  to be the angle between the segment  and the plain contains the shape. The vector can be seen as watching the shape S from one upper view point. Illustration can be seen in Figure 7.

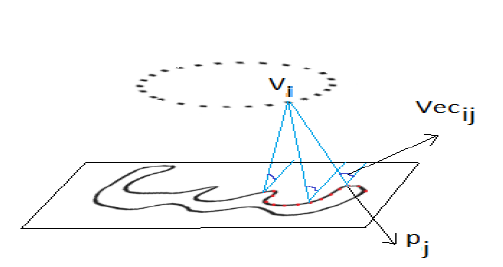


Figure 7 - In this figure we can see an example of three line segments drawn from the same viewpoint, generating the three angles  with the plane of the shape. When the parameter j goes over all contour points we get the vector  describing the shape from the view point with the parameter  goes over all viewpoints.

## Metric Embedding

Given two distributions, it is important to define a quantitative measure of their dissimilarity, with the intent of approximating perceptual dissimilarity as well as possible. Defining a distance between two distributions requires first a notion of distance between the basic features that are aggregated into the distributions. This is known as *ground distance*.

Mathematically, it would be convenient if these distribution distances were true metrics, which would lead to more efficient data structures and search algorithms. As well, this characteristic is usually required in important kernel based classification techniques. For example, to be able to run efficiently “Radial basis function” (RBF) one needs to calculate efficiently the distance between two samples. The dissimilarity metric that we have used is the Earth Movers Distance (EMD). EMD has experimentally verified to capture well the perceptual notion of a difference between images. Piotr Indyk has suggested an embedding technique in which the un-normed EMD metric is embedded into a normed space, so that the distance between the two images is comparable to the distance between the two points which represent the embedding of the two images.

Talk about the embedding and its usages in the SVM.

* EMD approximation. Do we approximate the EMD and then do the embedding? Or we do the embedding within the EMD calculation?

### Earth Movers Distance

Earth movers distance (EMD) is a measure of the dissimilarity between two histograms. Descriptively, if the histograms are interpreted as two different ways of piling up a certain amount of dirt, the EMD is the minimal cost of turning one pile to other, where the cost is assumed to be the amount of dirt moved times the distance by which it moved.

EMD has been experimentally verified to capture well the perceptual notion of a difference between images. [14] Computing EMD is based on a solution to the well-known transportation problem. It can be computed as the minimal value of a linear program.

[EMD formal definition]

EMD naturally extends the notion of a distance between single elements to that of a distance between sets, or distributions, of elements when used to compare distributions with the same overall mass, the EMD is a true metric. In [15] a major hurdle to using EMD is its computational complexity (for an N-bin histogram).Various approximation algorithms have been proposed to speed up the computation of EMD.

The embedding of EMD given in [14] provides a way to map weighted point sets A and B from the metric space into the normed space , such that the  distance between the resulting embedded vectors is comparable to the EMD distance between A and B. The motivation of doing so is that the working with normed space is desirable to enable fast approximate Nearest Neighbors (NN) search techniques such as LSH and kdTree. A work conducted by Shirdhonkar and Jacobs in [9] has proposed a method for approximating the EMD between two low dimensional histograms using a new on the weighted wavelet coefficients of the difference histogram. The approximation is done by transforming the histograms to  space so that the distance between the two vectors in the wavelet domain is the EMD approximation. They have proven the ratio of EMD to wavelet EMD is bounded by constants. The wavelet EMD metric can be computed in time complexity.

## Features Transformation and Dimensionality Reduction

Feature transformation is a group of methods that create new features (predictor variables). The methods are useful for dimension reduction when the transformed features have a descriptive power that is more easily ordered than the original features. There are 2 main approaches for this task. One is *feature selection* and the other *is feature transformation*.

Dimensionality Reduction is a process of reducing the number of random variables taken into consideration in the learning and classification of Data. Ideally, the reduced representation should have a dimensionality that corresponds to the intrinsic dimensionality of the data. Reducing the dimensionality of the features vectors would not only simplify and rapid the learning and classification task but rather boosts the classification accuracy. Feature transformation technique is much more suitable to be implemented in our approach. In this work we have chosen to use two techniques applied in sequential manner in order to obtain the most efficient and linearly discriminative components, *Principle Component Analysis* (PCA) and *Linear Discrimination Analysis* (LDA) Technique.

PCA was invented in 1901 by Karl Pearson. It is a linear technique for dimensionality reduction, which means that it performs dimensionality reduction by embedding the data into a linear subspace of lower dimensionality. Although there exist various techniques to do so, PCA is by far the most popular (unsupervised) linear technique. PCA is an orthogonal linear transformation that transforms the data to a new coordinate system such that the greatest variance by any projection of the data comes to lie on the first coordinate (names the first principal component), the second greatest variance on the second coordinate, and so on. Each principal component is a linear combination of the original variables. All the principal components are orthogonal to each other, so there is no redundant information. The principal components as a whole form an orthogonal basis for the space of the data.

The full set of principal components is as large as the original set of variables. But taking the first few principal components will preserve most of the information in the data, and reduces the data dimensions.

PCA is an unsupervised technique and as such does not include label information of the data. The following example demonstrates the problem drawback: Imagine 2 cigar like clusters in 2 dimensions, one cigar has y = 1 and the other y = -1. The cigars are positioned in parallel and very closely together, such that the variance in the total data-set, ignoring the labels, is in the direction of the cigars. For classiﬁcation, this would be a terrible projection, because all labels get evenly mixed and we destroy the useful information. A much more useful projection is orthogonal to the cigars, i.e. in the direction of least overall variance, which would perfectly separate the data-cases.

LDA is closely related to PCA in that they both look for linear combination of variables which best explain the data. LDA explicitly attempts to model the difference between the classes of data. In this method, variability among the feature vectors of the same class is minimised and the variability among the feature vectors of different classes is maximised.

The LDA performs dimensionality reduction while preserving as much of the class discriminatory information as possible. Without going into the math, in order to find a good projection vector, we need to define a measure of separation between the projections. The solution proposed by Fisher is to maximize a function that represents the difference between the means, normalized by a measure of the within-class scatter.

Although, LDA assumes that the distribution of samples in each class is Gaussian and that we cannot prove that the handwritten letters are distributes in a Gaussian manner, we selected LDA as

Even though LDA is preferred in many application of dimension reduction, it does not always outperform PCA. In order to optimize discrimination performance in a more generative way, a hybrid dimension reduction model combining PCA and LDA is used in this work.

In this system, we use a flavour the dimensionality reduction process can be outlines as follows: the pre-processed feature matrix M projected into subspace  using PCA and then into the subspace  using LDA. In the PCA stage, the largest t eigenvalues are selected to create the PCA projection matrix.  is the number of eigenvalues which guarantee energy E is greater than 0.98. The data preservation value is calculates as seen in Equation 5 where is the ith eigenvalue

|  |  |
| --- | --- |
|  | Equation 5 |

The dimensionality of is much smaller that the dimensionality of M. At the second phase LDA is used to project to. The dimension of subspace is smaller than the sunspace by 1.

Why we have used both and what give us every method and how we did join them together to get the most out of both.

## Letters Classification

Classification is the most important stage in the recognition process. It is a classification of each unknown object into one of a finite set of categories or classes. These categories could be a whole word, word-parts, letter or even strokes. The problem that makes the recognition of online handwriting recognition difficult is variation of shapes of the characters resulting from writing habits, styles, and the social and educational level of the writer.

In our process we perform sequence classification in 2 stages: The first classification task is letter classification and the other is word-part classification. In the first we use Support vector machine, but in the later we use k-NN method.

The classification of letters is affects the segmentation task where the selected segmentation is the segmentation that resulted the best letter classification overall score.

### 4.8.2. Support Vector machine

SVM (Support Vector Machine) is a powerful tool for classification developed by Vapnik in 1992. It has originally been proposed for two-class classiﬁcation; however, it can be easily extended to solve the problem of multiclass classification. In this work we use one-to-one/ one-to-many method to do so. SVM is widely used in object detection and recognition, content-based image retrieval, text recognition, biometrics, speech recognition, etc. Without losing generality, we consider the 2 class classification problem. We give a brief introduction for the SVMs basics. SVM paradigm has a nice geometrical interpretation of discriminating one class from the other by a separating hyperplane with maximum margin. SVM can be altered to perform nonlinear classification using what is called the kernel trick by implicitly mapping the inputs into high-dimensional feature spaces. Given a training sample set  where a d-dimensional sample  is the class label and N is the size of the training set.

Support vector machine first map the data from the sample from the input space to a very high dimensional Hilbet Space.. The mapping  is implemented implicitly by a kernel function  that satisfies Mercer’s conditions. Kernels are functions that give the inner product of each pair of vectors in the feature space, such that. Then in the high dimensional feature space H, we try to find the hyperplane that maximizes the margin between the two classes, and minimizing the number of training error.

The decision function is as follows:

|  |  |
| --- | --- |
|  | Equation 6 |

Where

|  |  |
| --- | --- |
| And | Equation 7 |

Training as SVM is to find  that minimizes the following quadratic cost:

|  |  |
| --- | --- |
|  | Equation 8 |

where  a parameter chosen by the user. A large  corresponds to a higher penalty allocated to the training errors. This optimization problem can be solved using quadratic programming. [16]

Many implementation of kernels have been proposed, one popular example is the Gaussian Kernel:

|  |  |
| --- | --- |
|  | Equation 9 |
|  |  |

[Add a picture]

#### Gausian dynamic time warping using DTW as a metric

Claus Bahlmann et al. in [17] have proposed combining DTW and SVMs by establishing a new kernel which they have named Gaussian DTW (GDTW) kernel. They have achieved, by incorporating DTW in the Gaussian kernel function, a method to handle online handwriting input which is a variable-sized sequential data. Clearly, the assimilation of the DTW in the Gaussian kernel will result the following kernel function

|  |  |
| --- | --- |
|  | Equation 10 |
|  |  |

We should note that the DTW is not a metric, as the triangle inequality do not hold in many cases, thus the GDTW may miss some important properties as satisfying the Mercer’s condition. This would imply that the optimization problem in 6 may not convergent to the global minimum.

### 4.8.3. SVM on imbalanced Database

Classifiers generally perform poorly on imbalanced datasets because they are designed

to generalize from sample data and output the simplest hypothesis that best fits

the data, based on the principle of Occam’s razor. This principle is embedded in the

inductive bias of many machine learning algorithms including decision trees, which

favor shorter trees over longer ones. With imbalanced data, the simplest hypothesis is

often the one that classifies almost all instances as negative.

## Segmentation

## Overview

Demarcation point (sometimes referred as *Segmentation point* or *Junction Point*) is a junction point in the written stroke that is a vertical handler between letters in a word-part.

Character segmentation has long been a critical area of the OCR process. The higher recognition rates for isolated characters vs. those obtained for words and connected character strings well illustrate this fact. A good part of recent progress in reading unconstrained printed and written text may be ascribed to more insightful handling of segmentation.(Casey & Lecolinet, 1996)

* The segmentation recognition catch 22, recognition needs good segmentation and good segmentation needs good recognition.
* In fact, researchers have been aware of the limitations of the classical approach for many years. Researchers in the 1960s and 1970s observed that segmentation caused more errors than shape distortions in reading unconstrained characters, whether hand- or machine-printed. (the same source above)
* Our segmentation is independent of the writer font size.

Several segmentation techniques have been proposed in the literature for Arabic OCR. However correct and efficient segmentation of Arabic text is still considered a challenging and a fundamental problem even for offline printed text. Performing such task in an online manner for handwritten texts when the segmentation is being done while the word is being written is even a more challenging. Over segmentation and under-segmentation are the main problems such algorithms encounter. Later we will show how this problem is handled by our method.

The under-segmentation problem is usually as a result of two letters combinations that doesn’t contain a horizontal handler between them. For example, the following word-part لم (Lam i.e. Not) which is a combination of the letters ل (L) and م (M). Our approach overcomes this problem by broaden the set of letters classes to include such combinations. We will refer to such combinations as *hyper-letters*. By this, even if the segmentation process did not identify a demarcation point between these two letters, it will be recognized in the later process as a single letter.

Many segmentation techniques were proposed in the literature, mostly for segmenting English cursive handwriting. These methods can be categorized to 2 main approaches:

* 1. Dissection
  2. Recognition Based Segmentation

Dissection techniques try to learn the characteristic of the segmentation poin and try to find these features in a candidate point. For example, in English cursive script segmentation a common feature is that segmentation point has local minima in the upper or lower contour of the word. Another feature is the slope of the segmentation point is low. Some techniques doesn’t try to segment a word to its letters but to a graphemes, which are a combination of 2 or3 letters or is a part of a letter.

Methods considered here also segment words into individual units (which are usually letters). However, the principle of operation is quite different. In principle no feature-based dissection algorithm is employed. Rather, the image is divided systematically into many overlapping pieces without regard to content. These are classified as part of an attempt to find a coherent segmentation / recognition result. The main interest of this category of methods is that they bypass the segmentation problem: no complex "dissection" algorithm has to be built and recognition errors are basically due to failures in classification. In recognition-based techniques, recognition can be performed by following either a serial or a parallel optimization scheme. In the first case, e.g. [11], recognition is done iteratively in a left-to-right scan of words, searching for a "satisfactory" recognition result. The parallel method [48] proceeds in a more global way. It generates a lattice of all (or many) possible feature-to-letter combinations. The final decision is found by choosing an optimal path through the lattice.

### Our Approach

In this thesis we propose a novel approach for Arabic text segmentation which is performed while the writer is scribing the word part. This approach employs both, dissection techniques and recognition base segmentation.

In our progress segmentation approach, every given point on the sequence is tested to check whether it is an admissible candidate segmentation point. The candidate point is verified to be a real segmentation point is a latter phase.

Our segmentation algorithm exploits the following facts about segmentation points in Arabic handwritten text:

* Segmentation points lies in a horizontal segment of the written text as shown in fig .
* Segmentation point between connected characters lies at baseline as shown in fig.
* Segmentation points do not reside in loops.

Overview of the Ongoing Segmentation Algorithm:

The algorithm tries to identify horizontal segments. When a horizontal segment is determined, the median point is set as a checkpoint. The subsequence between that starts with the last critical checkpoint and the current checkpoint is sent to the recognition engine to find its most likely letter candidates. For every candidate the recognition algorithm gives a recognition score. This scoring will be used in a latter phase to determine, among others, the best segmentation point. In the first phase the point only set as candidate point. Then when the next horizontal segment is identified, its checkpoint passes the same procedure of recognition. The checkpoint its best candidate has the higher recognition score will be selected as the segmentation point. If the latter point is selected, the first checkpoint is ignored; else, if the first checkpoint is selected to be the segmentation point, the second checkpoint becomes a candidate point.

When a pen up event occurs the algorithm tries to determine the best segmentation point between the candidate point and the mouse up point. In cases where there is no previous candidate point, the algorithm tries to “merge” the last segmentation point and mouse up point. All decisions done by the segmentation algorithm when choosing between 2 candidate points is almost solely based on the recognition results of its corresponding subsequence,

To identify horizontal segments while the sequence is being is scribed we actually try to identify an “Opening horizontal fragment point” and it’s corresponding “Closing horizontal fragment point.

A point can be of 3 types:

1. Opening horizontal fragment point
2. Closing horizontal fragment point
3. Mid Horizontal point
4. Candidate segmentation point

For a given point, the condition for it to be an “Opening horizontal fragment point”

1. Low slope.
2. The subsequence between the point and the previous segmentation point or the previous candidate point, if the is any or the sequence beginning, contains enough information. Later on we will define what much information mean.
3. Close to the baseline. Later we will explain how we calculate the baseline while progressing.
4. If the simplified contour contains only 3 points, the angle between the two edges in not too obtuse.

For a given point to be an “Opening horizontal fragment point” it should hold one of the following conditions

1. The next inspected point has a high slope rate.
2. The direction of the sequence in left-to-right.

### Online Baseline detection

The baseline is the line that corresponds to the line upon which one would write on ruled paper. Arabic text is written along this baseline, with the majority of foreground pixels in this area. Letters junctions points are usually reside on the baseline.

Both online and offline Arabic text recognition techniques try to estimate the baseline location. In our approach we needed a method that approximate the baseline on every step during the ongoing recognition process. As previously mentioned, the baseline helps us to determine whether a checkpoint is a candidate point.

The role of the baseline is mainly to avoid nominating a point to become a candidate point such as

Our approach is to Produce a polygonal approximation to the skeleton of the word. Speciﬁc features of this skeleton are then used to determine

the baseline

To approximate the baselint, at least 2 segmentation points are needed. The

[33] M. Pechwitz and V. Maergner. Baseline estimation for arabic handwritten words. In Frontiers in Handwriting Recognition, pages 479–484, 2002.

### Segmentation Selection

## Word Part Recognition

* Using multi class SVM is not good for very large number of classes.
* When trying to select the closest Word part from the huge database of word parts, it is better to use k-NN methodologies.
* In this case, we need a data structure that quickly retrieves the nearest neigbour.
* We will use kd-tree or LSH for this task. We will select Kdtree as we have managed to reduce the dimensionality of the data to a small number of dimension without losing precision.
* How our selection is done?

### KdTree/LSH

Maybe here we should embed the DTW in the Euclidean space, because we know that DTW is better than EMD. How can we embed DTW to the Euclidean space?

# Experimental Details and Results

# Summary and Conclusions

In this thesis we have proposed and implemented a system to recognize Arabic online handwriting script. Our approach successfully tackled many inherent difficulties in recognizing Arabic script such as segmentation of word parts, non-horizontal writing style, position-depended letter shaping and large word-part dictionary which has been overcome upon by combining both the analytic and the heuristic approach. We compared the Shape context to the MAD descriptor and showed that the MAD descriptor attained higher recognition rate. A novel online segmentation technique was introduced and used in the analytic phase. The Linear EMD embedding technique was used as a metric in the classification algorithms and showed promising results in measuring letters and word parts grasped dissimilarity. A contemporary composition of PCA and LDA, based on the method previously proposed in [18], was applied in the system and achieved qualitative dimensionality reduction. This approach can be used to overcome the “curse of dimensionality” in other fields of pattern recognition. We have integrated fast data retrieval methods to boost the performance of the system. The system demonstrated superior high recognition precision and low response time.

The system has several limitation and assumptions that are not always true in the field of Arabic handwriting; first, the system does not regard additional strokes, which can be used to improve the classification results; second, it assumes that letters can’t be written in more than one stroke which is a very reasonable assumption however not always true. Our future work will focus on waiving these limitations. Also we plan to test this approach on other cursive handwriting styles of languages other than Arabic.

## Contribution to the field

dsadasdasdsadsadsadsadsadas

## Future Directions

*In future direction we would like to rid of all the limitation mentioned in the limitation section.*

We have several directions we would like to develop. In this work we did not consider the additional strokes written and we tried to recognize only the main body of the written text. In future work we would like to be able to recognize the whole letter and use the additional strokes to be able to identify the exact letter written.

The following enhancement will be introduced to the process in a future work: 1. Since the embedding is done to the L1 space, we will try to use L1 dimensionality reduction techniques such as L1-PCA and L1-LDA. 2. We can add Metric Learning techniques such as LMNN to improve our recognition rate. 3. In this work all shapes that belong to the same letter are ascribed to the same cluster although it might look totally different, in our future work we will create more that cluster to the same letter to represent the main shapes that are common to describe a letter.[??] 4. The use of more advanced clustering techniques to reduce the number of samples in the training set and which will accelerate the classification process.

Also, we are planning to develop a word completion algorithm based on our ongoing recognition technique. The word completion feature is actually will suggest the user to automatically complete the word the user has intended to write, based on the first letters written by him.

# References

|  |  |
| --- | --- |
| [1] | L. Harmon, "Automatic recognition of print and script," *Proceedings of the IEEE,* pp. 1165 - 1176 , 1972. |
| [2] | S. Mozaffari and H. Soltanizadeh, "ICDAR 2009 Handwritten Farsi/Arabic Character Recognition Competition," *2009 10th International Conference on Document Analysis and Recognition,* pp. 1413-1417, 2009. |
| [3] | M. Al-Ammar, R. Al-Majed and H. Aboalsamh, "Online Handwriting Recognition for the Arabic Letter Set," *Recent Researches in Communications and IT.* |
| [4] | S. D. Connell, "Online Handwriting Recognition Using Multiple Pattern Class Models," Michigan, 2000. |
| [5] | R. Saabni and J. El-sana, "Efficient Generation of Comprehensive Database from Online Arabic Script Recognition". |
| [6] | I. A. Jannoud, "Automatic Arabic Hand Written Text Recognition System," *American Journal of Applied Sciences,* pp. 857-864, 2007. |
| [7] | F. Biadsy, R. Saabne and J. El-Sana, "Segmentation-Free Online Arabic Handwriting Recognition," 2010. |
| [8] | L. Harmon, "Automatic recognition of print and script," *Proceedings of the IEEE,* vol. 60, no. 10, pp. 1165 - 1176 , 1972. |
| [9] | S. Shirdhonkar and D. W. Jacobs, "Approximate earth mover's distance in linear time," University of Maryland, Maryland, 2008. |
| [10] | A. Gionis, P. Indyk and R. Motwani, "Similarity search in high dimensions via hashing," 1999. |
| [11] | D. Douglas and T. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *The Canadian Cartographer,* 1973. |
| [12] | S. Belongie, J. Malik and J. Puzicha, "Shape Matching and Object Recognition Using Shape Contexts," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 24, p. 509–521, 2002. |
| [13] | R. Saabni and A. Bronstein, "The Multi Angular Descriptor: new binary and gray images descriptor for shape recognition". |
| [14] | I. Poitr and N. Thamper, "Fast Image Retrieval via Embeddings," Third International workshop on Statistical and Computional Theories of Vision, Nice, France, 2003. |
| [15] | "The Earth Mover's Distance," The University of EDINBURGH, [Online]. Available: http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL\_COPIES/RUBNER/emd.htm. [Accessed 12 April 2012]. |
| [16] | J. Sadri, C. Y.Suem and T. D. Bui, "Application of Support Vector Machines for Recognition of Handwritten Arabic/Parsian Digits". |
| [17] | C. Bahlmann, B. Haasdonk and H. Burkhardt, "Online Handwriting Recognition with Support Vector Machines - A Kernel Approach," *Workshop on Frontiers in Handwriting Recognition (IWFHR),* pp. 49-54, 2002. |
| [18] | R. Saabni and A. Bronstein, "Fast Key-Word Searching Using'BoostMap'based Embedding," 2012. |
| [19] | T. M. Rath and M. Manmatha, "Word Image Matching Using Dynamic Time Warping". |
| [20] | P. Senin, "Dynamic Time Warping Algorithm Review," Honolulu, USA, 2008. |
| [21] | S. Salvador and P. Chan, "FastDTW: Toward Accurate Dynamic Time Warping in Linear Time and Space," Melbourne. |
| [22] | C. A. Ratanamahatana and E. Keogh, "Three Myths about Dynamic Time Warping Data Mining". |

# 4.7. Appendix

## 4.7.1. Wavelets

* Limitations of Traditional basis expansion.