Historical handwriting representation model dedicated to word spotting application

Peng Wang

UNIVERSITÉ JEAN MONNET SAINT ÉTIENNE

DOCTORAL SCHOOL SCIENCES INGÉNIEUR SANTÉ

P h. D. T H E S I S

to obtain the title of

Ph.D. in Computer Science

Historical Handwriting Representation Model Dedicated to Word Spotting Application

defended by

Peng WANG

Thesis advisors: Véronique ÉGLIN
Christine LARGERON
Christophe GARCIA
Antony MCKENNA

defended on November 18, 2014

Jury :

Reviewers :	Rolf INGOLD	-	DIVA, Université of Fribourg, Switzerland
	Laurent WENDLING	-	LIPADE, Université Paris Descartes, France
President :	Alain TRÉMEAU	-	LAHC, Université Jean Monnet, France
Examinator :	Josep LLADÓS	-	CVC, Universitat Autònoma de Barcelona, Spain
Advisors :	Véronique ÉGLIN	-	LIRIS, INSA de Lyon, France
	Christine LARGERON	-	LAHC, Université Jean Monnet, France
	Christophe GARCIA	-	LIRIS, INSA de Lyon, France
	Antony MCKENNA	-	IHPC, Université Jean Monnet, France

To my parents

Acknowledgments

This thesis is the result of three years intense work which would not have been possible without the support of many. I would here like to express my thanks to people who have been very helpful to me during the course of this work.

First of all, I would like to take this opportunity to gratefully acknowledge the wholehearted supervision of my main advisor Dr. Véronique Eglin during this work. Her dedication, skillful guidance, helpful suggestions and constant encouragement made it possible for me to deliver a dissertation of appreciable quality and standard. She has inspired me to tackle difficult problems with her encouragement and enthusiasm. It was a pleasure to work in such a friendly atmosphere that she created.

I would also like to thank my other advisors Prof. Christine Largeron, Prof. Antony McKenna and Prof. Christophe Garcia for providing me with valuable guidance and support at various stages of my work. They were so helpful that I could not finish my PhD without them.

Special thanks to Prof. Rolf Ingold and Prof. Laurent Wendling for accepting to review my thesis and providing me with valuable comments and suggestions to improve the quality of this thesis. In addition, I would like to thank my committee members for their valuable input that helped improve this work.

My cordial appreciation extends to Dr. Josep Lladós and Dr. Alicia Fornés for their supervision and hospitality during my research stay in Computer Vision Center, Barcelona. Without their sincere and valuable support and constructive suggestions, my work would not have been so fruitful. Moreover, my stay at CVC would not have been such a pleasurable one without the presence of friendly colleagues who helped me out in one way or another and exchanged useful views from time to time.

I owe special thanks to my friends Yuyao Zhang and Yann Leydier for helping me get through difficult times, and for all the support, camaraderie and caring that they provided.

I would also like to express my deepest gratitude to the Allocations Doctorales de Recherche project of the Region Rhôe Alpes for financially supporting the study and making it possible for me to pursue my research in a prestigious institution of great repute.

Finally and most importantly, I am forever indebted to my parents for their support when it was most required, and encouraging me to concentrate on my study. Without their help and encouragement, this study would not have been completed.

Handwriting Representation Model Dedicated to Word Spotting Application

Abstract: As more and more documents, especially historical handwritten documents, are converted into digitized version for long-term preservation, the demands for efficient information retrieval techniques in such document images are increasing. The objective of this research is to establish an effective representation model for handwriting, especially historical manuscripts. The proposed model is supposed to help the navigation in historical document collections. Specifically speaking, we developed our handwriting representation model with regards to word spotting application. As a specific pattern recognition task, handwritten word spotting faces many challenges such as the high intra-writer and inter-writer variability. Nowadays, it has been admitted that OCR techniques are unsuccessful in handwritten offline documents, especially historical ones. Therefore, the particular characterization and comparison methods dedicated to handwritten word spotting are strongly required.

In this work, we explore several techniques that allow the retrieval of single-style handwritten document images with query image. The proposed representation model contains two facets of handwriting, morphology and topology. Based on the skeleton of handwriting, graphs are constructed with the structural points as the vertexes and the strokes as the edges. By signing the Shape Context descriptor as the label of vertex, the contextual information of handwriting is also integrated. Moreover, we develop a coarse-to-fine system for the large-scale handwritten word spotting using our representation model. In the coarse selection, graph embedding is adapted with consideration of simple and fast computation. With selected regions of interest, in the fine selection, a specific similarity measure based on graph edit distance is designed. Regarding the importance of the order of handwriting, dynamic time warping assignment with block merging is added. The experimental results using benchmark handwriting datasets demonstrate the power of the proposed representation model and the efficiency of the developed word spotting approach.

The main contribution of this work is the proposed graph-based representation model, which realizes a comprehensive description of handwriting, especially historical script. Our structure-based model captures the essential characteristics of handwriting without redundancy, and meanwhile is robust to the intra-variation of handwriting and specific noises. With additional experiments, we have also proved the potential of the proposed representation model in other symbol recognition applications, such as handwritten musical and architectural classification.

Keywords: Comprehensive representation model, word spotting, graph-based, shape context

Contents

1 Introduction **1**
- 1.1 Objective . . . 2
- 1.2 Challenges . . . 4
- 1.3 General structure of the dissertation . . . 6

2 Handwritten Document Image Description and Representation **9**
- 2.1 Handwriting description and characterization . . . 9
  - 2.1.1 Low-level features . . . 10
  - 2.1.2 High-level features . . . 22
- 2.2 Handwriting representation . . . 28
  - 2.2.1 Bitmap representation . . . 29
  - 2.2.2 Shape coding . . . 29
  - 2.2.3 Bag-of-visual-word model . . . 31
  - 2.2.4 Graph-based model . . . 32
  - 2.2.5 Blurred shape model . . . 32
  - 2.2.6 Biologically inspired model . . . 33
  - 2.2.7 PDF-based representation . . . 35

3 Comprehensive Representation Model **37**
- 3.1 Preprocessing . . . 38
  - 3.1.1 Noise elimination . . . 39
  - 3.1.2 Binarization . . . 40
  - 3.1.3 Enhancement . . . 41
  - 3.1.4 Line segmentation and baseline detection . . . 42
- 3.2 Feature selection and extraction . . . 46
  - 3.2.1 Contour and skeleton based decomposition . . . 46
  - 3.2.2 Structural point detection . . . 56
  - 3.2.3 Shape context . . . 59
  - 3.2.4 Point feature performance evaluation (SIFT, SURF, Shape Context) . . . 63
- 3.3 Graph-based representation model . . . 65
  - 3.3.1 Graph-based approaches for pattern recognition . . . 65
  - 3.3.2 Graph-based representation model . . . 68
  - 3.3.3 Graph representation of handwritten document images . . . 70
  - 3.3.4 Approximate graph edit distance . . . 74

4 Word Spotting Approach Based on a Comprehensive Model **79**
- 4.1 State of the art . . . 79
  - 4.1.1 Appearance-based and structure-based approaches . . . 80

4.1.2 Learning and learning-free approaches . . . 85
4.1.3 Segmentation-based and segmentation-free approaches . . . 88
4.2 Our contribution . . . 93
4.2.1 Introduction . . . 93
4.2.2 Coarse selection . . . 96
4.2.3 Fine selection . . . 108
4.3 Evaluation . . . 127
4.4 Other potential applications . . . 131
4.4.1 Symbol classification . . . 131
4.5 Conclusion . . . 132

5 Conclusion and Perspectives 135
5.1 Conclusion . . . 135
5.2 Perspectives . . . 137
5.2.1 Segmentation-free . . . 137
5.2.2 Multi-writing styles . . . 138

Bibliography 139

Chapter 1

Introduction

Contents

1.1 Objective 2
1.2 Challenges 4
1.3 General structure of the dissertation 6

Nowadays, there are tens of thousands of ancient manuscript documents preserved in the libraries and museums. A great range of people are interested in these valuable resources, especially literary experts, historians, humanity scholars etc. The best way to make these corpora accessible to everyone and to protect them from being destroyed by a frequent handling is digitization. Today, as more and more libraries and museums have transformed their historical documents into digital format and made them available via electronic media, a new issue concerning how to efficiently index the digitized documents has been raised. There are many levels of indexation, from the name of the author and the title of the book to a full text transcription and its critical edition. Most users expect to access the digitized heritage through simple interfaces similar to the most common web-search engines, i.e. through a textual query to quickly find the document of interest. Nonetheless, there is still lack of such a system able to read accurately ancient manuscripts. Even though some collections have been manually annotated, it cannot be extended to all collections, which are substantial, since the manual document transcription is a very expensive and tedious task. The sensible thing is to create an automatic way to build document indexes for navigation. However, an efficient access based on automatic handwriting recognition to such collections requires that the system must overcome a large number of difficulties.

Currently, OCR (Optical Character Recognition) systems have achieved huge success on printed text documents. Nevertheless, they are not qualified to process handwritten documents due to its inability to handle the irregularity of handwriting in an unconstrained scenario.

The target of the thesis is to develop a representation model of handwriting so that it can be used to realize an efficient indexing system for digitized historical manuscripts. As part of the ANR CITERE research project (Circulations, territoires et réseaux en Europe de l'âge classique aux Lumières) , initiated by Pierre-Yves Beaurepaire of Université de Nice Sophia-Antipolis and Antony McKenna of Université Jean Monnet Saint Etienne, the thesis is under a joint supervision between two academic partners, one from Université Jean Monnet Saint Etienne with

Professor Antony McKenna (Institut d'Histoire de la Pensée Classique, Institut CLAUDE LONGEON, CNRS UMR 5037) and Professor Christine Largeron (Laboratoire Hubert Curien, CNRS UMR 5516), and the other one from INSA de Lyon with Professor Christophe Garcia and Associate Professor Véronique Eglin (Laboratoire d'InfoRmatique en Image et Systèmes d'information, CNRS UMR5205). The goal of the project is to realize online comments from both scientific and cultural aspects on the whole heritage concerning Huguenot correspondences and clandestine philosophical manuscripts during the period of 1685-1789. It is a multidisciplinary project that has a fundamental component in the history of classical thought. It focuses on the role that written communication, such as letters and manuscripts of philosophical scholars, plays in the development of the Renaissance and the formation of the philosophical spirit of that time. Fig. 1.1 gives one example of the correspondences during this period.

1.1 Objective

The thesis aims to develop original methods for the characterization and retrieval of historical manuscripts in the context of a large-scale corpus (over 120,000 documents). This thesis is meaningful in the sense that the objects included in the study cannot be modeled by standard representations due to the presence of highly heterogeneous and composite content. It therefore requires the development of flexible and adaptive methods dedicated to the specific content. The proposed method is supposed to handle the variability of the scriptures and to be robust to noises and distortions appearing in the historical document images.

The objective of our work can be divided into two parts. First part of this work should be concerned with the representation of historical handwriting that is necessary for all handwriting analysis applications, such as handwriting retrieval and recognition, writer identification and so on. There are many conventional approaches to handwriting analysis related applications. All of them reveal the importance of the description of handwriting. Our work is aimed at developing a model that can comprehensively describe the handwriting without any constraints. The complete description of handwriting should be capable to tackle the specific problems related to historical documents, such as aging of materials and impregnating ink traces, irregular ink traces, folding, tearing, breakage and other damages on paper. The initiation of the proposed representation model is designed for Latin languages, but it is not restricted to Latin languages.

In the second part, we are focusing on how to use the representation model developed in the first part to address handwritten word spotting problem. As a specific pattern recognition task, handwritten word spotting is defined as finding keywords in handwritten document images. For historical documents, it offers a solution when searching information in digital libraries. There are two main categories of word spotting problems depending on the representation of the query. One is called Query-By-String (QBS), which uses an arbitrary string as input. It

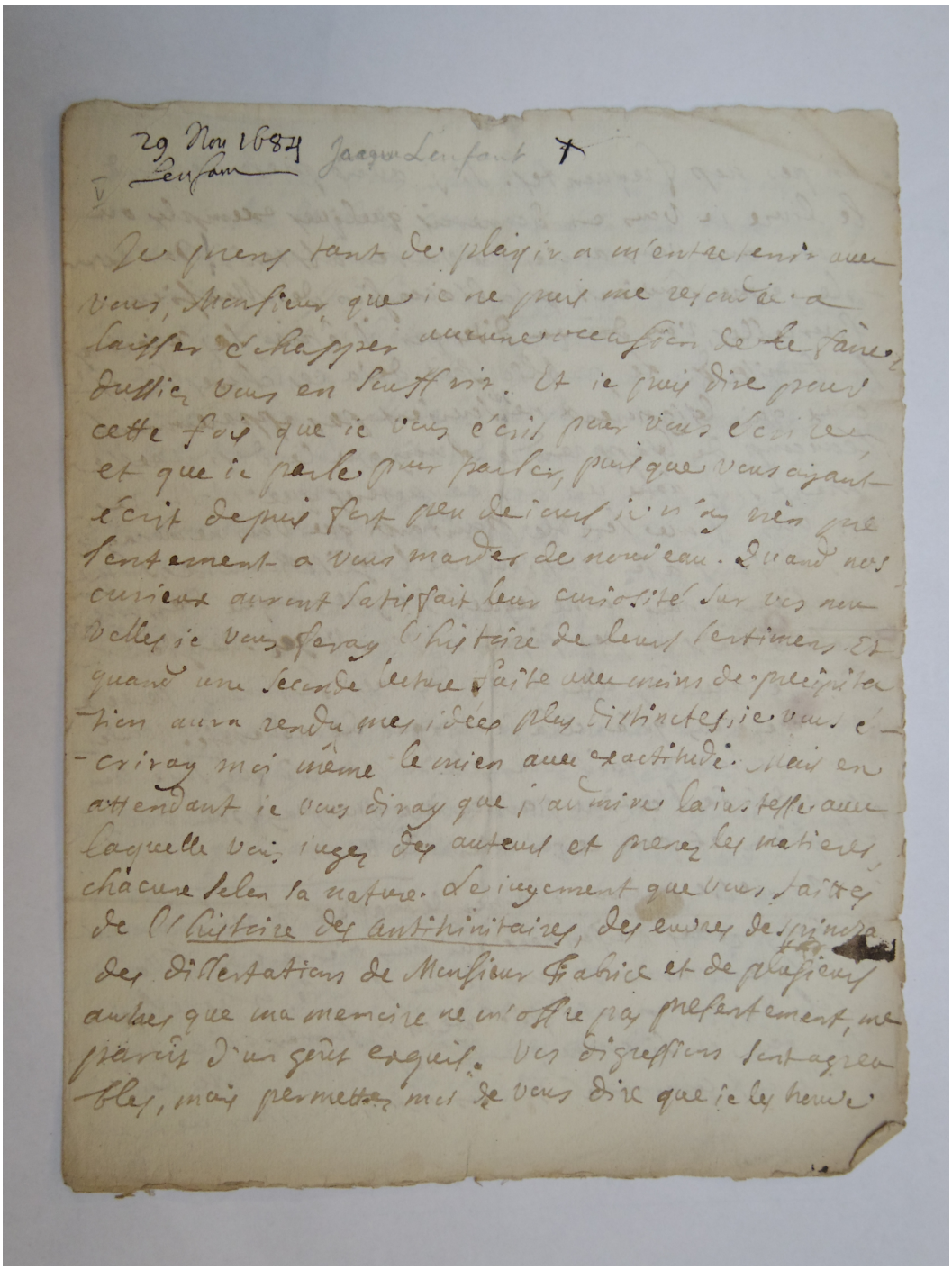


Figure 1.1: One example of the French philosopher Pierre Bayle's correspondence in the 17th century (shelf-mark: Thott 1208 collected in the Royal Library in Copenhagen, Denmark)

typically requires a large amount of training materials since character models are learned a priori and the model for a query word is build at runtime from the models of its constituent characters. The other one is named Query-By-Example (QBE). The input is one or several exemplary images of the queried word. Considering the specific corpora (without groundtruth) and the practical problems that we are facing, we concentrate on the QBE word spotting. Since a suitable similarity measure is as important as the complete characterization of handwriting in the solution of word spotting problems, in this part, we mainly work on developing innovative similarity measures based on the proposed representation model. In a word, one of our goals is to offer users a set of word spotting solutions depending on the situations and the requirements of users.

In the view of value, this thesis opens an access to the content analysis of handwritten images, promoting new forms of navigation (including the visualization of damaged or untranscribed parts and the location of a portion of the manuscript collection in its original context). This work will help fill the gap in analyzing images of large collections of manuscripts. By targeting the development of an innovative "technical tool" with the function of removing barriers preventing the unveiling of true valuation of large corpora of modern manuscripts, this work will find opportunities in the development of written corpora and archives. This work will be very attractive for all scholars who want a suitable tool to read and study digitized documents in major European libraries.

1.2 Challenges

As a specific computer vision task, besides standard difficulties, handwriting analysis poses many particular challenges. One of the biggest challenges for document image analysis systems is to handle the great variability of handwriting with a stable representation model. It has been observed that one's handwriting can be influenced by many factors, such as cultural background, education, habits, physical conditions and so on. It reflects somewhat the personality by carrying the individual characteristics of each person. The same content written by different persons can be varied (See Fig.1.2). Even though the same person writes the script consistently, it may still be different under certain circumstances. Many historical collections are the work of one author, which limits the amount of variations in the writing. One example of such collections is the George Washington collection. The techniques presented in the dissertation assume that the analyzed document collection was produced by a single writer or that the handwriting styles of the analyzed documents written by different persons are very similar.

Besides handwriting variations, we have to face the problems caused by the bad quality of images. There are various reasons that can result in such problems, such as the digitization from reproduction (see Fig. 1.3). Some libraries already have copies of various documents on microfilm. Handling original manuscripts is expensive and has to be performed by trained professionals, so scanning is often done from micro-

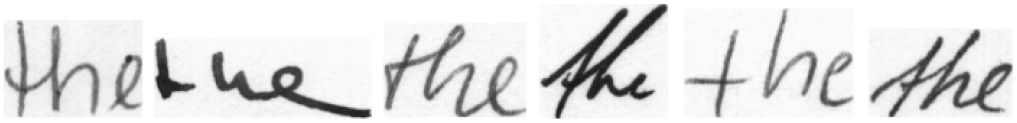


Figure 1.2: Variability of handwriting: the same word written by different persons

film, which does not require the same level of care and can be automated. However, this additional reproduction step adds noise and should therefore be avoided.

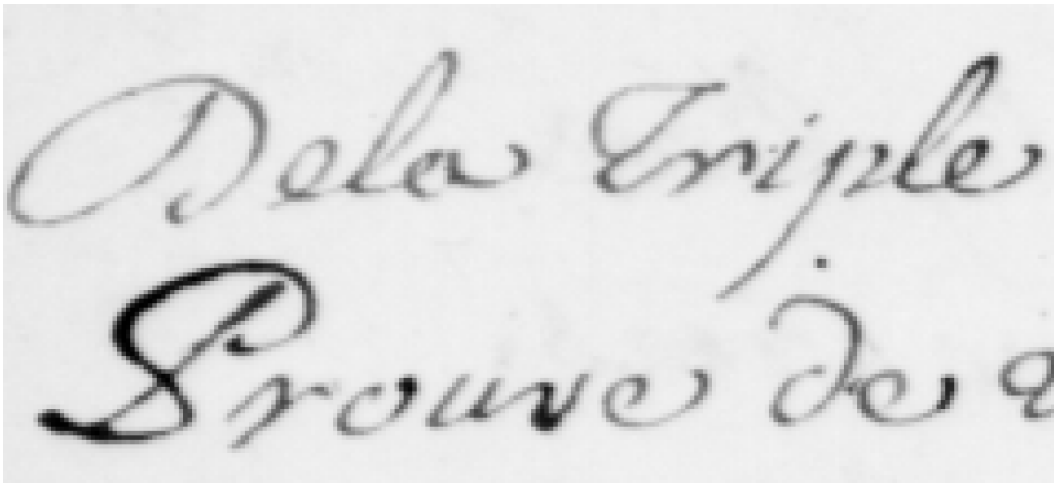


Figure 1.3: Digitization of the microfilm with low resolution (from French philosopher Pierre Bayle's collection)

Although space is becoming less of a concern, some digitization efforts have relied on lossy image compression formats, such as JPEG, to store scanned documents. Depending on the type of compression that is used, various artifacts may be introduced. Fig. 1.4 shows one example of the distortion caused by image compression.

Moreover, regarding the historical documents, aging is an additional problem. It is a common phenomenon that the handwritten shapes are broken due to the degradation of ink (see Fig. 1.5). There are still other distortions and noises related to handwritten document images. Details will be introduced in Chapter 3. Most of these noises and distortions can affect or even damage the handwritten shapes. How to cope with this problem is also one of difficulties to overcome in our work.

(a) (b)

Figure 1.4: Example of compression artifacts. (a) edge image calculated from a lossless TIFF-compressed image (b) edge image calculated from a lossy JPEG-compressed image. Lossy JPEG compression introduces artifacts that can complicate image processing. The images show the influence of the artifacts on edge detection output [Rath 2005]

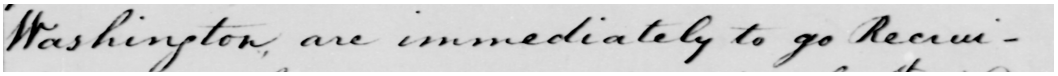


Figure 1.5: Example of broken manuscripts caused by ink degradation (from the George Washington collection)

1.3 General structure of the dissertation

The dissertation is divided into five chapters, which include the introduction of the PhD, the state of the art on handwriting description and representation, our comprehensive handwriting representation model, a set of word spotting approaches that we develop based on the proposed model and the evaluation of our propositions.

In chapter 2, we present an in-depth overview of the features and representations that have been used for handwriting document analysis, concerning different applications. As a subfield of computer vision and pattern recognition, many conventional descriptors and representation models developed for other computer vision and pattern recognition applications can be directly adapted in handwritten document analysis. However, concerning the specificity of document images, there are also some features and characterization methods exclusively designed for document image analysis. We have conducted a thorough study of these works by analyzing the strengths and weaknesses of each feature and model as well as the suitable situations to apply them. Finally, we summarize all of them in chapter 2.

Chapter 3 is devoted to the comprehensive handwriting representation model we propose. Unlike conventional approaches which describe handwritten patterns in the view of only appearance or only structure, we have introduced a graph-based representation for modeling the characteristics of handwriting with the consideration of both topological and morphological facets. As is known to all, the graph is a powerful tool for preserving the structural properties of the object while representing it. It is an ideal model for depicting handwriting, since the structural information is a critical factor for distinguishing handwritten shapes. Moreover, we introduce morphological information concerning handwriting into our model by using Shape Context descriptor to describe the contour of shapes. In this way, two complemen-

tary aspects of handwriting, topology and morphology, are integrated together in our hybrid model to give a complete description of handwritten documents.

Afterwards, several word spotting solutions concerning different scenarios (segmented words or entire pages) are presented in chapter 4. With the same intention of exploiting two facets (topology and morphology) of handwriting, we have developed different approaches. To compare word images, we have designed two novel methods based on the proposed representation model. One is to respectively calculate the distance between two word images in terms of graph part and Shape Context part. Then take the final distance as the weighted sum of two parts. Linear discriminant analysis technique is used for automatically learning the weights. The other proposition is to fuse the graph representation and Shape Context description at an early stage by considering the Shape Context as the label of vertexes of the graph. Subsequently, an approximate graph edit distance based on dynamic time warping assignment is calculated as the final distance between two word samples. Furthermore, in order to make the approach more efficient with regard to the processing time as well as the constraints of segmentation, we adapt a coarse-to-fine scheme. In the coarse selection, a sliding window is employed to scan the text lines. An graph embedding approach based on the statistics of vertex label and the relationship among vertexes is adapted, so that the graph-based representation can be compared much more quickly. Moreover, another proposition for the comparison between the query and the sliding window is based on a combination of low-level and high-level descriptions of the image, i.e. the projection profile, the upper/lower border profile, the orientation distribution, loops and ascenders/descenders. The performances of proposed approaches are evaluated on different benchmark databases with the comparison of the state of the art. Results have verified the effectiveness of our propositions.

Finally, we summarize the entire PhD study and give perspectives for the future in chapter 5.

Chapter 2

Handwritten Document Image Description and Representation

Contents

2.1 Handwriting description and characterization 9
2.1.1 Low-level features 10
2.1.2 High-level features 22
2.2 Handwriting representation 28
2.2.1 Bitmap representation 29
2.2.2 Shape coding 29
2.2.3 Bag-of-visual-word model 31
2.2.4 Graph-based model 32
2.2.5 Blurred shape model 32
2.2.6 Biologically inspired model 33
2.2.7 PDF-based representation 35

Working on images in their original raw format, e.g. a pixel matrix produced by an upstream processing step, is often difficult and inefficient. Representing images in terms of features allows a more compact and descriptive characterization of images with limited redundancy. The right features and the appropriate representation associated with the best adapted similarity measure are considered as fundamentals of handwriting analysis.

2.1 Handwriting description and characterization

In object/image recognition, a region can be described using a scalar or a set of scalars based on the geometric properties of the object. Such scalars or sets of scalars are called descriptors since they describe objects recognized by the artificial vision system.

Generally, features used in handwriting analysis can be classified into two categories, as Koerich et al. [Koerich 2006] indicate. They are respectively high-level features and low-level features. High-level features are more abstract and systemic features, such as ascenders, descenders and loops, which are typically more concerned with the object as a whole, or larger components of it. To an extent, high-level features contain semantic knowledge that low-level features do not comprise.

They are usually powerful and useful, especially for recognition tasks, because of the linguistic information carried by them, but they are less robust and sensitive to distortion. These features do not work very well on unconstrained handwriting, because the results of high-level feature extraction tend to be erroneous due to the large shape variations in the natural cursive handwriting. On the other hand, low-level features are the ones that concentrate more on individual components, provide details rather than the overview. The features such as entropy, profiles, zernike moments, point features are low-level. They are usually less informative but more reliable [Hu 1997]. The definitions of high-level and low-level features are relative. The same feature can be applied to different scopes. Moreover, depending on the arrangement of features in the representation stage, the drawbacks of each feature category can be compensated.

In this chapter, we make a thorough study of the state of the art on features used in handwriting analysis and recognition in the first part. The review is made in terms of categories. Furthermore, since the representation of images plays an equally critical role as features in the image reconstruction and recognition, the second part is devoted to an overview of the representations that have been used for modeling handwriting in the context of different handwriting applications.

2.1.1 Low-level features

2.1.1.1 Geometric features

Each of the features described here may be expressed using a single number. Some of them have been used to quickly determine the coarse similarity between word images in the context of word spotting. Some of them have been used together with probability distribution function to formalize the individuality of writing style. Depending on the applications, the following scalar features are introduced in two categories. One is for handwriting recognition, retrieval and word spotting. The other includes writing style classification, writer identification, and signature verification.

For the first category, common geometric features are usually collected in view of the entirety of the given image or sliding window (bounding box):

- Height h of the image in pixels
- Width w of the image in pixels
- Aspect ratio w/h
- Area $w \cdot h$

While the aspect ratio and area features are redundant, their distributions differ from those of the height and width features.

Secondly, the geometric features adapted in the tasks such as writer identification are pixel-wise. Since writer identification focuses on summarizing the general characteristics of handwriting instead of understanding a single pattern (letters,

characters or words), the local geometric features are usually calculated all over the handwriting and then are represented in a statistical manner. Next, we present some features that are widely used in writer identification.

- Orientation of the fragment

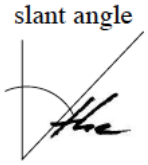


Figure 2.1: Slant angle [Marti 2001]

The most prominent visual attribute of handwriting that reveals individual writing style is slant (see Fig. 2.1), which is known as a very stable personal characteristic [Maarse 1983].

The slant can be described by several attributes. The orientation of local fragments is one of the attributes [Bulacu 2007]. These fragments can be computed either on the contour or on the skeleton of the handwriting. Take the orientation of the contour fragment for example. It is determined by two contour pixels taken a certain distance apart (see Fig. 2.2(a)) and the angle that the fragment makes with the horizontal is computed using Eq. 2.1.

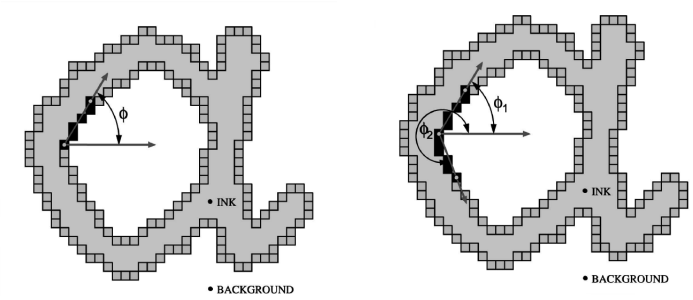
$$\phi = arctan(\frac{y_{k+\varepsilon} - y_k}{x_{k+\varepsilon} - x_k}) \tag{2.1}$$

where ε is the parameter that controls the length of the analyzing contour fragment.

The disadvantage of using contour fragments instead of skeleton ones to compute the orientation is the influence of the ink-trace width. However, if we use the probability distribution function of such contour-based orientations to characterize the handwriting, this drawback of using contour representation can be eliminated.

- Hinge

In order to capture the curvature of the ink trace, which is very discriminatory between different writers, besides orientation, hinge feature is designed by M. Bulacu et al. [Bulacu 2007]. The central idea is to consider not one, but two contour fragments attached to a common end pixel and, subsequently, use the combination of the orientations (ϕ_1, ϕ_2) of the two legs of the obtained "contour-hinge" (see Fig. 2.2(b)).



(a) (b)

Figure 2.2: Schematic description of the extraction methods of (a) contour fragment orientation and (b) contour hinge [Bulacu 2007]

The classifications in terms of applications are not strictly defined. Some features can be adapted for different applications. For instance, the orientation distribution of ink trace has also been used for image characterization [Journet 2008] and writing classification [Eglin 2007], while it will also be exploited in our contribution to handwritten word spotting in chapter 4.

2.1.1.2 Holistic features

Given a word/line image, a series of holistic features can be extracted. A set of preprocessing is necessarily carried out, e.g. the skew and slant corrections, adjusting the upper/lower baseline to a standard position, horizontally scaling, so as to be robust against handwriting variations. Suppose that pixel intensity values in an image $I \in R^{h \times w}$ are referred to as $I(r, c)$, where r and c indicate the row and column index of the pixel.

- Projection profile

  Projection profile captures the distribution of the ink along one of the two dimensions in the image. A vertical projection profile is computed by summing the intensity values in each image column separately:

$$pp(I, c) = \sum_{r=1}^{h} (255 - I(r, c)) \tag{2.2}$$

- Word profile (upper/lower border profile)

  Upper/lower word profile features are computed by recording - for each image column - the distance from the upper (lower) boundary of the word image to the closest "ink" pixel. Ink pixels are determined by a thresholding algorithm that classifies pixels into the categories ink and paper. If an image column

does not contain ink, the feature value is computed by linear interpolation between the two closest defined values.

$$up(I,c) = \begin{cases} undefined & if\ \forall r \quad is\_ink(I,c,r) = 0 \\ argmin_{r=1,\cdots,h}(is\_ink(I,c,r) = 1) & otherwise \end{cases} \tag{2.3}$$

$$lp(I,c) = \begin{cases} undefined & if\ \forall r \quad is\_ink(I,c,r) = 0 \\ argmax_{r=1,\cdots,h}(is\_ink(I,c,r) = 1) & otherwise \end{cases} \tag{2.4}$$

- Transition profile

  To capture part of the inner structure of the handwriting, transition profile records, for every image column, the number of transitions from the background to an "ink" pixel. Similarly, the number of black pixels between the upper and lower border.

- Gravity center

  The center of gravity is the average coordinate, weighted with the density (in the case of binary image, the density is 1 or 0.). It has been used in [Marti 2001] as a sequential feature. For each column, a gravity center is computed and recorded as Eq. 2.5.

$$gc(I,c) = \frac{1}{h}\sum_{r=1}^{h} r \cdot I(r,c) \tag{2.5}$$

All the features presented above require that the preprocessings be accurate and precise. Moreover, due to their definitions, these holistic features, except projection profile, cannot be directly applied on color/grey-level images but on binary images. Therefore, all of these features are not reliable when the quality of the images is bad, especially with the degradation of ink.

2.1.1.3 Frequency-based features

Frequency-based features are usually used in writer identification and document classification applications. It is because the texture differences in the spatial domain are not able to discriminate writing styles. They are affected by the assignments of writing samples. The same writer's different writing samples might lead to totally different textures. The differences of writing styles reside in the curvature, interlinkage and slant of writing traces. These cannot be revealed in the spatial domain texture. There are many ways of transforming the image from the spatial domain to the frequency domain. In the following, we list the most common approaches.

Gabor-filtering based features Multi-channel Gabor filtering is inspired by the psycho-physical findings of the cortex that has a set of parallel and quasi-independent mechanisms usually modeled by band-pass filters. As a well-known texture analysis technique, Gabor filtering has been applied in the fields of writer

identification [Bensefia 2005, Shahabi 2009] and document classification [Eglin 2007] and has achieved outstanding performance. There is a big family of two-dimensional Gabor functions that are designed to model the properties of simple cells in the visual cortex regarding the specific information to extract. The configurations of parameters specify the functions of different parts involved in the system and the selection of parameters is highly dependent on the image. It has been proved by [Eglin 2007] that the response of Gabor filtering is concentrated on the regions with high frequencies.

With Gabor filtering, many features can be obtained. Features that are most commonly combined with Gabor filtering are listed below.

- Gabor-energy features

  The quantity of Gabor energy is computed based on the bank of Gabor filters. This feature simulates a specific orientation selective neuron in the visual cortex called the complex cell. It is closely related to the local power spectrum. The work of [Grigorescu 2002] has made a profound investigation on texture features based on Gabor filtering. The comparison results provided by Grigorescu et al. reveal the true power of Gabor energy. Subsequently, there are also many features derived from Gabor energy.

- Oriented patterns

  With an elaborate selection of basic parameters of Gabor filtering functions, the responses of Gabor filtering allow us to decompose the initial image into a set of separable directional maps containing oriented patterns (see Fig. 2.3). The feature extracted from such specific patterns is useful for the distinction of different textures.

Wavelet transform Wavelet transforms have proved to be a useful tool for many image processing applications. They have given good results for edge detection and texture identification. Wavelet transform can describe the content in a multi-resolution way. The applications of wavelet transform in handwriting analysis domain are usually writer identification and document characterization. There are discrete wavelet transform (DWT) and continuous wavelet transform (CWT). DWT provides a decomposition of an image into details having different resolutions and orientations. It is not translation invariant and has been mainly used for image compression. Compared to DWT, CWT is translation invariant and is often used as a redundant representation of an image. Two-dimensional CWT has been applied for handwritten numeral recognition [Romero 2007, Du 2010]. Nevertheless, we can notice that the basic wavelets decomposition considers the image into its horizontal and vertical edges, and corners. Consequently, the lack of directional selectivity and the sensitivity to translation make it difficult to adapt to very variant shapes where the orientations, the scales and the locations are massively variable.

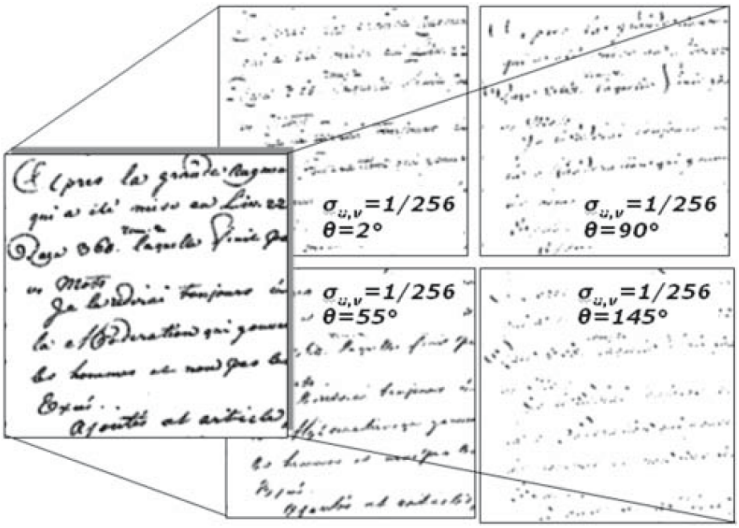


Figure 2.3: Handwriting decomposition using four directional maps [Eglin 2007]

Curvelet transform The curvelet transform has received a lot of attention in recent years due to its unique characteristics. This transform was developed from the wavelet transform and it has overcome some inherent limitations of wavelet in representing directions of edges in images. With the curvelet transform, images can be decomposed into different scale spaces, which makes the indexation of linear singularities of handwritten shapes possible. Since the curvelet transform is conceptually a multi-scale nonstandard pyramid, it can offer a solution that can describe both the global and local shapes of handwriting and meanwhile decorrelate these two image properties. G. Joutel et al. [Joutel 2008] have adapted the curvelet as a generic tool for applications to handwriting analysis (layout analysis, word spotting and document classification). In the work of [Joutel 2008], the curvelet transform is mainly used to extract orientations of handwriting fragments in different scales.

Steerable pyramid The steerable pyramid [Simoncelli 1995] is a linear multi-scale, multi-orientation image decomposition technique, that provides a useful front-end for image processing and computer vision applications. The steerable pyramid can capture the variation of a texture in both intensity and orientation. Concerning handwritten document analysis, it has been adopted for script identification [Benjelil 2009, Benjelil 2012]. The distinctive characteristics of scripts (intensity and orientation) can be analyzed after the steerable pyramid decomposition, as shown in Fig. 2.4. Compared to the wavelet transform and the curvelet transform, steerable pyramid decomposition contains not only the orientation information of the scripts, but also the intensity of ink trace.

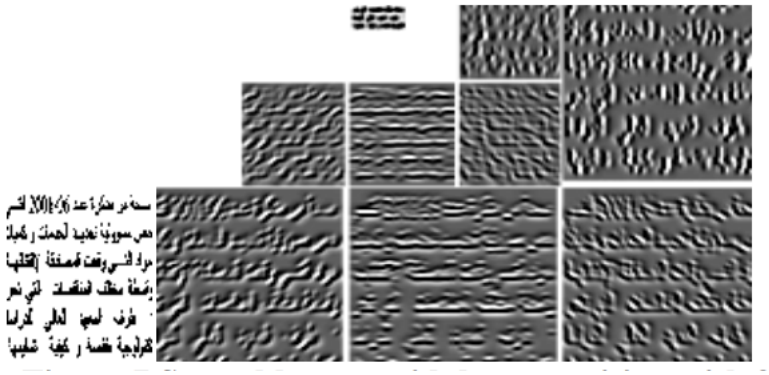


Figure 2.4: Steerable pyramid decomposition with 2 levels and 4 orientations of a printed Arabic text block [Benjelil 2009]

2.1.1.4 Zone features

- Zernike Moments

  A moment describes the layout (arrangement of image pixels). Moments are global region-based descriptors for shape and constitute a combination of area, compactness, irregularity, and higher order descriptor together. Zernike moment (ZM) is an excellent region-based moment which has attracted the attention of many image processing researchers since its first application to image analysis [Teague 1980]. It can be defined as a set of complete complex orthogonal basis functions that are square integral and are defined over the unit disk. These orthogonal functions are named Zernike polynomials as shown in Fig. 2.5. Since Zernike moments are orthogonal, i.e. there is no redundancy or overlapping of information between the moments, the moments are uniquely quantified based on their orders. Zernike moments are often applied to character recognition [Kan 2002]. The distinguishing property of Zernike Moments is the invariance of its magnitude with respect to rotation.

- HOG

  HOG is short for Histogram of Oriented Gradient, which is proposed by N. Dalal and B. Triggs [Dalal 2005] in 2005 for human detection. The basic idea of HOG is that local object appearance and shape can often be characterized pretty well by the distribution of local intensity gradients or edge directions, even without precise knowledge of the corresponding gradient or edge positions. In practice this is implemented by dividing the image window into small spatial regions ("cells"), for each cell accumulating a local 1-D histogram of gradient directions or edge orientations over the pixels of the cell. The combined histogram entries form the representation.

  After the success of HOG in human detection [Dalal 2005, Dalal 2006,

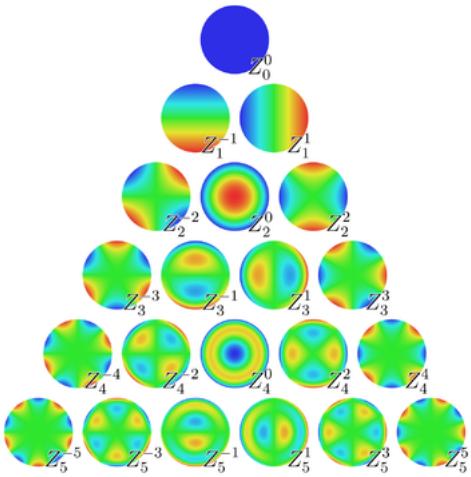


Figure 2.5: The first 21 Zernike polynomials (Z_n^m), ordered vertically by radial degree (n) and horizontally by azimuthal degree (m) [Tahmasbi 2012]

Suard 2006] and sketch-based image retrieval [Hu 2010], J. Almazán et al. [Almazán 2012b] have introduced HOG into handwritten word spotting application. In their segmentation-free system, the document images are divided into equal-sized cells and represented by 31-dimension HOG histograms (see Fig. 2.6). Queries are represented analogously using cells of the same size in pixels. The score of a document region can be calculated as the convolution of the query with respect to that region, using the dot product as a similarity measure between the HOG descriptors of each cell. The results presented in [Almazán 2012b] prove the efficiency and flexibility of HOG in document image analysis. Besides, there are some previous studies using the histogram of gradients for handwritten word spotting in a similar way to HOG [Rodríguez 2007, Leydier 2007, Leydier 2009].

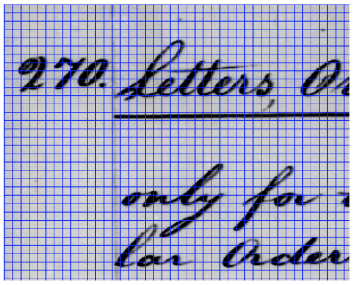


Figure 2.6: Grid of HOG cells [Almazán 2012b]

2.1.1.5 Point features

To explore suitable and effective local description for information retrieval on handwritten document images, we study different existing point descriptors which have become well-known for image recognition and retrieval in recent years. Depending on the nature of the considered descriptor, local characterization can be restricted to a small neighborhood or based on a large amount of information in the scope of the entire shape. When it comes to point features, one cannot avoid discussing interest points, which are the fundamentals of point features. Since point descriptors have been well received for years, the related issue — interest points — has also been widely studied. Here we give an overview only of interest points recently regarded as favorable and the corresponding descriptors concerning handwriting retrieval and recognition.

Interest points Many approaches have been proposed for detecting interest points in standard computer vision tasks, while in consideration of the specificity of the handwritten document images, we mainly study two types of interest points, which are respectively DoG (Difference of Gaussian) points and Hessian points.

- DoG (Difference of Gaussian) detector [Lowe 1999]

  Lowe proposed an efficient algorithm for object recognition based on local 3D extrema in the scale-space pyramid built with DoG filters. The input image is successively smoothed with a Gaussian kernel and sampled. The DoG representation is obtained by subtracting two successive smoothed images. Thus, all the DoG levels are constructed by combined smoothing and subsampling. The local 3D extrema in the pyramid representation determine the localization and the scale of the interest points. An illustration of DoG detector is given in Fig. 2.7.

  Compared to its predecessor LoG (Laplacian of Gaussians) detector, DoG can significantly accelerate the computation process. Nevertheless, the drawback of DoG representation is that local maxima can also be detected in the neighborhood of contours or straight edges, where the signal change is only in one direction. These maxima are less stable because their localization is more sensitive to noise or small changes in the neighboring texture.

- Hessian [Mikolajczyk 2002]

  The Hessian affine region detector relies on a multiple scale iterative algorithm to spatially localize and select scale and affine invariant points. At each individual scale, the Hessian affine detector chooses interest points based on the Hessian matrix at that point:

$$H(x) = \begin{bmatrix} L_{xx}(x) & L_{xy}(x) \\ L_{xy}(x) & L_{yy}(x) \end{bmatrix} \tag{2.6}$$

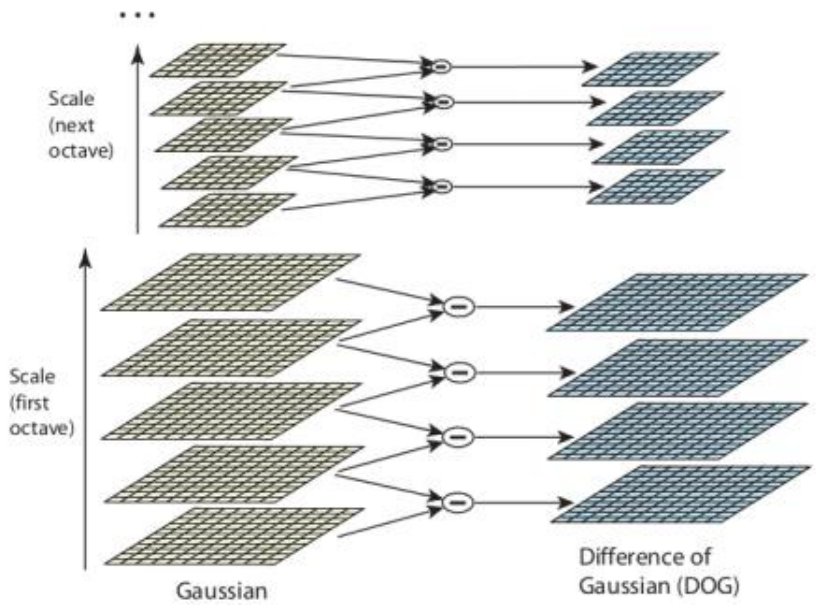


Figure 2.7: Illustration of the DoG filter application [Lowe 2004]

where $L_{aa}(x)$ is the second partial derivative in the a direction and $L_{ab}(x)$ is the mixed partial second derivative in the a and b directions. It is important to note that the derivatives are computed in the current iteration scale and thus are derivatives of an image smoothed by a Gaussian kernel. At each scale, interest points are those points that are simultaneously local extrema of both the determinant and the trace of the Hessian matrix. The trace of the Hessian matrix is identical to the LoG (Laplacian of Gaussians):

$$DET = \sigma_I^2(L_{xx}L_{yy}(x) - L_{xy}(x)^2) \tag{2.7}$$

where σ_I^2 is a factor used for appropriately scaling the derivatives.

Choose points that maximize the determinant of the Hessian as interest points. These interest points based on the Hessian matrix are also spatially localized using an iterative search based on the Laplacian of Gaussians. Predictably, these interest points are called Hessian-Laplace interest points. The Hessian detector solves the problem of DoG not being robust to the noises or small changes in the local texture.

Regarding the specificity of manuscripts, especially historical ones, whether the conventional interest point detectors introduced above are still the best option remains to be explored. However, there is no doubt of the necessity of finding stable and meaningful points for handwritten document analysis.

Point descriptors Concerning local description, Shape Context (SC), Local Binary Pattern (LBP), Scale Invariant Feature Transform (SIFT) and Speed Up Ro-

bust Features (SURF) are studied. All of them have been employed in handwriting retrieval and recognition tasks. Each of them has its own pros and cons. Next, we present a brief review of the four descriptors and the scenarios where they are used.

Shape Context [Belongie 2002] Shape Context (SC) descriptor captures the distribution over relative positions of other shape points and thus summarizes the global shape in a histogram. Due to its specific definition, Shape Context is a translation, rotation, scale invariant descriptor. It was first developed for shape matching and object recognition in [Belongie 2002]. Afterwards, researchers started to use it with certain adaptation for symbol classification as well as handwriting recognition and indexing [Nguyen 2008, Lladós 2007, Su 2011, Costagliola 2011]. Since Shape Context can encode the information of the entire shape, it is more tolerant to small distortions than other point descriptors. In our proposed handwriting represented model, Shape Context also plays an important role. The details on how it is adapted are presented in chapters 3 and 4. Fig. 4.31 gives an illustration of Shape Context computation.

Figure 2.8: The illustration of Shape Context computation [Belongie 2002]

Local Binary Pattern [Wang 1990] Local Binary Pattern (LBP) is a texture feature which has been highly successful for various computer vision problems such as face recognition and background subtraction. It is powerful in terms of its stability against variation of illumination. The LBP operator describes each pixel by the relative grey-levels of its neighboring pixels. Fig. 2.9 gives an illustration of different numbers of neighbors in four scales. If the grey-level of the neighboring pixel is higher or equal to the grey-level of the reference pixel, the value is set to one, otherwise to zero. The descriptor describes the results over the neighborhood as a binary number. Since the correlation of pixels decreases with the distance, the radius of neighborhood is usually kept small or the contribution of neighboring pixels is weighted by distance, as Fig. 2.9 shows.

Many works on handwriting recognition and analysis have employed LBP [Du 2010, Nicolaou 2013, Nicolaou 2014, Biglari 2014]. Depending on the specificity of tasks, different adjustments are made. Du et al. [Du 2010] have applied LBP after wavelet decomposition for writer identification. LBP histogram is calculated for each subband wavelet transform and the final feature of the image is the concatenation of four corresponding histograms. The works of [Nicolaou 2013, Nicolaou 2014] try to enhance the characteristics associated with writing styles and fonts by select-

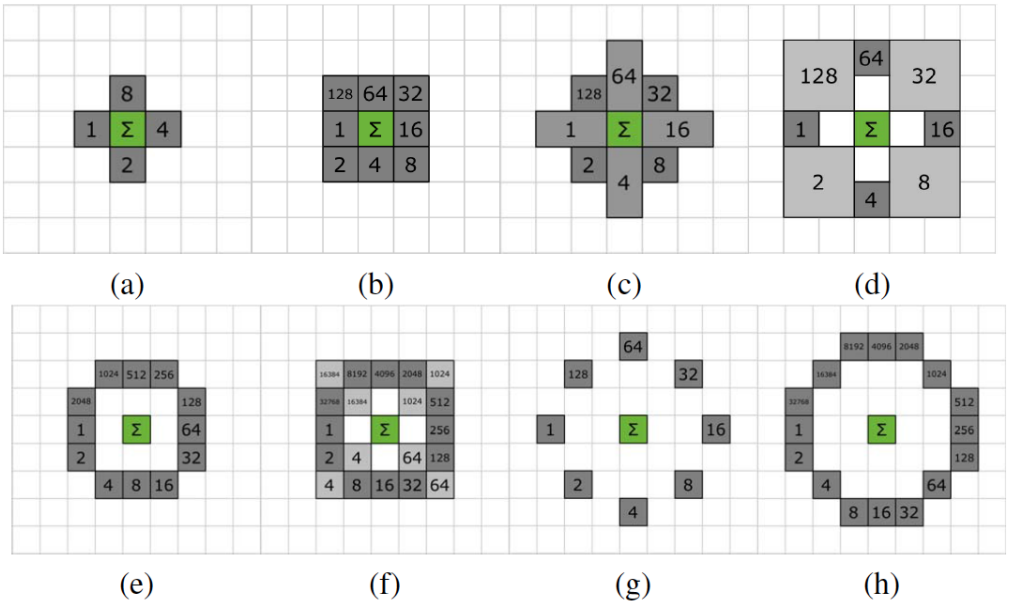


Figure 2.9: Indicative LBP operators: (a) LBP1,4 (b) LBP1,8 (c) LBP1.5,8 (d) LBP2,8 (e) LBP2,12 (f) LBP2,16 (g) LBP3,8 (h) LBP3,16. Dark grey represents pixels with 100% contribution, grey represents pixels with 50%, light grey pixels with 25%, and green is the reference pixel. ($LBPr, b$, where r is the radius of the neighborhood, and b is the number of samples.) [Nicolaou 2014]

ing specific LBP patterns. For Persian and Arabic handwritten digit recognition, images are divided into blocks and LBP histograms are computed over each block. The evaluations of these works all demonstrate the strength of LBP as a textural and structural descriptor.

SIFT [Lowe 1999] The original SIFT descriptor records the gradient orientation distribution of the 16x16 neighborhood of the reference point. It is well known for its scale-invariant and rotation-invariant characteristics with the cost of expensive computation. Its applications include many tasks in computer vision, such as object recognition, 3D modeling, gesture recognition, video tracking and so on. In the document image analysis domain, SIFT is also well received. Both Rusiñol and his group [Rusiñol 2011, Aldavert 2013, Almazán 2014] and Rothacker et al. [Rothacker 2013] have employed dense SIFT in their word spotting approaches and achieved outstanding performance with it. Concerning handwriting recognition, Zhang et al. [Zhang 2009] have developed a novel descriptor dedicated to handwritten characters based on SIFT, which is called "character-SIFT". Fig. 2.10 shows a 2 x 2 SIFT descriptor array computed from an 8 x 8 set of samples.

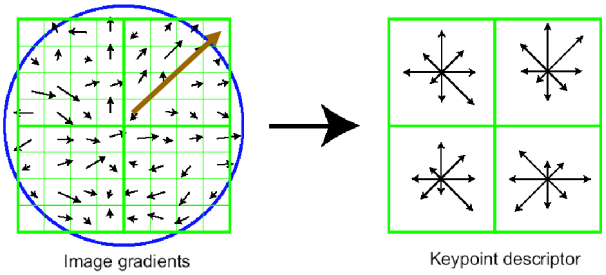


Figure 2.10: Illustration of SIFT extraction. A keypoint descriptor is created by first computing the gradient magnitude and orientation at each image sample point in a region around the keypoint location, as shown on the left. These are weighted by a Gaussian window, indicated by the overlaid circle. These samples are then accumulated into orientation histograms summarizing the contents over 4x4 subregions, as shown on the right, with the length of each arrow corresponding to the sum of the gradient magnitudes near that direction within the region. [Lowe 2004]

SURF [Bay 2008] SURF is considered as an approximate version of SIFT. The difference is that SURF adopts integral images and calculates Harr wavelet instead of orientation histogram, which makes SURF more computationally efficient than SIFT. SURF is commonly used to address the issue of speed.

2.1.1.6 Simulated dynamic features

There are many particular features which can be only extracted from so-called "online" handwritten shapes, for example, speed of writing, pen pressure during writing, information about pen-ups and pen-downs and so on. The efficiency and significance of such dynamic features have been confirmed by a great many works dedicated to online handwriting recognition. In the light of the usefulness of the dynamic information, some researches attempt to simulate the dynamic process of handwriting, i.e. extract the trajectories from the image representation. In this way, 2-dimensional static manuscripts can be described in one dimension by means of temporal data.

The key point of extracting simulated dynamic features is trajectory reconstruction. The conventional techniques of automatically reconstructing the trajectory involve the minimization of global parameters such as length, average curvature, or directional changes [Rousseau 2006, Niels 2007]. Once the drawing order of handwriting is recovered, the online handwriting recognition system can be used for offline scripts.

2.1.2 High-level features

2.1.2.1 Ascender/descender

Ascenders are strokes above the upper baseline (e.g. upper part of "b" or "l"). In the same manner, we refer to strokes below the lower baseline as descenders.

The use of ascenders/descenders can be first found in works on document image categorization [Myers 1995, Spitz 1997]. The main idea of these works is to convert characters into a shape code by using different features. Ascender/descender is one of the most popular features used in such applications. Recently, many significant works dedicated to word spotting and handwriting recognition have also used strokes, especially ascenders and descenders, as part of their description of handwriting. The work of [Lavrenko 2004a] offers an approach for historical handwriting recognition, considering severely degraded situation. It adapts both scalar features and profile-based features. The number of ascenders and the number of descenders are two out of six scalar features used in this approach. Not only for handwritten text, ascender/descender is also interesting for printed documents. S. Bai and et al. [Bai 2009] develop a fast keyword spotting approach for printed document images without performing OCR. They extract 7 different features (including ascenders/descenders) and convert the word into a Word Shape Code (WSC) that describes the features from left to right. Later, a similar idea is adapted to word spotting on cursive handwritten documents with a modified character shape code by S. Sarkar [Sarkar 2013]. Moreover, with the work of [Pourasad 2012], it turns out that ascenders/descenders are discriminant characteristics not only for Latin language but also for Farsi and Arabic scripts. An example of baselines of a handwritten French word is given in Fig. 2.11.

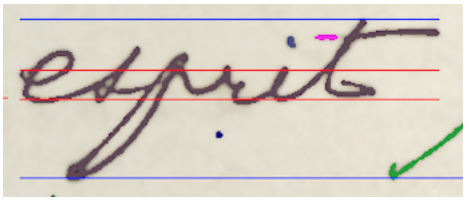


Figure 2.11: Baselines of a handwritten word

2.1.2.2 Loop

As a prominent pattern, loop is very distinguishing no matter in terms of letters or writing styles. A lot of previous researches like [Blankers 2007, Steinherz 2009] have shown the importance and usefulness of loops. A loop is defined as a closed curve, where the ends cross each other at some intersection point as shown in Fig. 2.12.

There are many approaches dedicated to loop detection in handwriting. Since online handwriting has more information than offline handwriting, the detection of loop is relatively simpler. Here, we are focusing on offline loop detection and analysis. Regarding handwriting, there are two types of loops, evident ones and hidden ones. It is relatively easier to discover evident loops. For example, based on the graph representation of handwriting, we can use the DFS (Depth First Search), a classic method in graph domain to detect loops. Since loops have been employed in our proposed representation model, more details about the loop detection will be

Figure 2.12: Loops in handwriting marked by red color

presented later in chapter 4. Visually, we can easily perceive that the loops of small letters like "a", "b", "d", "o" differ from the loops of "l" and "h" or the loops of "g", "j" and "y", and of course from the loops in capital letters. In other words, once we can capture such characteristics of loops, we obtain more significant information on the recognition of handwritten letters. To describe loops, many attributes can be used, such as the location of loops, the area/perimeter of loops, the main orientation of loops and so on. We will also highlight the importance of loops in our work due to its discriminant power.

2.1.2.3 Grapheme

Grapheme is a high-performing feature for offline writer identification application. It is generated by splitting the ink trace into fragments (see Fig. 2.13). In the approaches using graphemes, there are two important techniques, segmentation and normalization. A brief review on these two aspects is given in the following.

Segmentation How to decompose text images into effective fragments is a crucial step in grapheme-based algorithms. Whether extracted fragments are distinctive directly affects the final performance of the approach. Nowadays, there are mainly two types of segmentation methods. One is based on the contours of characters. The other extracts the skeleton of characters first, and then splits the ink trace into useful fragments.

The typical segmentation method assumes a binary input image, and breaks cursive writing heuristically on the vertical minima of the ink trace, which is originally given in [Lecolinet 1995] for OCR and aims to produce the most character-like segments. However, this method loses information of some joints, or ligatures which also contain writer-specific information. As research on segmentation develops, more segmentation methods are introduced.

- Contour-based segmentation methods

  Compared with the recent skeleton-based segmentation methods, studies on contour-based segmentation methods are more numerous and were carried out earlier. The segmentation of this category is dependent on the contour of the connected components. After tracing the contour, we can start segmenting the texts into fragments. Three segmentation methods are studied.

A. Segment on Y-minima

In Schomaker and Bulacu's work [Schomaker 2004, Schomaker 2007], the vertical local minima of the lower contour are found and the nearest local minimum in the upper contour is determined for each of them. If the distance between two corresponding points is similar to the ink-line width, the connected component will be segmented at this point. The assumption implicit in the minima splitting method is that the character body contains writer-specific information. Fig. 2.13 shows one example of segmenting a sample into fragments based on Y-minima of the contour.

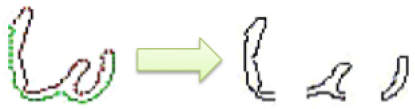


Figure 2.13: Segmenting a character into graphemes on Y-minima [Ghiasi 2010]

B. Segment by preserving ligature

An alternative hypothesis is that the ligatures between characters contain writer-specific information, and should be preserved. In this way, all the local minimum should be preserved. The implementation of the ligature method initially employs the same minima detection process, but splits at the middle point between two adjacent minima. A notable effect of this process is that graphemes are no longer guaranteed to be connected components themselves.

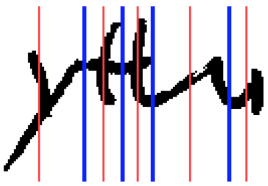


Figure 2.14: Comparison of splitting points produced by the minima (blue) and ligature (red) segmentation methods [Gilliam 2011]

In Fig. 2.14, different segmentation results for the same sample are shown. Blue lines represent the segmentation based on Y-minima, and red lines correspond to ligature-based segmentation. An experiment on efficiency is conducted by T. Cilliam et al. [Gilliam 2011]. Since these two techniques are complementary, their combination is also tested. The results show the minima segmentation method performs better than both the ligature segmentation method and the combination method.

C. Segment on fixed width

In the Y-minima segmentation method, connected components are divided so that the resulting fragments are more meaningful. However, extracted fragments for writer identification do not need to be meaningful, and their shapes are important. Moreover, the segmentation of a connected component in its local minima results in losing information about the shape of the minima. To avoid this disadvantage, G. Ghiasi and R. Safabakhsh [Ghiasi 2010] propose a novel segmentation method. First, a number of segments with specific length are chosen from the lower contour. Then two points with the same x coordinate as the end points of each segment are determined on the upper contour. Also follow the ink-trace width. The beginning points of different segments on lower contours can be considered some distance apart. This distance parameter is called "gap". It is a tested parameter. 10 and 20 were used in the experiments in [Ghiasi 2010]. An example of applying fixed-width segmentation is given in Fig. 2.15.

Figure 2.15: Segmenting a connected component using fixed width [Ghiasi 2010]

The number of graphemes obtained from this approach is more than the ones from segmentation on Y-minima. They contain all the information of all parts of connected components, but there are a lot of overlapping parts and repeated information.

- Skeleton-based segmentation methods

  After obtaining the skeleton of connected components, the segmentation of strokes is performed as follows: between each starting and ending point, all the points involved in the formation of a stroke will be saved in a list with their directions and the thickness. Three criteria of decomposition are set. First of all, the points of minimum thickness (local minimum) are marked as points of cutting. Second, the junction points are also considered as decomposition points. The last criterion is if the tracing algorithm detects a point that is already marked as visited; we stop the tracing and mark it as a segmenting point [Daher 2010b].

Normalization For the tasks like handwriting retrieval and recognition, before using graphemes, we need firstly to normalize them. The normalization method is specific to the representation of the grapheme and essential to allow the comparison between writings which vary in size. The size normalization can be operated by fitting either a single dimension or both dimensions.

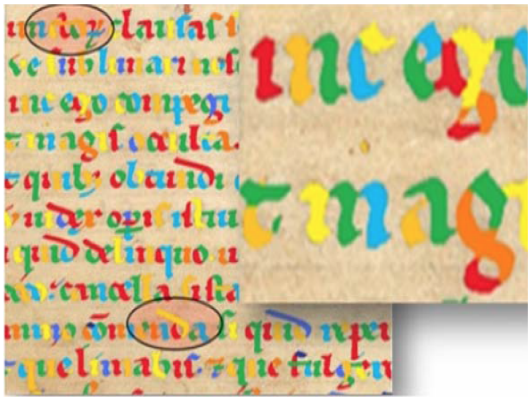


Figure 2.16: Example of strokes decomposition into graphemes [Daher 2010b]

- Aspect-ratio normalization

  Preserving the aspect-ratio by scaling in only one dimension retains ratio information that may be writer-characteristic [Bucalu 2005].

- Square-ratio normalization

  Some forms of constant scaling in both dimensions have been shown to be beneficial to the writer identification rate.

Figure 2.17: Comparison of graphemes produced by the ratio (top) and the square (bottom) grapheme size normalization methods [Gilliam 2011]

2.1.2.4 Lexicon

***N*-gram-based textual descriptor** N-gram language model is designed to provide a simple approximation for sentence probabilities based on the relative frequencies of word sequences of the length n. For $n = 1$, $n = 2$, and $n = 3$, we respectively use the term unigram, bigram and trigram. N-gram language model was originally proposed for speech recognition. Later, the integration of word bigram and trigram models into HMM-based handwriting recognition systems became popular

[Zimmermann 2004]. Currently, it has also been adapted to handwritten word spotting in order to address the query-by-string problem [Aldavert 2013, Almazán 2014].

For the query-by-string word spotting, the transcriptions are divided into a set of consecutive and overlapping blocks of unigrams, bigrams and trigrams respectively. This simple representation allows us to extract information from which characters compile a word and to encode some neighborhood information. An example of the n-gram frequencies generated by a transcription is shown in Fig. 2.18. The n-gram-based textual descriptor is obtained by simply accumulating the occurrences of each n-gram into a histogram and normalizing this histogram by its L2-norm. The number of different n-grams available is obtained by generating a codebook which maps each n-gram into a dimension of the textual descriptor. This codebook is generated from the training set where the textual information is available. Then, in the retrieval phase, the n-grams which do not appear in the codebook are ignored.

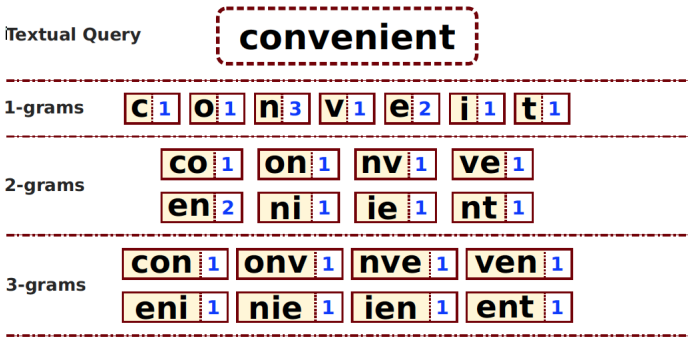


Figure 2.18: Example of the n-gram textual descriptor [Aldavert 2013]

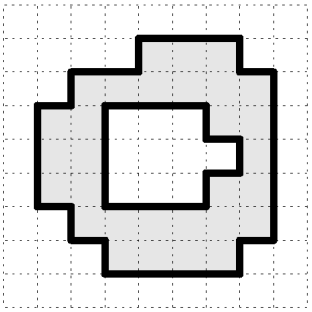
An exhaustive review of features, descriptors used in the context of handwriting analysis and recognition has been presented in this section. As a summary, we would like to point out that each feature has its pros and cons regardless of low-level or high-level. The selection of features is supposed to be made in a comprehensive consideration of the application requirements, such as accuracy, complexity and speed.

2.2 Handwriting representation

As mentioned in the beginning of this chapter, image representation plays an identically important role in handwritten document analysis to that of features do. The adaptation of an appropriate representation model can much improve the performance of the algorithms. Hence, we carry out an in-depth study of the representation models which have been used in handwritten document analysis, especially the ones dedicated to handwriting retrieval and recognition.

2.2.1 Bitmap representation

A bitmap representation encodes the entire binary image as a two-dimensional array where each entry corresponds to one pixel (see Fig. 2.19). Since each pixel is either black (0) or white (1), the entry in the array requires as little as 1 bit of memory space. This significantly reduces the memory requirements for storing the image and makes the bitmap encoding very popular. On the other hand, any image manipulation is very slow, since each pixel has to be processed individually.



0	0	0	0	0	0	0	0	0
0	0	0	0	1	1	1	0	0
0	0	1	1	1	1	1	1	0
0	1	1	0	0	0	1	1	0
0	1	1	0	0	0	0	1	0
0	1	1	0	0	0	1	1	0
0	0	1	1	1	1	1	1	0
0	0	0	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0

(a) (b)

Figure 2.19: (a) Original image (b) its bitmap representation [Slavik 2000]

2.2.2 Shape coding

Shape coding is one of the representation models that were used for handwriting recognition in early times. Initially, it was developed for the printed text. Along with its success, researchers started to use it for handwriting with adaptation. The general idea is to encode the characteristics of characters or lexicons or words into a simple and compact feature vector. In reality, the selection of these characteristics can be different from case to case.

2.2.2.1 Chain-code representation

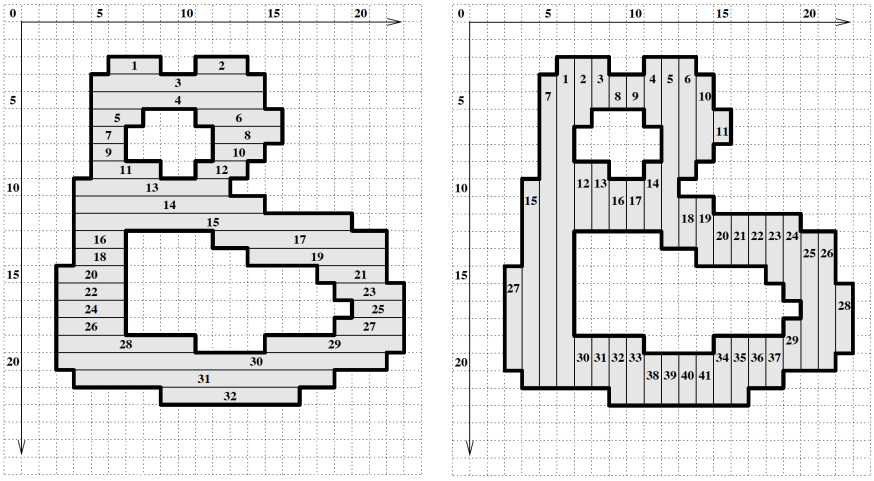
Contour as a dominant property of the shape, is selected to represent the image by using the chain code [Kimura 1993, Kim 1996], one succinct and convenient encoding technique in early days. The approach is to store only the information of the contour pixels by its x and y coordinates and represents the pattern as a list of exterior and interior contours. Fig. 2.20 gives one example of using a specific chain-code technique - Freeman code - to represent handwriting. Since any minor changes on the contour can introduce contour breaks or illegal contour points, chain-code representation becomes more deficient in such situations.

(a) (b)

Figure 2.20: (a) Different chain codes and corresponding angles (b) contours replaced by direction codes (colors represent codes) [Siddiqi 2009]

2.2.2.2 Run-length representation

In the work of [Slavik 2000], Slavik et al. choose to use the run-length as the technique to depict patterns. Run-length encoding keeps track of horizontal/vertical runs of black pixels, storing the coordinates of the first pixel and the run length. Fig.2.21(a) shows an image of a letter "B" together with the corresponding horizontal runs. There are 32 different runs, hence this image can be represented as an array of 32 triples of the form (row, first column, run-length). This is the basic version of image representation using the run-length. To describe precisely the images, contextual information should be added, i.e. the track of neighbor (above and below for horizontal, left and right for vertical) runs that touch the current run.



(a) (b)

Figure 2.21: (a) Horizontal and (b) vertical run-length representations of an image [Slavik 2000]

2.2.2.3 Character shape code

Character shape codes have been developed in the field of machine printed text recognition [Spitz 1997]. They are used to classify characters based on their optical forms into a few classes, called shape codes. Each character is assigned to exactly one shape code. The definition of shape code classes can be various. Usually, the principles of defining the criteria are relatively simple, just by indicating the presence or the absence of the characteristics, e.g. ascenders/descenders, loops and so on. Many characteristics are selected for making shape code. Each combination of them indicates a specific character. It needs to be pointed out that the shape code classes are clearly defined for printed characters, whereas they are no longer unique for handwritten characters due to different individual handwriting styles. Ambiguity may be introduced because of this. Different adaptations have been made regarding particular handwriting variations. One of them is to segment characters into graphemes and to apply the shape code for each graphem, as shown in Fig. 2.22. The simplicity of this approach is a big advantage. Nevertheless, its inferior robustness to the handwriting variations makes itself less competent.

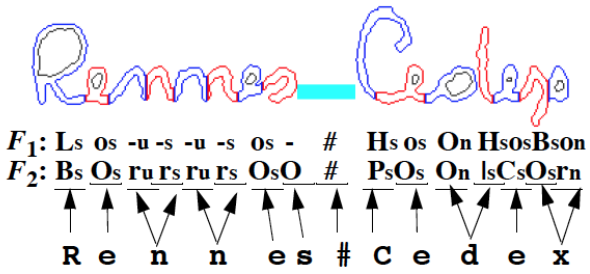


Figure 2.22: One example of representing a word (or sequence of words) image using character shape coding [El-Yacoubi 1999]

2.2.3 Bag-of-visual-word model

Bag of visual words (BoVW) is a well-known technique for image representation inspired by models used in natural language processing. Recently, BoVW has attracted the attention of researchers in the handwritten document analysis domain, due to its significant success on standard computer vision applications during the last decade [Rusiñol 2011, Aldavert 2013, Rothacker 2013].

Generally speaking, keypoints are grouped into a large number of clusters. Those with similar descriptors are assigned to the same cluster. By treating each cluster as a "visual word" that represents the specific local pattern shared by the keypoints in that cluster, a visual-word vocabulary describing all kinds of such local image patterns is created. With the keypoints mapped into visual words, an image can be represented as a "bag of visual words", or specifically, as a vector containing the (weighted) count of each visual word in that image, which is used as feature

vector. Since the BoVW model is analogous to the bag-of-words representation of text documents in terms of form, many existing techniques for text retrieval and categorization are applicable to the similar problems of images.

The original BoVW model lacks the spatial information of visual words and the relations among them, both of which are important aspects of describing the image. Concerning the problem of spatial information, spatial pyramid matching (SPM) scheme has been introduced. Latent semantic analysis (LSA) [Deerwester 1990] is adopted for finding out the underlying relations among visual words.

2.2.4 Graph-based model

Graph-based representation is a popular approach to capture and model the structural properties of objects. It has been widely used in the pattern recognition domain, for instance, in molecular compound recognition in chemistry and symbol recognition in document analysis. Recently, it has also been adapted to handwriting recognition and analysis. Researchers start using graphs in the context of single character recognition, such as the graph-based recognition of Chinese characters reported in [Lu 1991, Zaslavskiy 2009]. The graph-based model proposed in [Lu 1991] reflects the hierarchical nature of handwritten characters, especially of Chinese characters, which have a more complex structure than Latin languages. Later, Fischer et al. developed a graph representation model based on the skeleton of word image, which contains only vertexes without edges [Fischer 2010a, Fischer 2013]. By adding enough intersection points among keypoints, the structural information is still preserved and meanwhile graph representation is robust to the bad quality of the skeleton. Currently, due to the expensive computational cost of graph matching, the transformation of graphs into feature vectors calls for attention. In [Chherawala 2011], a directed acyclic graph is first constructed for each subword, and then is converted into a topological signature vector that preserves the graph structure. Moreover, Lladós et al. [Lladós 2012] adapt the graph serialization concept in graph matching for handwritten word spotting. By extracting and clustering the acyclic paths of graphs, one-dimensional descriptor, bag-of-paths (BoP), is generated for describing the words.

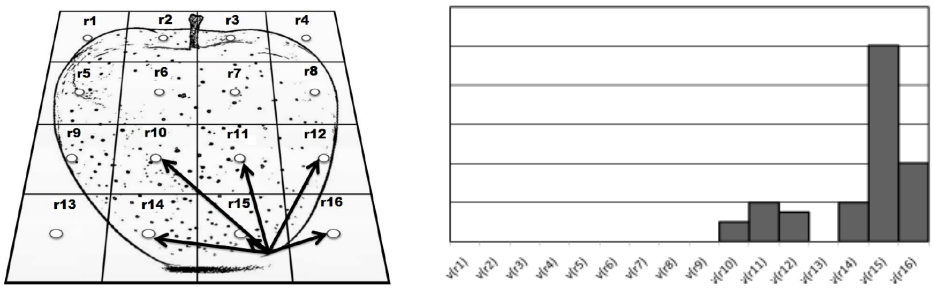
With the existing works using the graph-based model to represent handwritten patterns, the unique true strength of graphs in modeling the objects has been demonstrated. Its capability of preserving the structure of objects makes it very powerful in distinguishing patterns. However, the expensive computation of graph matching is a common drawback that obstructs the wide application of the graph-based representation in practice.

2.2.5 Blurred shape model

Blurred Shape Model (BSM) [Escalera 2009] is originally designed for symbol recognition. It codifies the shape in terms of a set of interest points. Taking into account these pixels, the Blurred Shape Model representation defines a set of spatial regions

by means of a grid. Then, spatial relations among interest points from neighbor regions are computed, and the representation vector is obtained by the accumulation of contributions of all interest points.

Given a set of points forming the shape $S = x_1, \cdots, x_m$ of a particular symbol, each point $x_i \in S$, called SP, is treated as a feature to compute the BSM representation. The image region is divided into a grid of $n \times n$ equal-sized sub-regions (cells), noted as r_i. Each cell receives votes from the interest points in it and also from the interest points in the neighboring sub-regions. Thus, each SP contributes to a density measure of its cell and its neighbors, and thus, the grid size identifies the blurring level allowed for the shape. This contribution is weighted according to the distance between the point and the centroid c_i of the region r_i. A simple demonstration is given in Fig. 2.23. In the end, each image is represented by the BSM as a single vector.



(a) (b)

Figure 2.23: BSM density estimation example (a) distances from a contour point to its neighbor centroids (b) vector descriptor update using the distances of (a) [Escalera 2009]

The Blurred Shape Model encodes the spatial probability of the appearance of the shape pixels and their context information. Contrary to the representation based on point features, BSM allows the comparison of two images by directly comparing their feature vectors, e.g. using the Euclidean distance, which is convenient and efficient while dealing with large-scale datasets. The main shortcoming of BSM is that it cannot be applied without precise word segmentation. Since its construction is based on the grid, the definition of grid needs a clear boundary of words or characters.

2.2.6 Biologically inspired model

Recently, Van Der Zant et al. [van der Zant 2008] innovatively applied computational models of the neurophysiology of vision to classify handwriting images which are the most difficult to retrieve with a low frequency of occurrence. This biologically inspired method was first proposed by T. Serre et al. [Serre 2007] for object

recognition. It works very well on a standard set of computer vision problems. The theory behind this model is to stimulate the system of the visual cortex (see Fig. 2.24). It is composed of four layers. They are respectively named Gabor functions, local pooling, radial basis functions and global pooling.

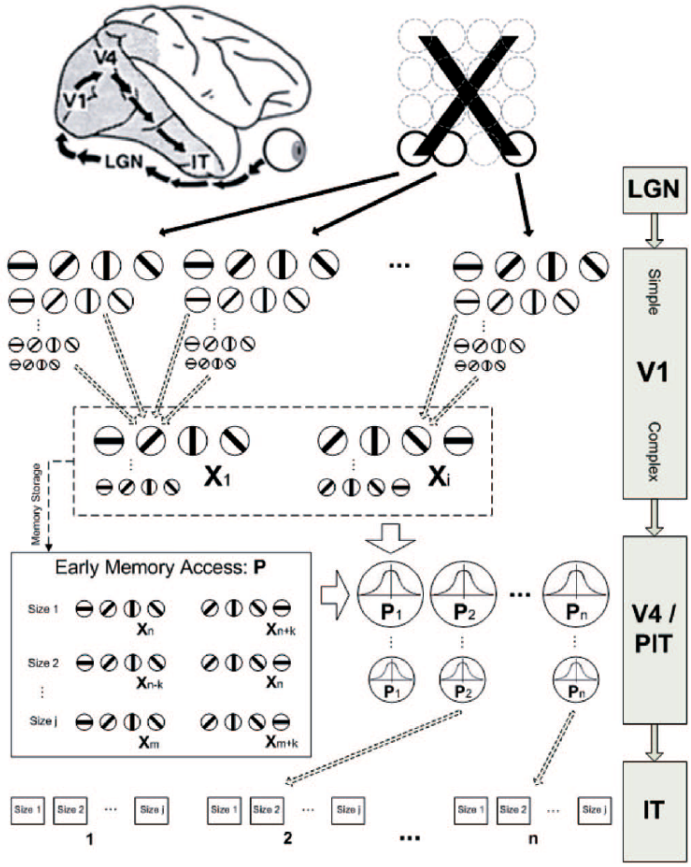


Figure 2.24: Overview of visual processing according to the standard model of visual processing [van der Zant 2008]

- Layer 1: Gabor functions

  This layer consists of simple cells that process the signals transmitted by the lateral geniculate nuclei (LGN). Gabor functions are used to model the response of neurons.

- Layer 2: local pooling

  Complex cells in the second layer respond according to the maximum activation among the input given by simple cells. Oriented bars or edges within the receptive fields ignite the activation of complex cells.

- Layer 3: radial basis functions

  The third layer contains both simple and complex cells. Simple cells take the output of complex cells and stretch over all orientations, combining bars and edges into more complex shapes. Radial basis functions are implemented for the simple cells of this layer.

- Layer 4: global pooling

  There are only complex cells in the fourth layer. A complex cell in this layer will respond according to the most active simple cell of the previous layer. It is selective for the same combination of oriented bars.

  For further details, please refer to the description provided in [van der Zant 2008].

2.2.7 PDF-based representation

In this section, an overview of significant handwriting representation models used in the context of writer identification is presented. It should be noted that the details of writer identification methods are not exhaustively discussed in the thesis since it is not the subject of this PhD work. The focus of this part is the most commonly used representation model in writer identification approaches, the PDF (probability distribution function)-based representation.

Besides handwriting recognition and retrieval, researches on writer identification and similar applications (document classification, signature verification etc.) have also been active and popular in the document analysis domain for decades. As we call it writer identification, to make it possible, at least one handwriting sample of the writer is required. Usually, a writer identification system performs a one-to-many search in a large database with handwriting samples of known authorship and returns a likely list of candidates (see Fig. 2.25). In this way, it can be considered as a special kind of image retrieval. Compared to handwriting recognition and retrieval tasks, what is in common is that both of them need to extract features and to build a model to represent the handwriting images. The different part of these two applications is the way in which they deal with the variations among diverse handwritings. For handwriting recognition approaches, they seek to find an invariant representation model capable of eliminating inter-writer variations in order to obtain unified shapes of characters and words robustly. On the contrary, the solution of writer identification needs to enhance such inter-writer variations so as to make the personal characteristics of handwriting more prominent.

In general, most writer identification approaches represent handwriting with different features in a statistical way. The probability distribution functions (PDFs) of different features are calculated and the likelihood is measured between two samples. The features used in the context of writer identification and verification are usually classified into two categories, macro- and micro- features [Srihari 2005] (in some works, they are also referred as textural and allographic features). The ones which capture the global characteristics of writers' individual writing habit and style

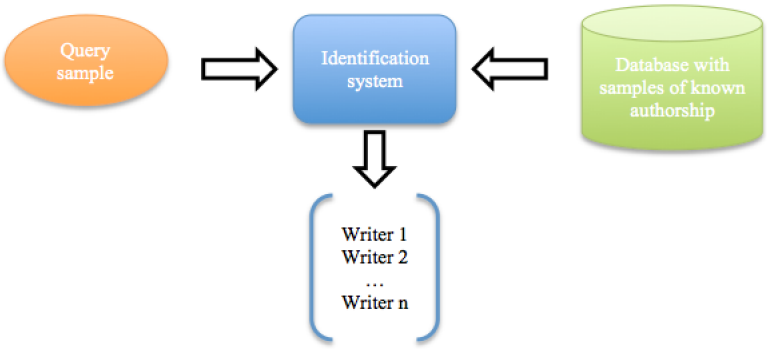


Figure 2.25: Demonstration of a writer identification system

are regarded as macro-features, whereas the ones that capture finer details at the character level are micro-feature. Grey-level entropy and threshold, number of ink pixels, number of interior/exterior contours, number of four-direction slope components, average height/slant, paragraph aspect ratio and indentation, word length, and upper/lower zone ratio all belong to macro-feature group. They are real values that can be directly used for similarity measure. Micro-features respectively capture the finest variations in the contour, intermediate stroke information and larger concavities and enclosed regions.

Depending on the selection of features, the PDFs are computed differently. There are directional PDFs, grapheme emission PDFs and so on. For the low-level features, usually the PDFs are directly calculated after the feature extraction, whereas a classification of high-level features needs to be performed first before calculating the PDFs. The output of the PDF is a vector of probabilities capturing a facet of handwriting uniqueness.

Two of the most critical factors in pattern recognition tasks, features and representations, have been thoroughly studied in this chapter. We have given an elaborate introduction to low-level and high-level features as well as representation models which have been used in handwritten document applications. After the study of the state of the art, we established a new handwriting representation model, which is described in detail in the following chapter.

Chapter 3

Comprehensive Representation Model

Contents

3.1 Preprocessing 38
3.1.1 Noise elimination 39
3.1.2 Binarization 40
3.1.3 Enhancement 41
3.1.4 Line segmentation and baseline detection 42
3.2 Feature selection and extraction 46
3.2.1 Contour and skeleton based decomposition 46
3.2.2 Structural point detection 56
3.2.3 Shape context 59
3.2.4 Point feature performance evaluation (SIFT, SURF, Shape Context) 63
3.3 Graph-based representation model 65
3.3.1 Graph-based approaches for pattern recognition 65
3.3.2 Graph-based representation model 68
3.3.3 Graph representation of handwritten document images 70
3.3.4 Approximate graph edit distance 74

The representation of images is a basic but crucial part in all applications dealing with handwritten documents. A good handwriting representation model is supposed to keep a proper balance between all the information present in the image, especially the specificity and internal diversity of the content.

In most pattern recognition approaches that are developed for indexing handwritten document images, pages are segmented into words and are then matched as images and grouped into clusters containing all instances of the same word. Some structural feature sets like the profile-based features proposed by Rath and Manmatha in [Rath 2007] have been developed to recover the "inner" structure of words. But one of their limitations is their strong dependency on skew/slant angle normalization and baseline detection. Since stroke order and position as dynamic information reveal the handwriting execution, other significant works based on the application of loops and strokes are developed from skeleton-based description [Lavrenko 2004b, Pervouchine 2005, Jager 1996, Kégl 1999]. In most of these

works, structural information is used to model writer style for its identification and examination. After a long-term study of the state of the art, we realized that the requirement for a robust and complete handwriting representation is not yet satisfied. There are still significant lacks of the capability of dealing with large irregularities and inherent variations of handwriting, the presence of diverse noises, the unconstrained environment and so on. It is necessary to consider a mixed means of characterization for the handwritten document.

If we formulate handwritten document images in terms of composition, it can be written as

$$Handwritten\ Image(x,y) = Shape(x,y) + Texture(x,y) + Noise(x,y) \quad (3.1)$$

where (x, y) represents the coordinates of the pixel.
The first two parts contain useful information of the content, while the last part is random distortion, which is of no help in the analysis of the images. A comprehensive representation model should be able to cover the first two dimensions with the consideration of their local relevance.

In this chapter, we mainly present a novel representation model for handwriting that comprises both morphological and topological properties. The chapter starts with the preprocessing techniques used in our approach, and then we discuss the feature selection and extraction. The third part gives an elaborate explanation on the proposed representation model.

3.1 Preprocessing

The preprocessing of historical document images is a large topic, which contains a lot of techniques specifically considering different applications. Many researchers have devoted themselves to this study for a long time with abundant outcomes. Since the focus of our research is to develop an appropriate and effective representation model for handwriting, and not the preprocessing, we did not conduct a deep study of preprocessing. In this section, we give a brief introduction and some small demonstrations on the preprocessing techniques that we apply in our case. A workflow of basic preprocessings is presented in Fig. 3.1.

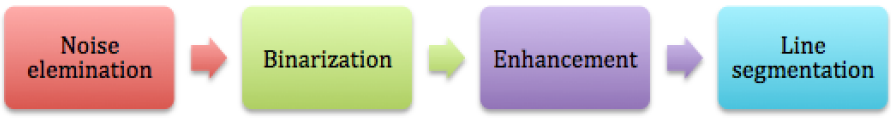


Figure 3.1: Workflow of the regular preprocessing in our case

3.1.1 Noise elimination

All document images, no matter acquired by scanner or digital cameras, almost always include some noisy artifacts. Usually, cost is often a major concern, which makes it difficult to use the best available equipment. What is worse, some libraries scan microfilms, which does not require the same level of care and can be automated. However, this additional reproduction of the images produces more noise. Bilevel scanning is another issue during the digitization process. Usually there is no control over the threshold that is being used. A lot of information is lost due to this process. Historical documents usually should not be scanned in black and white. Besides the quality loss caused by the digitization process, degradation due to age is also a typical problem. Since the document images that we deal with are historical ones, depending on the age as well as the quality of preservation efforts, a large amount of noise can be exhibited. Dirty marks, stains and missing parts, non-uniform paper color, faded ink, ink bleeding (ink traveling laterally in the paper) and ink through (ink traveling through the paper from the other side of a page) are all the noise that occur in the historical documents. Thus, one primary function of preprocessing is to eliminate or reduce the noise.

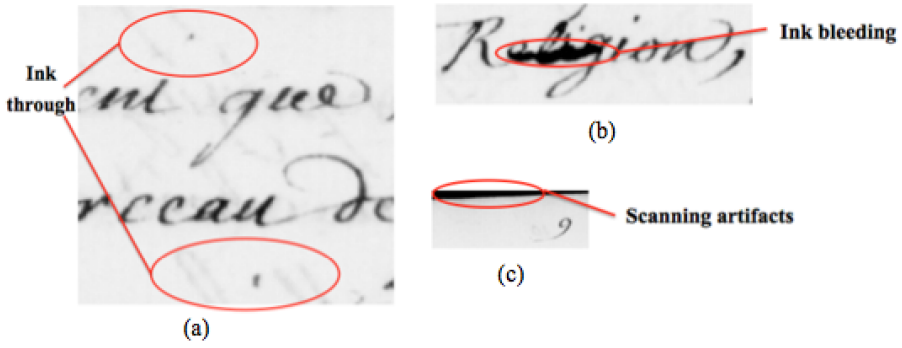


Figure 3.2: Examples of artifacts and noise that typically occur in historical document images (a) ink through (b) ink bleeding (c) scanning artifacts (all images taken from clandestine philosophy corpus)

A Gaussian filter is applied to the image. This is a simple but frequently used smoothing process in order to eliminate the noise on the image. The side effect of this operation is that the quality of the image is decreased and the object becomes blurred. As a trade-off, the greater the amount of noise reduction, the more details are filtered. To keep the side effect minimal, the size of the filter mask is critical. In our case, the size of the mask is generally three times the standard deviation. This way, almost the whole Gaussian bell is taken into account and at the mask's edges weights will asymptotically tend to zero. The reason for choosing Gaussian kernel is that it is the most effective for our images. Gaussian filtering is calculating the convolution of the sigma kernel and each pixel on the image, and then sum up the

result as the output value of the pixel.

3.1.2 Binarization

After applying the Gaussian filter, the image is converted into a binarized image. In the literature, there are many works on the binarization of document images. In general, document binarization falls into the two categories: (i) global (ii) local. In a global approach, threshold selection results in a single threshold value for the entire image. Global thresholding has a good performance in cases where there is a good uniformity of the foreground and the background. However, in the case of historical documents, there exist degradations that diminish robustness of this class of binarization. Examples of degradations include shadows and non-uniform illumination, ink seeping, smear and strains. To deal with degradations, the current trend is to use local information that guides the threshold value pixelwise in an adaptive manner.

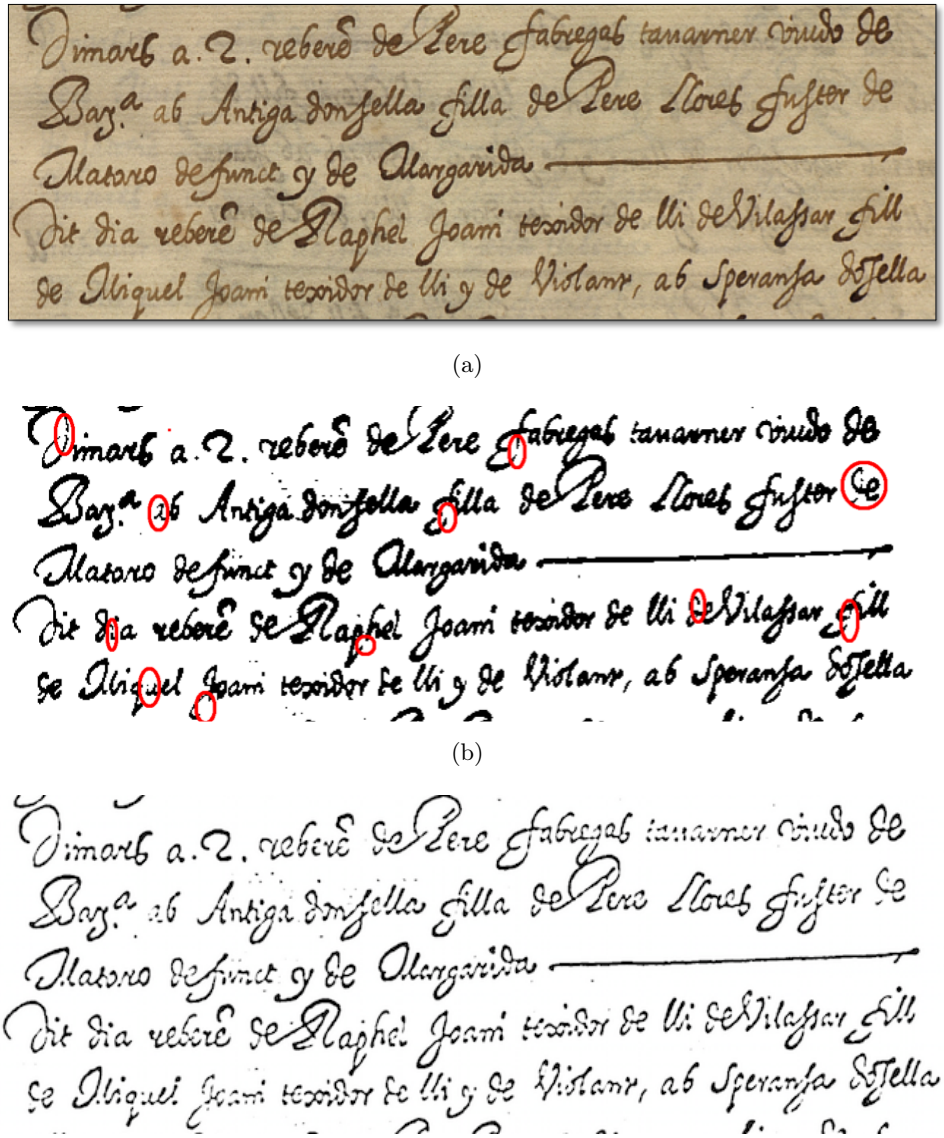
The selection of the binarization method relies on several aspects, such as the size of the image, the degradation degree of the ink, the noise. In our work, we mainly use two methods depending on the characteristics of the images. They are respectively Otsu's method [Otsu 1979] and the adaptive threshold method [Jain 1988].

Otsu's method is a classic approach to automatically perform clustering-based image thresholding, or the reduction of a grey-level image to a binary image. The algorithm assumes that the image to be thresholded contains two classes of pixels or bi-modal histogram (e.g. foreground and background), then calculates the optimum threshold separating those two classes so that their combined spread (intra-class variance) is minimal. In our case, Otsu's method is mostly applied to the word images with little degradation instead of the page images.

As for the adaptive threshold method, the threshold is a variable, which differs at each pixel on the image. The variable is calculated as the weighted average of the area minus a constant. The size of the area and the constant are empirically selected. The adaptive binarization is more robust and useful than the simple binarization with a fixed threshold, especially dealing with images with strong illumination or reflective grads.

Fig. 3.3 demonstrates part of a page binarization with different approaches. From the images it can be noted that the Otsu approach preserves more ink traces than the adaptive threshold approach. The amounts of noise on the binary images obtained by two approaches are more or less the same. However, it is interesting that the broken strokes caused by the binarization appear differently in the two approaches. Considering the fact that the performance of the adaptive approach is sensitive to the parameter configuration, the Otsu approach is a better choice for global binarization (i.e. an entire page).

It is known that binarization has a drawback: due to the conversion of the image from grey-scale to binary, high information loss will occur. However, without binarization the further operation cannot be processed. In the future, we will seek a method to extract directly from grey-level images without binarization.



(a)

(b)

(c)

Figure 3.3: Binarization of different methods (a) original image (b) Otsu (c) the adaptive threshold

3.1.3 Enhancement

The quality of historical document images is often unsatisfactory. Because of the nature of such documents, degraded images are often hard to read, have low contrast, and are corrupted due to various artifacts. When the quality of images is too poor, it is necessary to enhance its readability and comprehensibility by increasing the contrast. The most common strategy is to separate primarily textual foreground and background of images. Three popular methods are Otsu's thresholding tech-

nique, entropy technique and the minimal error technique. There are also other segmentation algorithms specifically dealing with the existence of specific noises, such as the work presented in [Wang 2001] for the bleed-through (from the backside of the paper). After separation, according to the characteristics of the degradation, different enhancement models are applied. Usually, a linear model is used adaptively to approximate the paper background [Shi 2004]. Then the document image is transformed according to the approximation to a normalized image that shows the foreground text on a relatively even background. However, the common degradations in document images are in many cases non-linear and so require special treatment. The approach proposed in [Agam 2007] including foreground segmentation, foreground enhancement, image enhancement and linear blending can handle the multiple degradations in a generic way with either a min-max or probabilistic model.

3.1.4 Line segmentation and baseline detection

3.1.4.1 Line segmentation

For many document analysis applications, line segmentation is a prerequisite. Many conventional approaches cannot be applied before the lines are segmented. Line segmentation allows ascenders and descenders of consecutive lines to be separated. In the manuscripts, it is observed that the lines consist of a series of horizontal components from left to right. In literature, there are many approaches for addressing the problem of handwritten text line segmentation, which can be categorized into two groups: bottom-up approaches and top-down approaches. The methodology of the top-down group is first to segment the document page into zones and then to segment the zones into lines. The approach that we applied is one of the most successful top-down approaches. It is the projection profile based approach using linear approximation and regression, which has been widely used in line and word segmentation for printed documents. It is very easy to understand and implement. Even though it is not efficient for non-standard and irregular page layout, since this phenomenon does not happen a lot in our case, we still choose to use it.

In the projection profile technique, a 1-D function of pixel values is obtained by projecting the image onto the vertical axis. Let f(x,y) be the intensity value of a pixel (x,y) in a grey-level image. Then the vertical projection profile is

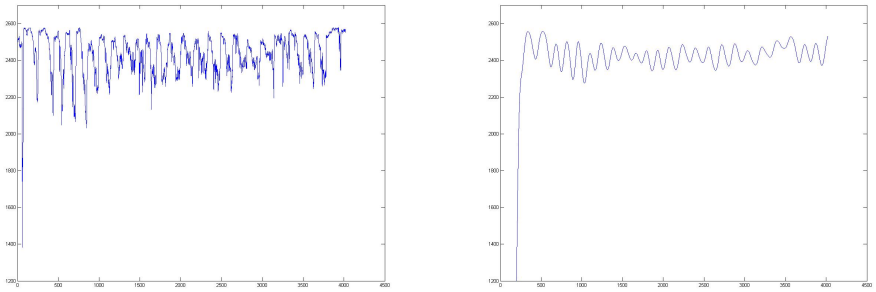
$$P(y) = \sum_{x=0}^{W} f(x, y) \tag{3.2}$$

where W is the width of the image. The distinct peaks in the profile correspond to the white spaces, and distinct local minima correspond to the text (black ink). Line segmentation, therefore, involves detecting the position of local maxima. However, due to noise and variation of handwriting, there are a lot of false local minima and maxima, as shown in Fig. 4.14 (a). Hence, the projection profile $P(y)$ needs to be smoothed. In our implementation, we use a Gaussian low-pass filter to eliminate

false alarms and reduce sensitivity to noise. A smoothed profile is shown in Fig. 4.14 (b). The local maxima are then obtained from the first derivative of the convolution of the projection function and the Gaussian low-pass filter function by solving for y such that,

$$P'(y) = P(y) * G(y) = 0 \tag{3.3}$$

This projection based line segmentation technique is robust to the variation of the size of lines.



(a) (b)

Figure 3.4: (a) The original projection profile (b) the smoothed projection profile

The problem of this method is the lack of adaptation to the variation of handwritten texts. Since this method was originally created for printed documents, it supposes that all the text lines are absolutely horizontal. However, this is not the same case in handwritten documents. Therefore, we can find that some words (in the red bounding boxes in Fig. 3.5) are separated into two lines due to big skews of lines. To avoid this problem, we plan to use another line segmentation method developed from the Hough transform.

3.1.4.2 Baseline detection

In Latin manuscripts, it is observed that the lines consist of a series of horizontal components from left to right. Projection profile techniques have been widely used in line and word segmentation for printed documents. The vertical density histogram is the simplest and most common method used to find the baseline and border line [Lecolinet 1995, Bozinovic 1987].

Four horizontal border lines are defined in [Bozinovic 1987]: top border line of word; top border line of baseline; bottom border line of the baseline; bottom border line of word. These lines are defined as l_1, l_2, l_3, l_4, which are located in top to bottom order on vertical y axis as shown in Fig. 3.6.

The calculation algorithm for finding these border lines goes through these steps:

1. $y_0(y)$ vertical density histogram of the word is calculated, $y \in [0, h_{tam}]$, h_{tam} is the height of the word and equals the histogram width.

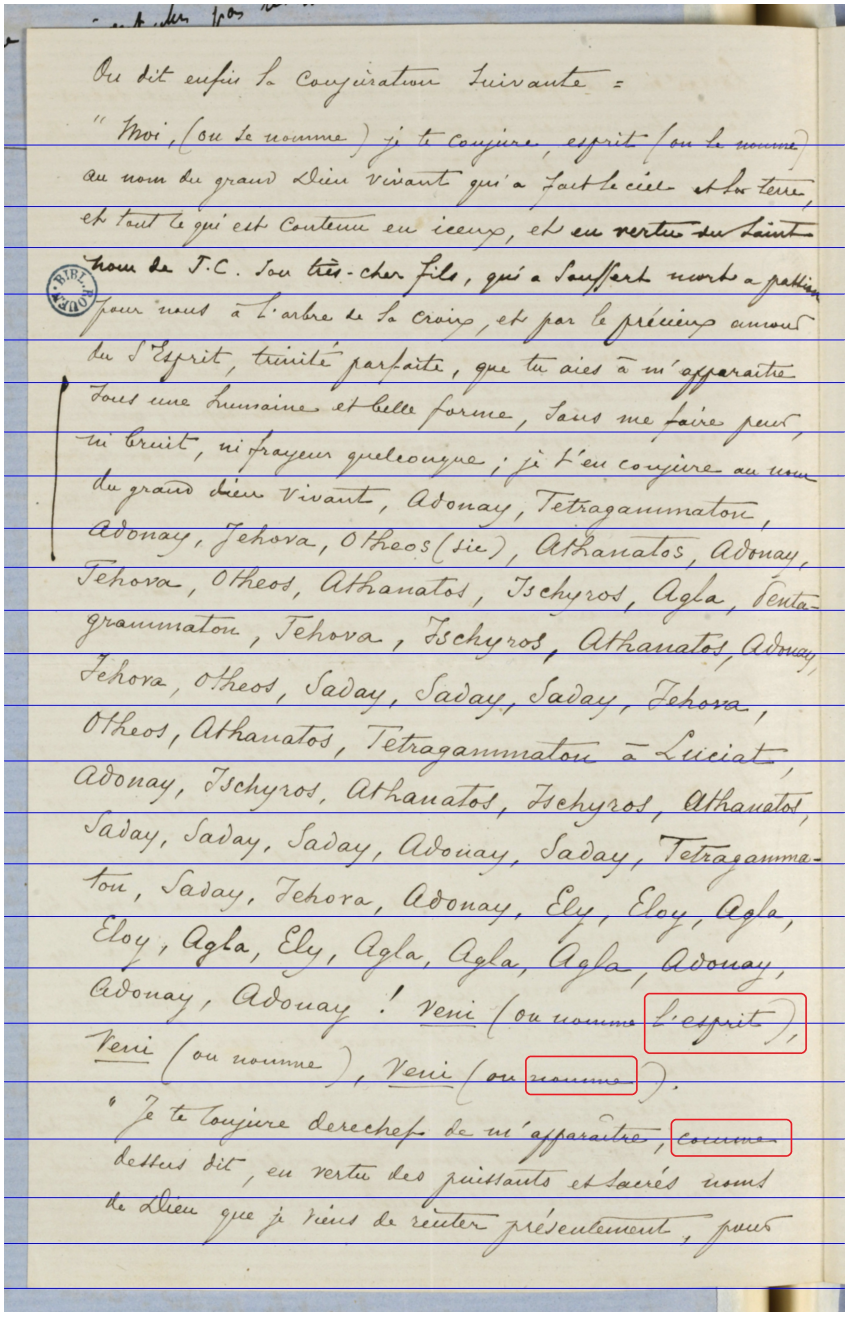


Figure 3.5: The result of line segmentation

Figure 3.6: Baseline of the word [Aida-zade 2009]

2. Smoothed $s_0(y)$ histogram is calculated:

$$s_0(y) = \sum_{i=-\Delta y}^{i=\Delta y} y_0(y+i) \tag{3.4}$$

Here Δy is the parameter. This parameter may vary depending on the size of the word image. In other words, it depends on parameters like author script, scanner, paper and pen quality. In this step, the raw histogram is smoothed around Δy pixels.

3. Starting from the top, the first position where $s_0(y)$ does not equal zero marked as l_1.

4. Starting from the bottom, the first position where $s_0(y)$ does not equal zero marked as l_4.

5. Starting from the center of the l_1 and l_4 towards the l_1, position where $s_0(y)$ is closer to zero, is refined l_1.

6. Starting from the center of the l_1 and l_4, position where $s_0(y)$ is closer to zero, is refined l_4.

7. The peak point where $s_0(y)$ has its maximum value is defined as p.

8. l_3 is defined as a local minimum value between p and l_4.

9. l_2 is defined as a local minimum value between l_1 and p.

10. if l_1 and l_2 are very close to each other, then

$$l_1 = l_2 - max((l_4 - l_3), (l_3 - l_2)) \qquad with\ |l_1 - l_2| < d_y \tag{3.5}$$

11. if l_3 and l_4 are very close to each other, then

$$l_4 = l_3 - max((l_2 - l_1), (l_3 - l_2)) \qquad with\ |l_3 - l_4| < d_y \tag{3.6}$$

In Fig. 3.6, the baseline detection is shown for the normal word with no slant or slope, whereas in practice, it is hard to find handwritten words where all letters are ideally written on the same horizontal line (Fig. 3.6). In this case, due to wrong

baseline determination, the ascenders or descenders defined by distance from the top and bottom borders of the body area, might also be identified incorrectly. As shown in Fig. 3.7, the first "y" letter's lower part is enough below the bottom border and that is why it is correctly detected as a letter with a descender. At the same time, the second and third "y" letters in the same word do not satisfy the described rule and therefore are not detected as letters with descender. To avoid this, we have to implement a more precise approach, which can identify different baselines for different parts when the text skews. There are already many existing solutions to this problem. We can explore later.

Figure 3.7: Baseline detection by the standard method [Aida-zade 2009]

3.2 Feature selection and extraction

3.2.1 Contour and skeleton based decomposition

It is not yet clear how the biological vision systems perform shape understanding. Nevertheless, two concepts have been popular in the shape domain, contour vs. skeleton (medial axis). It has been proved that contour-based approaches [Belongie 2002, Felzenszwalb 2007, Latecki 2000, Ling 2007] are often good at representing detailed shape information and somewhat robust against occlusion, but they are sensitive to articulation and non-rigid deformation [Bronstein 2008]. Skeleton-based approaches [Aslan 2008, Bai 2008, Sebastian 2004, Siddiqi 1999], on the other hand, can cope well with non-rigid deformations, but only carry rough structural information. The skeleton forms the centers of the maximal disks inside the contour boundary, and the radii of these maximal disks can be used to represent the thickness of an object. Similar shapes sometimes have similar skeleton structure and path. Fig. 3.8 illustrates an example where two shape instances of the same object show similar radius sequences. In the past, these two streams of work have been studied mostly in isolation. Sometimes, for a certain type of shape, a piece of contour segment (part) might be very informative to distinguish it from the others, even though it might be very small. It is worth mentioning that the contour information is somewhat implicitly linked with the skeleton, through the use of disks, in some existing works [Sebastian 2004, Zhu 1996, Katz 2003]. Nevertheless, very few studies have tried explicitly to address the issue of combining contour and skeleton in shape representation.

Indeed, contour and skeleton provide complementary information on handwriting that increases the coherence of the handwriting description. Inspired by the above

observations, our model takes advantage of both dimensions.

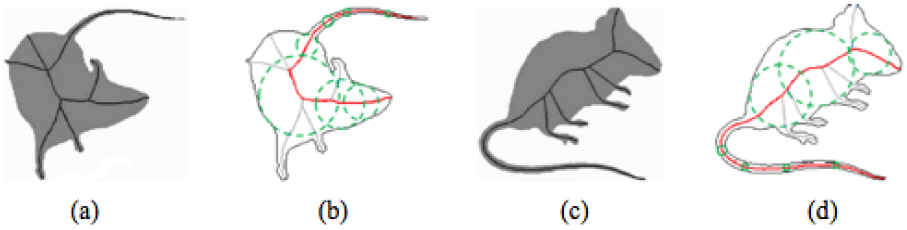


Figure 3.8: (a) (b) (c) (d) displays two non-rigid shapes which have the same radius sequence on the skeleton paths

3.2.1.1 Contour tracing

A canny edge detector is applied to extract edges. Then, a contour tracing method based on the border following algorithm proposed in [Abe 1985] is applied on the detected edge image (binary image). Border following is one of the fundamental techniques in the processing of digitized binary images. Since the outer borders and the inner borders have a one-to-one correspondence to the connected components of l-pixels and to the holes, respectively, the proposed algorithm created several rules to be followed to find the contours. The information to be extracted is the surrounding (8 neighbors) relation between two types of borders: the outer borders and the inner borders.

To extract the outline/boundary (as shown in Fig. 3.9(b)) of the text without inner contour (as shown in Fig. 3.9(c)), we have to apply an algorithm called Sandwich [Tan 2008]. The function of Sandwich algorithm is to fill the holes present in the text. The reason for choosing the outline instead of the contour is that we think the inner contour will become a distraction in the later steps for summarizing the characteristics of the samples. The procedure of the Sandwich algorithm is described below.

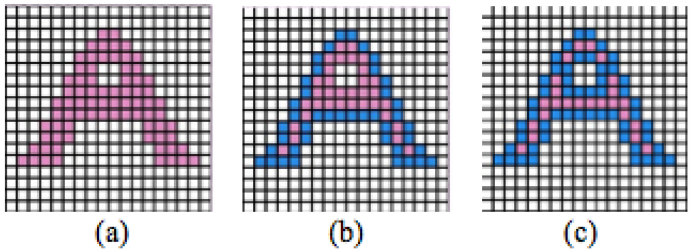


Figure 3.9: (a) A text sample (b) the outline of the sample (c) the contour of the sample

Scanning vertically (or horizontally) from top to bottom (or from left to right), a character white run can be located by a beginning pixel BP and an ending pixel EP corresponding to "01" and "10" illustrated in Fig. 3.10 ("1" and "0" denote white background pixels and black foreground pixels in Fig. 3.10). We scan word images vertically column by column. Clearly, two vertical white runs from the two adjacent scanning columns are connected if they satisfy the following constraint:

$$BP_c < EP_a \quad \& \quad EP_c > BP_a \tag{3.7}$$

where $[BP_c \quad EP_c]$ and $[BP_a \quad EP_a]$ refer to the BP and EP of the white runs detected in the current and adjacent scanning columns. Consequently, a set of connected vertical white runs form a white run component. Character holes can be detected based on the openness and closeness of the detected white run components. Generally, a white run component is closed if all neighboring pixels on the left of the first and on the right of the last constituent white run are text pixels.

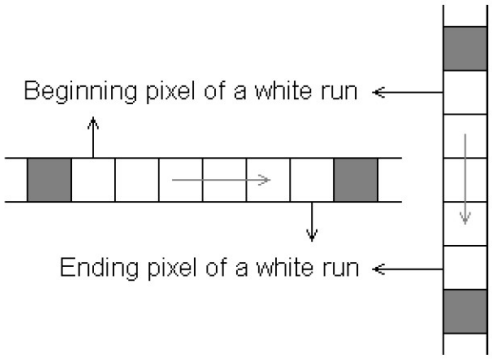


Figure 3.10: Illustration of the beginning and ending pixels of a horizontal and a vertical white run

It should be noted that due to document degradation, there normally exist a large number of tiny concavities along the character stroke boundary. As a result, a large number of character reservoirs of a small depth will be detected by the above vertical scanning process. However, these small reservoirs are not desired and can be identified by their depth. Besides, the initial Sandwich algorithm is designed for printed document images. Therefore it does not consider the variations in handwriting. For example, the holes are not closed in handwriting as shown in Fig. 3.11. In such case, the white run component is detected as open.

Three kinds of contours are extracted with different preprocessings. First of all, we apply the affine contour extraction method [Abe 1985] without using the Sandwich algorithm. The second is the contour extracted after applying the morphological operation "opening". Opening operation is a typical morphological skill used to eliminate small holes and blanks. It is the synthesis of erode and dilate operations. A structural element is created first as the mask. Then, this mask is applied to the whole image. The last one is the contour extracted after applying

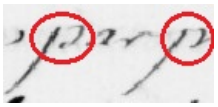
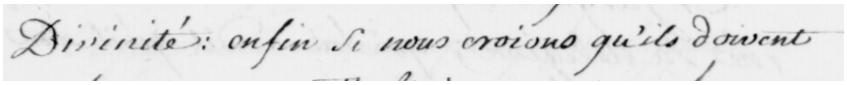


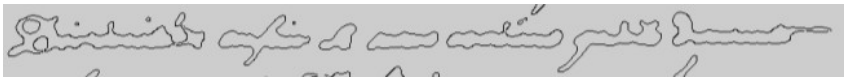
Figure 3.11: The illustration of the beginning pixel and ending pixel of a horizontal and a vertical white run

the Sandwich algorithm.



(a)

(b)



(c)

(d)

Figure 3.12: (a) The original image (b) the contour extracted directly from the original image (c) the contour extracted after applying morphology processing (d) the contour extracted after applying the Sandwich algorithm

As shown in Fig. 3.12, the contours traced with different processing vary a lot. The first contours (Fig. 3.12(b)) extracted directly by using the approach of [Abe 1985] are pretty detailed and precise, but the lines are broken at several places due to the degradation of the ink. Since this method is very sensitive, some noise introduced at this step cannot be ignored. There are two reasons that make us not choose to use this contour tracing method directly. One is because the method defines both inner and outer contours, which is not practical in our approach. The other reason is that the noise presented in the final tracing result is too great. The second contour image (Fig. 3.12(c)) presents the contours detected after the opening operation. Most of the inner contours are ignored and the outline still keeps some dominant curvature information. However, in further study, we found that even the contour extracted after applying the morphological operation is still not enough to represent the text. A more precise outer contour is required and it is the reason we employed the Sandwich algorithm in the end. As seen in Fig. 3.12(d), it is a balance between the first and second methods. So far, our efforts have not resulted in significantly

improved performance, but we believe that using only the outer contour can result in more stable features and better matching performance.

3.2.1.2 Skeletonization

The term of "skeleton" has been used in general to denote the representation of a pattern by a collection of thin (or nearly thin) arcs and curves. It is useful to describe properties of a shape. It provides a simple and compact representation of shapes that preserves the topology of the objects. There are a number of techniques to obtain skeletons of handwriting samples. Applicability of a particular technique depends on the problem in hand. Distance transform method is one of the early widely used technologies, firstly explored by H. Blum's research [Blum 1967]. The distance transform is defined for each point of an object as the smallest distance from that point to the boundary of the object. This process of computing the distance transform produces a distance map whose ridges projected back on to the image plane generate skeleton-like structures. For a grey-level image, a distance transform cannot be performed directly, since the object boundary locations are not known. Therefore, binarization preprocessing is necessary. Skeletons and median lines of objects can be computed by finding the local maxima of the distance map. The object can be entirely reconstructed by replacing each point of the skeleton by a discrete disc with a radius given by the distance transform. The skeleton extracted by the distance transform is very sensitive to deformation of contours, which we usually encounter. Besides, thinning is also popular for skeletonization, mainly because of its implementation simplicity and high computational speed. A lot of skeleton and stroke extraction methods have been proposed based on thinning. Such methods use thinned image as the first approximation to the desired skeleton and then use various methods to correct junctions and branches, recover loops and sometimes also to recover the writing sequence. Since these methods need to perform a binarization process before extracting the skeleton, when the degradation of the images is severe, the skeleton is often distorted. Later, skeletonization approaches that do not require any segmentation or binarization were developed. In [Lebourgeois 2007], the presented skeletonization approach, based on color images, uses the diffusion of gradient vectors to obtain a thin and continuous axis of the main contrasted objects. In addition, H. Daher et al. [Daher 2010a] propose a method to extract the centerline of the ink trace directly from the grey-level image by analyzing the orientation and the stroke thickness at different points of the handwritten shapes. Both binarization-free methods [Lebourgeois 2007, Daher 2010a] are robust to the degradation of the ink trace, the background noise and the irregularity of the ink. However, compared to distance transform and thinning approaches, these methods do not require any segmentation but require a laborious parameterization.

Under an overall consideration of the complexity of the approach and our situation, we decide to use thinning algorithms for skeletonization. Later in this section, we give a detailed introduction on the thinning algorithm used in our algorithm.

In this section, we consider some general aspects of iterative thinning algorithms

or, more precisely, the algorithms that delete successive layers of pixels on the boundary of the pattern until only a skeleton remains. The deletion or retention of a (black) pixel p would depend on the configuration of pixels in a local neighborhood containing p.

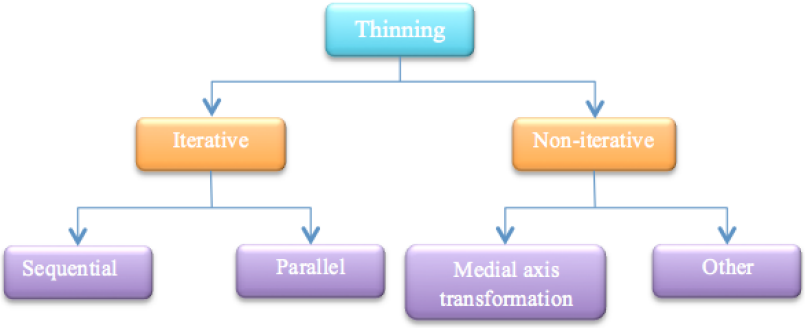


Figure 3.13: The category of thinning methods

According to the way they examine pixels, these algorithms can be classified as sequential or parallel, as shown in Fig. 3.13. In a sequential algorithm, the pixels are examined for deletion in a fixed sequence in each iteration, and the deletion of p in the nth iteration depends on all the operations that have been performed so far. On the other hand, in a parallel algorithm, the deletion of pixels in the nth iteration would depend only on the result that remains after the *(n-1)*th; therefore, all pixels can be examined independently in a parallel manner in each iteration. The results that thinning methods produce have many drawbacks, such as erroneous branches and other artifacts, affecting subsequent feature extraction.

3.2.1.3 Zhang & Suen's thinning algorithm

Zhang's algorithm [Zhang 1984] is a fast parallel thinning algorithm applied on binary images. It consists of two subiterations: one aims at deleting the southeast boundary points and the northwest corner points while the other one is aimed at deleting the northwest boundary points and the southeast corner points. End points and pixel connectivity are preserved. Each pattern is thinned down to a "skeleton" of unitary thickness. The method for extracting the skeleton of a picture consists in removing all the contour points of the picture except those points that belong to the skeleton. We divide each iteration into two subiterations.

In the first subiteration, the contour point P_1 is deleted from the digital pattern

P_9	P_2	P_3
P_8	P_1	P_4
P_7	P_6	P_5

Figure 3.14: The order of nine pixels in a 3x3 window

if it satisfies the following conditions:

$$2 \leq B(P_1) \leq 6 \quad \text{(a)}$$
$$A(P_1) = 1 \quad \text{(b)}$$
$$P_2 \times P_4 \times P_6 = 0 \quad \text{(c)}$$
$$P_4 \times P_6 \times P_8 = 0 \quad \text{(d)}$$

where $A(P_1)$ is the number of 01 patterns in the ordered set P_2, P_3, P_4, ... , P_8, P_9 that are the eight neighbors of P_1, as shown in Fig. 3.14, and $B(P_1)$ is the number of nonzero neighbors of P_1, that is

$$B(P_1) = P_2 + P_3 + \cdots + P_8 + P_9 \quad (3.8)$$

If any condition is not satisfied, then P_1 is not deleted from the picture.

In the second subiterations, only conditions (c) and (d) are changed as follows:

$$P_2 \times P_4 \times P_8 = 0 \quad \text{(c')}$$
$$P_2 \times P_6 \times P_8 = 0 \quad \text{(d')}$$

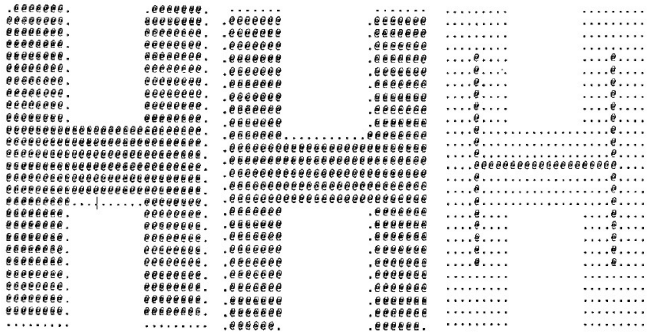
and the rest remains the same.

By conditions (c) and (d) of the first subiteration, it removes only the southeast boundary points and the northwest corner points which do not belong to an ideal skeleton. Similarly, the second subiteration deletes northwest boundary points or southeast corner points. Fig. 3.15 gives one example of Zhang & Suen's thinning algorithm.

Fig. 3.16 presents a small demonstration of the skeletons extracted by different approaches. As we can see, compared to the thinning algorithm, the diffusion-based approach obtains a more continuous ink trace, such as the descender part of the letter "g". However, the problem of the diffusion-based approach is that divergences often occur at the end of the strokes. Even though this phenomenon can be improved by the set-up of the parameters, it cannot be eliminated. In an overall consideration, we choose the thinning approach - Zhang & Suen's algorithm - as our skeletonization method.

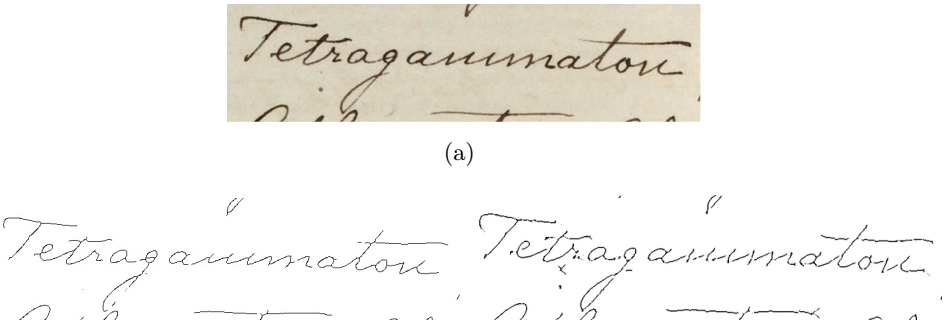
3.2.1.4 Tensor voting for continuous skeleton

Since we are also working on historical documents, the degradation of the ink happens quite often. After binarization, the ink trace might not be continuous anymore



(a) (b) (c)

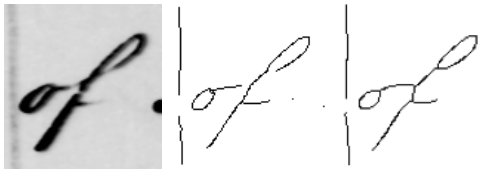
Figure 3.15: Illustration of thinning the character "H" by subiterations



(a)

(b) (c)

Figure 3.16: (a) Original word image (b) Zhang & Suen's approach (c) diffusion-based approach [Lebourgeois 2007]



(a) (b) (c)

Figure 3.17: (a) Original grey-level image (b) skeleton extracted by Zhang & Suen's algorithm (c) recovered skeleton by the tensor voting method

at the degraded parts (Fig. 3.17(b)), which leads to a broken skeleton. In order to recover the real topology of the handwriting, we adapt the tensor voting method [Medioni 2000] to connect the closest discontinuous parts of the skeleton. It allows us to merge the most common kinds of discontinuities found in the skeleton. Tensor voting is a rich expansion of perceptual organization techniques, which has been applied to address many individual problem instances in early vision and visualization, such as the motion flow estimation for a scene with multiple moving objects and the shape reconstruction from stereo from the same perspective.

Initially, tensor voting technique is developed in order to reconstruct shapes from point clouds. It allows the simultaneous communication among various types of tokens. Moreover, the information exchange between tokens uses tensor fields rather than scalar ones as in other methods. The communication is thus much richer. The methodology is grounded on two elements: tensor calculus for data representation, and linear tensor voting for data communication.

- Tensor representation

  Points can be represented simply by their coordinates. A local description of a curve is given by the point coordinates, and its associated tangent or normal. A local description of a surface patch is given by the point coordinates and its associated normal. Here, however, we do not know in advance what type of entity (point, curve, surface) a token may belong to. Furthermore, because features may overlap, a location may actually correspond to multiple feature types at the same time. To capture first order differential geometry information and its singularities, a second order symmetric tensor is used. It captures both the orientation information and its confidence, or saliency. Such a tensor can be visualized as an ellipse in 2-D, or an ellipsoid in 3-D. Intuitively, the shape of the tensor defines the type of information captured (point, curve, or surface element), and its size represents the saliency.

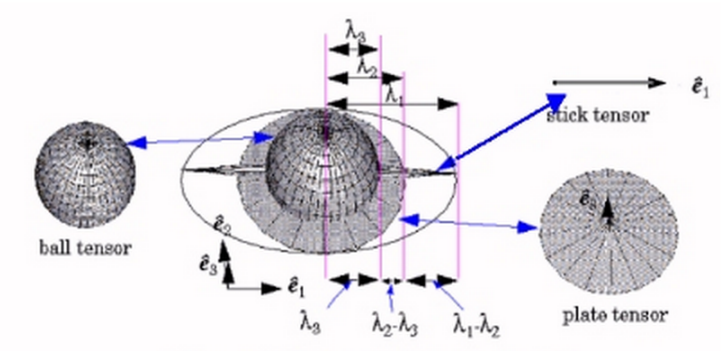


Figure 3.18: Tensor decomposition [Medioni 2002]

Any second order symmetric tensor, therefore, can be decomposed in the basis tensors (stick, plate in the 2-D case). The tensorial representation allows

for a unified treatment of inliers of smooth structures, depth and orientation discontinuities and noise before any hard decisions are made for the role of each point. This property is extremely useful in perceptual grouping when the role of a point cannot be determined accurately from the initial data. Instead, the tensor represents the likelihood of all potential roles the point can play simultaneously.

- Tensor voting

  The input tokens are first encoded as tensors. These initial tensors communicate with each other in order to derive the most preferred orientation information (or to refine the initial orientation if given) for each of the input tokens (token refinement or sparse tensor vote), and to extrapolate the inferred information at every location in the domain for the purpose of coherent feature extraction (dense extrapolation or dense tensor vote). Each token is first decomposed into the basis tensors, and then broadcasts its information. A voting field for each basic component is used to seek the orientation and magnitude of the votes cast. All voting fields are based on the fundamental 2-D stick voting kernel. As seen in Fig. 3.19 the orientation of the stick vote is normal to the smoothest circular path connecting the voter and receiver that is perpendicular to the normal (the stick) at the voter's location. The magnitude of the vote decays with distance and curvature according to the following equation:

$$V(d, \rho) = e^{-\frac{d^2+cp^2}{\sigma^2}} \tag{3.9}$$

  where d is the distance along the smooth path, c is the curvature of the path and ρ is the scale of the voting field that essentially controls the size of the voting neighborhood and the strength of the votes. Given the fundamental 2-D stick voting kernel, the other voting fields can easily be derived. The 2-D ball voting field can be obtained by integrating at each position the votes cast by a rotating stick.

- Vote analysis

  Vote accumulation is performed by tensor addition or equivalently by addition of 3x3 matrices (in the 3-D case), therefore it is computationally inexpensive. After voting has been completed, we can compute the eigen system of the resulting tensor which encapsulates all the information propagated to the location. The stick component, which is parallel to the eigenvector corresponding to the largest eigenvalue and its magnitude (saliency) is equal to the difference of the largest and second largest eigenvalue, encodes the likelihood of the point belonging to a smooth surface. The plate component spanned by the eigenvectors corresponding to the two largest eigenvalues and whose saliency is the difference of the second and third eigenvalue encodes the likelihood that the point belongs to a smooth curve or a surface junction. Finally, the ball component that has no preference of orientation and whose saliency is equal

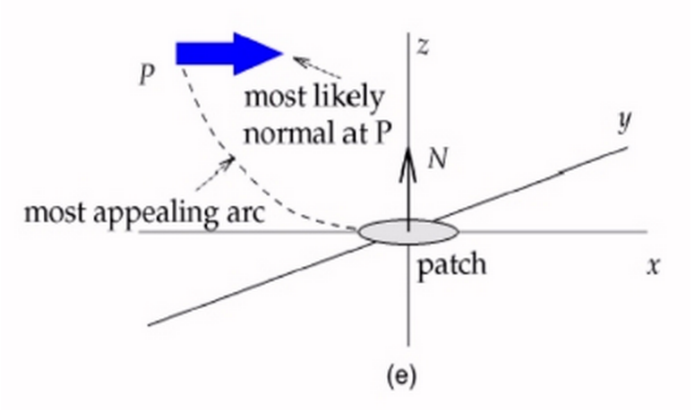


Figure 3.19: Vote generation [Medioni 2002]

to the smallest eigenvalue encodes the likelihood of the point being a junction. Assuming that noisy points are not organized in salient perceptual structures, they can easily be identified by their low saliency.

- Structure extraction

  Structures can be extracted after a dense tensor vote has been performed. The difference is that in this case votes are also collected at previously unoccupied locations and surface, curve and junction saliency maps are filled in. Surfaces and curves are extracted as the local maxima of surface and curve saliency respectively, while junctions can be extracted as the local maxima of junction saliency without any form of propagation. Since calculating votes for every location in the volume containing the data points is pointless and impractical, surface and curve extraction begins from seeds, location with highest saliency, and voting is performed only towards the directions indicated by the surface normals and curve tangents.

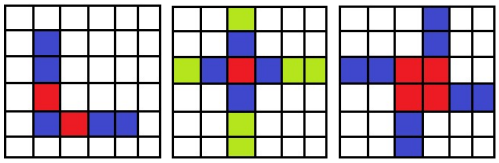
3.2.2 Structural point detection

After obtaining the skeleton of the text, we start to detect structural points. The structural points in our application are referred to three types of points. The first one is starting/ending points. The second type is branch points including junction and crossing points. The last one is high-curved points.

Since the skeleton obtained from the last step is one-pixel wide, it is easy to detect a starting/ending point. For each black pixel (skeleton pixel), we apply a 3x3 mask to check the nearest 8 neighbors of the pixel. If there is only one black pixel among 8 neighbors, the reference pixel is considered as a starting/ending point.

Concerning the branch points, it is not easy to design comprehensive rules for detecting, because the skeleton is connected by an 8-connected relationship. If the graph is connected by an 8-connected relationship, it is not always the case that

only the pixel having only two neighbors is part of a line segment stem, or that the pixel is a junction if it has more than two neighbors. Fig. 3.20(a) shows an L-shape line segment, of which the red pixels have three neighbors according to an 8-connected relationship. But they cannot be regarded as junctions. Fig. 3.20(b) shows a cross graph. Only the red pixel should be detected as a junction, even though the blue pixels also have more than three neighbors. Fig. 3.20(c) shows another scenario where the number of the neighbors is again not able to correctly detect the line segment stem and junctions. Though it is possible to elaborate a set of comprehensive rules to handle the above scenarios, we find the Hit-and-Miss transformation can easily detect the junctions and crossings.



(a) (b) (c)

Figure 3.20: (a) L-shape line segment (b) cross-figure graph (c) another scenario

The Hit-and-Miss transform [Zhao 1991] is a basic binary morphological operation, which is generally used to detect particular patterns in a black-and-white image. The template, a.k.a., kernel, which is a small matrix, defines the particular patterns, with 1 meaning foreground pixel, 0 meaning background pixel, and "don't care" pixels. For instance, Fig. 3.21(a) gives a T-junction template. The template slides on the input binary image pixelwise, does AND operation for all the pixels inside template expect those "don't care" pixels. The corresponding pixel in the output binary image is designated as the outcome of the "binary convolution". "True" means the template is exactly matched. Triple junction is a basic junction type. Fig. 3.21(a), (b), (c) give three types of triple junction templates.

The starting/ending points and branch points are informative about the topology of the shape, while the skeletal curves are informative about its geometry. This is true, since the skeleton approximates the loci of the center of the pen during the writing. The geometry of the skeletal curve is given by its curvature:

$$k = \frac{x'y'' - y'x''}{(x'^2 + y'^2)^{\frac{3}{2}}} \tag{3.10}$$

The most salient parts of a curve are given by the curvature extremes. Nevertheless, it is very hard to use the definition of Equation 3.10 on digital curves to find the curvature extremes as the curves are usually not smooth. Smoothing the curve could suppress extremes. Therefore it is not an optimal solution. The curvature at point t of a curve C is instead estimated using the angle between two vectors $\overrightarrow{a}$

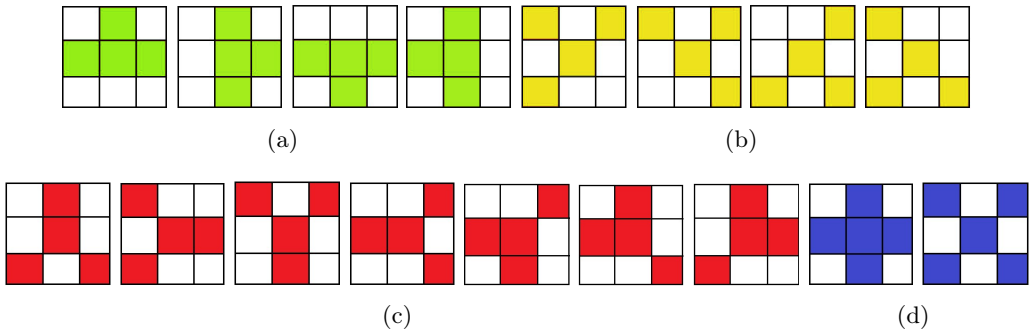


Figure 3.21: (a) Type1: triple junction (T-junction) template (b) type2: triple junction template (c) type3: triple junction template (d) crossing point templates. Red and blue pixels indicate foreground pixels. White pixels indicate "don't care" pixels

and $\overrightarrow{b}$:

$$\begin{aligned} \overrightarrow{a}(t) &= C(t+k) - C(t) \\ \overrightarrow{b}(t) &= C(t-k) - C(t) \\ \theta &= \arccos \frac{(\overrightarrow{a} \cdot \overrightarrow{b})}{(|\overrightarrow{a}| \cdot |\overrightarrow{b}|)} \end{aligned} \tag{3.11}$$

where k is a parameter that defines the support of a given point. The angle can be easily thresholded to select high curvature points and the non-maxima can be removed. In the implementation, we use k=4 and the threshold equals $\pi/4$.

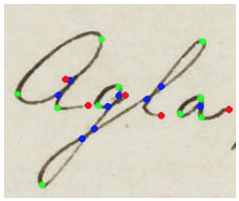
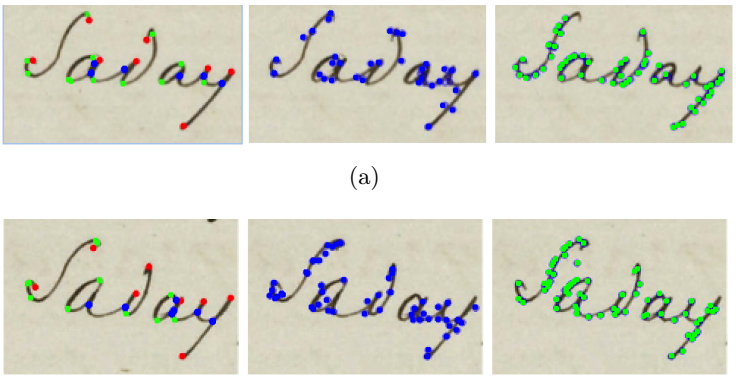


Figure 3.22: Example of structural point detection (red — starting/ending points, green — high-curved points, blue — branch points)

As stability is an important standard for evaluating the point detector, we make two comparison experiments to investigate the inter and intra stability of interest point detection. Fig. 3.23 gives a qualitative example of the point detection for two instances of the same word using different detectors. It is obvious that the main difference among structural points, DoG points and Hessian points is the number. For the same sample, the number of structural points is much less than DoG and Hessian points, even after the elimination of some artificial DoG and

Hessian points caused by noise. Another observation is that for different instances of the same word, the stability of structural points is higher than the other two kinds of points. It is because the structural point detection is completely based on the natural configuration of characters. The most variant factor for structural point detection is writing style. Otherwise locations of points are consistent. Since DoG and Hessian points are identified upon gradient variation, the detection is relatively random in terms of single point. Besides, in the context of historical handwriting images, many factors can cause gradient variation, such as degraded ink trace, stains and so on.



(a)

(b)

Figure 3.23: (a),(b) are the interest points detected on two instances of the same word. The first column shows the detection of structural points; the second column shows the detection of DoG points; the third column shows the detection of Hessian points

Fig. 3.24 presents how the point detection changes as the degradation of images increases. It can be seen that as the image becomes more degraded, the numbers of DoG points and Hessian points are sharply decreased, while the number of structural points maintains the same. The last column in Fig. 3.24 gives an interesting result, which reveals that structural points are actually the most stable part of DoG and Hessian points. Besides visual observation, we also calculate the repeatability rate for each detection to have a quantitive inspection. The repeatability rate is defined as the number of points repeated between two images with respect to the total number of detected points [Bay 2008]. For the case in Fig. 3.23, the repeatabilities are $R_{structural} = 80\%$, $R_{DoG} = 65.71\%$ and $R_{Hessian} = 72.73\%$, respectively.

3.2.3 Shape context

In the case of handwriting recognition and retrieval, contextual information has been proved to be an important factor by many existing works. The scope of the context can be very varied. It can come from neighboring graphemes and also from the whole word image. In order to extract and describe such important information, we choose to adapt the shape context descriptor in our application. Shape context

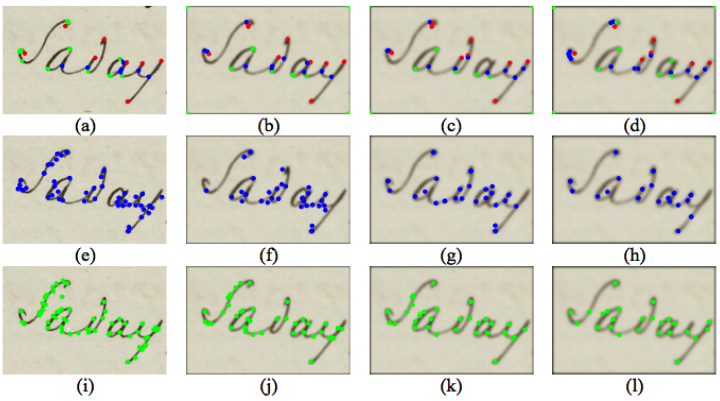


Figure 3.24: Point detection as degradation increases: the first row is structural point detection; the second row is DoG point detection; the third row is Hessian point detection. The extent of degradation increases from left to right in terms of column.

is a local descriptor invented by S. Belongie et al. [Puzicha 2000, Belongie 2002]. It was originally introduced for correspondence recovery and shape-based object recognition. The shape context descriptor at a given point captures the distribution over relative positions of other shape points and thus summarizes the global shape in a rich descriptor.

The use of a shape descriptor is motivated by the nature of documents, which are most often in grey-scale, or in binary. A shape descriptor is well suited to capture information in such documents. The shape context descriptor contains rich information about the local geometry of the object and has a good performance on partially occluded objects [Belongie 2002]. Besides, the study of J. A. Rodríguez et al. [Rodríguez 2007] has also provided a convincing proof of the feasibility of Shape Context. It has been applied to symbol recognition with different resolutions [Su 2011].

The Shape Context (SC) is a point descriptor, so there is no difference in the basic procedure of matching objects between with Shape Context and with other point descriptors such as SIFT [Lowe 2004], SURF [Bay 2008], and BRIEF [Calonder 2010]. The general workflow of matching objects with Shape Context is illustrated in Fig. 15. After selecting points, a SC descriptor is generated for each key point. Then, by calculating the distance among points, the corresponding points between two objects are found. The next step is to estimate the plane transformation between two objects with the correspondences found in the previous step. The purpose of this step is to eliminate outliers and to keep the best matching points.

In the application of the Shape Context, the shape of the object is represented as a finite set of points. It is composed of a discrete set of points sampled from internal or external contours of the object, giving a set $P = p_1, ..., p_n$, $p_i \in R^2$, of n points. They need not, and typically will not correspond to keypoints such as maxima of curvature or inflection points. In the original approach with Shape

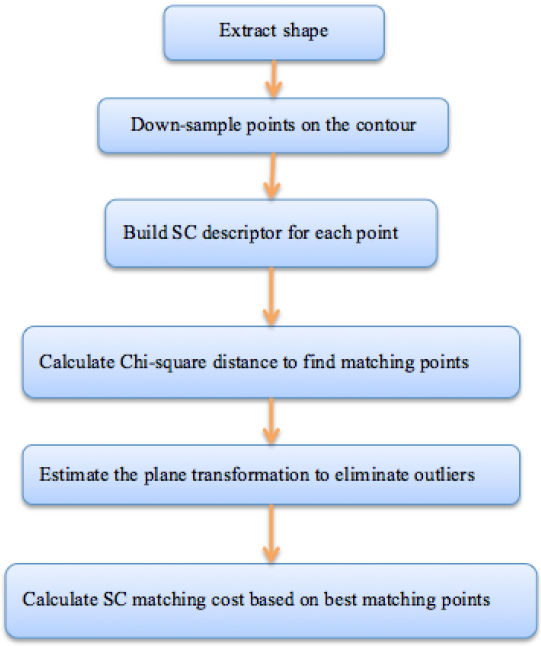


Figure 3.25: The workflow of matching objects with the Shape Context

Context descriptor, the subset of points is down-sampled from the contour roughly with the same interval. Assuming contours are piecewise smooth, we can obtain as good an approximation to the underlying continuous shapes as desired by picking n to be sufficiently large.

For each point p_i on the first shape, we consider the set of vectors originating from a point to all other sample points on a shape. These vectors express the configuration of the entire shape relative to the reference point. Obviously, this set of n-1 vectors is a rich description, since as n gets large, the representation of the shape becomes exact.

The full set of vectors as a shape descriptor is much too detailed since shapes and their sampled representation may vary from one instance to another in a category. The distribution over relative positions is defined as a more robust and compact, yet highly discriminative descriptor. For a point p_i on the shape, we compute a coarse histogram hi of the relative coordinates of the remaining n-1 points.

$$h_i(k) = \#\{q \neq p_i : (q - p_i) \in bin(k)\} \tag{3.12}$$

This histogram is defined to be the Shape Context of pi. Bins that are uniform in log-polar[2] space make the descriptor more sensitive to positions of nearby sample points than to those of points father away. An example is shown in Fig. 4.31.

Consider a point p_i on the first shape and a point q_j on the second shape. Let $C_{ij} = C(p_i, q_j)$ denote the cost of matching these two points. As Shape Contexts

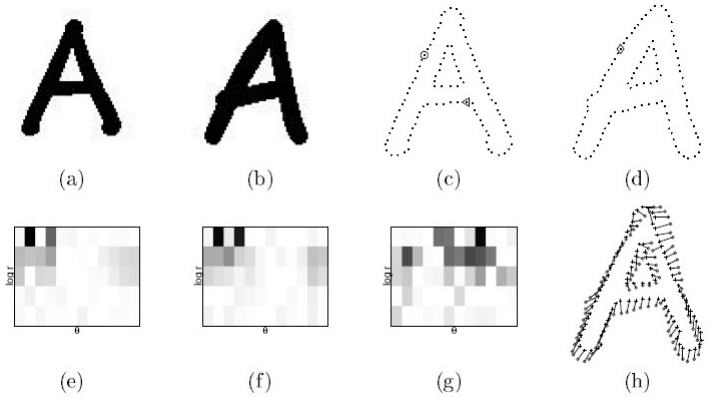


Figure 3.26: The illustration of Shape Context computation and matching: (a)(b) the original objects. (c)(d)Shape edge points. (e)(f)(g) Shape contexts for reference points marked as in ∘,◇,◁ in(c)(d). Each shape context is a log-polar histogram of the coordinates of the rest of the point set measured using the reference point as the origin. In this illustration, 5 bins are used for $\log\Upsilon$ and 12 bins for θ. (h) Correspondences found using bipartite matching, with weights defined by the χ^2 distance between histograms

are distributions represented as histograms, it is natural to use Chi-square distance χ^2 test statistic:

$$C_{ij} \equiv C(p_i, q_j) = \frac{1}{2}\sum_{k=1}^{K}\frac{[h_i(k) - h_j(k)]^2}{h_i(k) + h_j(k)} \tag{3.13}$$

where $\mathrm{h}_i(k)$ and $\mathrm{h}_j(k)$ denote the K-bin normalized histogram at p_i and q_j respectively.

We measure shape context distance between shapes P and Q as the symmetric sum of shape context matching costs over best matching points, i.e.

$$D_{sc}(P,Q) = \frac{1}{n}\sum_{p\in P} arg\min_{q\in Q} C(p, T(q)) + \frac{1}{m}\sum_{q\in Q} arg\min_{p\in P} C(p, T(q)) \tag{3.14}$$

where $\mathrm{T}(\cdot)$ denotes the estimated TPS shape transformation, $n = |P|$ and $m = |n|$.

In the original recognition algorithm using SC descriptor, all the points used to calculate the SC cost are from the contour of object. By down-sampling the contour with a certain interval, each object is represented by a subset of points from contour. Since the theory of this algorithm is to extract contour information of object and describe it by recording the relative location of points, its efficiency is based on

- Distinctiveness of contour
- Selection of sub-points

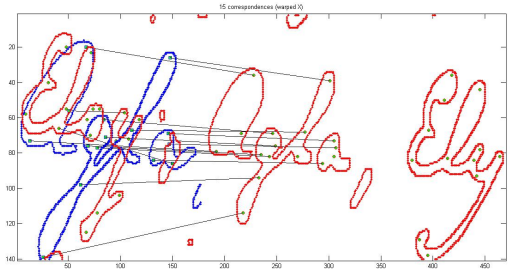


Figure 3.27: The correspondences between points in the query and the test sample using SC descriptor and an example of three SC histograms

If the contour of the object is too blurred, general, indistinct, the method will lose its power. In our case, the contour extraction is applied to the outline of the text. Only the boundary is extracted. Since the handwritten text is quite flexible and the stroke can be very thin compared to the printed text, the contours extracted are very cursive. Within one word, there are usually a lot of small ups and downs in the contours. This becomes a distraction when comparison is conducted. Besides, when the quality of images is too low, the contour is saw-edged and we cannot get enough points from the contour, both of which phenomena decrease the final performance.

Therefore, we need to bring some topological information into the representation of words, which should be relatively stable and invariant to the quality of images and the writing style. Inspired by this, we decide to use the structure points in our model. We still keep using the SC descriptor, but the selection of reference points is different. We choose the points most typical and intuitive of the characters, such as the structural points introduced in the previous section. Then calculate the SC cost based on the interest points. One shortage of this model is that the amount of interest points is quite few, which entails that the comparison result is too sensitive to the detection of interest points and the tolerance of outliers is very limited.

The above proposition substantiates the importance of combining properties from different aspects as well as the need of a more complex mechanism for feature fusion. How to integrate the topological and morphological properties of handwriting as an entirety in an efficient way and to take best advantage of two aspects turns out to be a vitally important step. To solve this problem, we have developed a comprehensive representation model based on the graph.

3.2.4 Point feature performance evaluation (SIFT, SURF, Shape Context)

To study the performance of different descriptors, we use the same points with different descriptors to compare two images. Fig. 3.28 shows that the wrong matching of points appears much more with SURF than with Shape Context and SIFT.

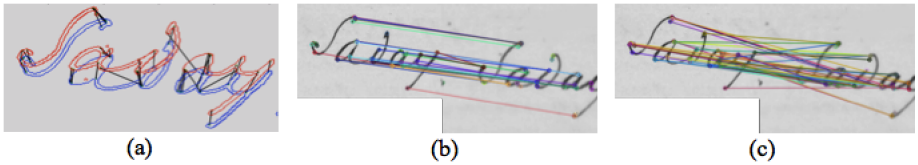


Figure 3.28: Matching examples using different descriptors with structural points: (a) Shape Context, (b) SIFT, (c) SURF

The purpose of the experiments presented in this section is to explore the performance of different combinations of point detectors and local descriptors. Based on the observation and analysis of experimental results, discussion and conclusion are made.

We conduct the experiments on a dataset extracted from the humanity corpus containing more than 200,000 letters written by different French philosophers. Our dataset contains 4 collections of different writing styles. There are 11 pages containing approximately 2000 words. From all the word classes (a class is a set of the same word instances) in the dataset, we select a subset of 9 classes. Since the dataset consists of authentic historical letters, the number of samples per class varies from 2 to 12. Each sample in the classes is used as a query and this makes a total of 51 queries. Fig. 3.29

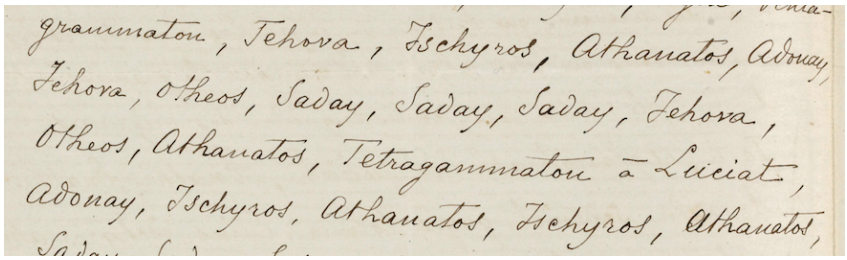


Figure 3.29: Sample of the dataset used in the experiment

The evaluation protocol is the following: for each query, we rank all the regions of interest that are selected from the coarse representation. A region is classified as positive if it overlaps more than 30% with the annotated bounding box in the ground truth, and negative otherwise. We combine the retrieved regions of all the documents and rerank them according to their scores. For each word class, we report the Average Precision (AP) and Average Recall (AR) for 5 ranks (rank 3, rank 5, rank 10, rank 20 and rank 30), which are standard measures in retrieval systems. Overall results (Fig. 3.30) are evaluated by computing the mean Average Precision (mAP) and the mean Average Recall (mAR) over the 9 classes. If we exclude the influence of interest points and focus only on the local descriptors, the results show that SC gives the best performance among three descriptors for all kinds of interest

points. Compared to SIFT and SURF, SC is built on a relatively broader region, which relatively encodes the spatial information in terms of the entire image. SIFT and SURF descriptors focus only on local spatial information (within the patch), which becomes a clear disadvantage. M. Rusiñol's work [Rusiñol 2011] has also proved the necessity of using spatial distribution of features by applying the SPM (Spatial Pyramid Matching) method.

Overall, SC is a very powerful descriptor for handwriting regardless of the selection of points. Fig. 3.30 gives an average distribution of scores based on all word classes of different writing styles. However, the results show that the performance is quite writing-style-dependent. For the compact and solid handwriting styles without any elongations, it has been observed that DOG or Hessian points associated to SIFT descriptors are more efficient. One example is given in Fig. 3.31. In this case, contextual information carried in SC does not help substantially the retrieval because the distinguishability of information is sharply reduced. For the cursive handwriting style, SC based on structural points (SSC) can produce a more relevant description because a good space occupancy associated with a distinguishing shape is one necessary condition to the application of SSC. Moreover, the relevance between precision and recall for each combination shows that, depending on the category of the query and the category of the writer, it is meaningful to select the method that gives individually the best performance. These results show that interest points carry very distinctive information. When the data is poor, interest points must be numerous and even redundant. On the other side, when the data is abundant, it is efficient to consider structural points that simplify the representation and the overall computation cost.

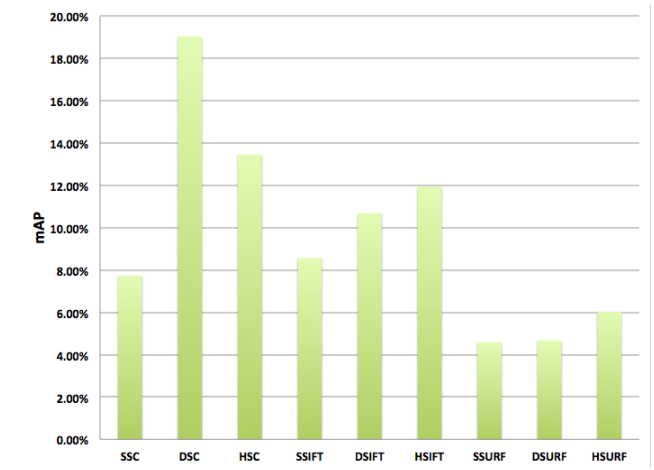
3.3 Graph-based representation model

3.3.1 Graph-based approaches for pattern recognition

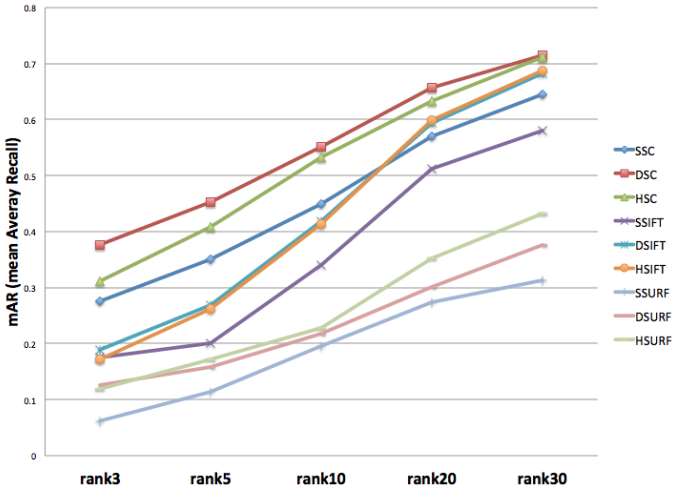
Graphs, representative of mathematics, are remarkably versatile tools for modeling. There are two main approaches in pattern recognition application. One is a statistical approach, where patterns are represented by means of feature vectors, and the other is a structural approach, in which patterns are represented by symbolic data structures [Bunke]. Graph-based representation is one of structural approaches for pattern recognition. It has been widely used in applications related to geometrical pattern recognition and indexing, such as symbol recognition, medical and biomedical applications, action recognition and so on.

- Symbol recognition

  The usage of graph for symbol spotting and symbol recognition has existed for decades. It has achieved a huge success by representing graphical documents with graph representations. The approaches proposed by Messmer and Bunke [Messmer 1996] and Lladós et al. [Lladós 2001] were among the first few approaches of symbol spotting using graph representation. Afterwards,

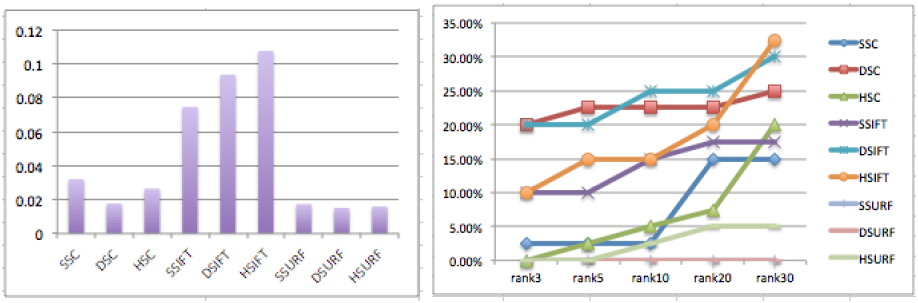


(a)



(b)

Figure 3.30: Overall result of mAR and mAP (SSC-Structural points + Shape Context, DSC-DoG points + Shape Context, HSC-Hessian points+Shape Context, SSIFTStructural points+SIFT, DSIFT-DoG points + SIFT, HSIFT-Hessian points+SIFT, SSURF-Structural points+SURF, DSURFT-DoG points+SURF, HSURF-Hessian points+SURF)



(a) (b)

Figure 3.31: Results of the query "c'est" in collection 2

Luqman et al. [Luqman 2011] also proposed a graph embedding based symbol spotting method, where the candidate regions containing symbols are filtered out before using other criteria. Recently, Dutta et al. [Dutta 2012] proposed an approach based on the utilization of product graph for spotting symbols on graphical documents.

- Chemical

  Besides symbol recognition and spotting, graph theory has also been largely applied to the characterization of chemical structures. Chemical graph theory is a branch of graph theory that is concerned with analyses of all consequences of connectivity in a chemical graph. Chemical graph serves as a convenient model that can represent different chemical objects as molecules, reactions, crystals, polymers, clusters etc. Chemical systems may be depicted by graphs using the simple conversion rule [Mishra , Shimoni 1998, Mahé 2005].

- Action recognition

  Currently, graph-based representation has also captured the interest of people working on action/activity recognition due to its simplicity and effectiveness. Since the human pose can be easily modeled by a graph, many researchers have shifted their focus from the appearance-based model to the graph-based model. Usually, the graph of a person contains a set of nodes that correspond to the parts of the human body, as well as a set of edges that are spatial relationships between the nodes [Gupta 2009, Raja 2011]. Yao and Li [Yao 2012] first applied the graph-based representation model to still action images. Each key-point is represented by view-independent 3D positions and local 2-D appearance features. Afterwards, the usage of the graph model has been expanded to video. O. Çeliktutan et al. [Çeliktutan 2013] used the graph to model not only the structure of the human pose at each single moment, but also the movements of the joints as a chain graphical structure. In this way, both spatial and temporal information are encoded into the graph model.

3.3.2 Graph-based representation model

The usage of a graph-based model for handwriting analysis is still not as fashionable as the appearance model. However, its specialty of capturing and modeling the structural properties of the objects has made it very popular in molecule compound recognition in chemistry and symbol recognition in document analysis. However, only a few attempts have been made in handwriting recognition researches with graph representation or related concepts up to now. Researchers started using graphs in the context of single character recognition, such as the graph-based recognition of Chinese characters reported in [Lu 1991, Zaslavskiy 2009]. Even though the hierarchical model proposed in [Lu 1991] reflects the hierarchical nature of handwritten characters, especially of Chinese Characters, which have a more complex structure than Latin languages, in general, a more sophisticated representation mechanism is required to distinguish similar characters. Later, Fischer et al. developed a graph representation model based on the skeleton of word image, which contains only vertexes without edges [Fischer 2010a, Fischer 2013]. By adding enough intersection points among keypoints, the structural information is still preserved and meanwhile graph representation is robust to the bad quality of the skeleton. However, it also leads to the fact that graphs may contain too many vertexes. Recently, due to the expensive computational cost of graph matching, the transformation of graphs into feature vectors has attracted attention. In [Chherawala 2011], a directed acyclic graph is constructed for each subword first, and then is converted into a topological signature vector that preserves the graph structure. Moreover, A. Dutta [Lladós 2012] adapted the graph serialization concept in graph matching for handwritten word spotting. By extracting and clustering the acyclic paths of graphs, a one-dimensional descriptor, bag-of-paths (BoP), is generated for describing the words. This attempt is interesting but the performance is far from satisfying.

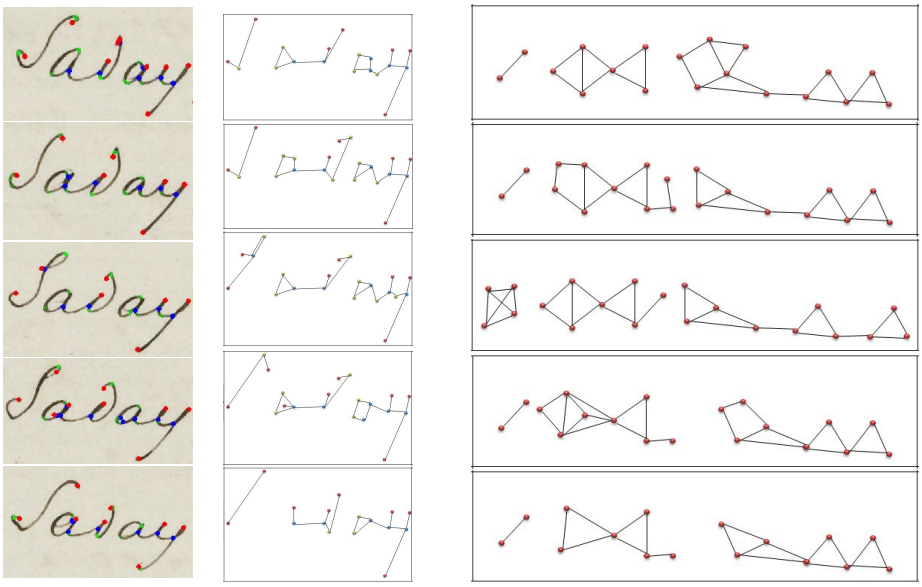
3.3.2.1 Definition of graph

Since the graph is a very classic and abstract data structure, the idea of constructing a graph according to our application can be quite diverse. We made our first attempt on two simple proposals. The elements for constructing the graphs are structural points and strokes. Two proposals are explained as follows, and the examples are given in Fig. 4.18.

1. Vertex - Structural point, Edge - Stroke
2. Vertex - Stroke, Edge - Structural point

To evaluate two proposals, we first check the consistency of the representation in terms of signatures of graph. Here, we use very basic signatures, as follows:

- Number of vertexes
- Number of edges



(a) (b) (c)

Figure 3.32: Graph construction: the images in the first column (a) are the original examples; the images in the second column (b) are the graphs made based on the first proposal; the images in the last column (c) are the graphs made based on the second proposal

- Number of 1-degree vertexes
- Number of 2-degree vertexes
- Number of 3-degree vertexes
- Number of 4 or more than 4-degree vertexes

The signatures of a graph are much more than we have listed above. To make a simple test, we first use only these six signatures. The result shows that the representation using structural points as vertexes and strokes as edges has more consistency than the second proposal. After this step, we further investigate the similarities between these two types of graph by calculating graph edit distance manually. The first proposal shows the superiority again in the cost of edit distance. The analysis of the results can be that the construction using strokes as vertexes is more sensitive to small variations such as one high-curved structural point, which happens to be one dominant characteristic of handwriting. Concerning the first idea, its construction accords more with the nature of characters, which makes more senses and is more tolerant of variations. Hence, we choose to use the first proposal as our prototype of graph construction, which is that the vertexes of a graph correspond to the structural points and the strokes connecting them are considered as the edges (see Fig. 3.33).

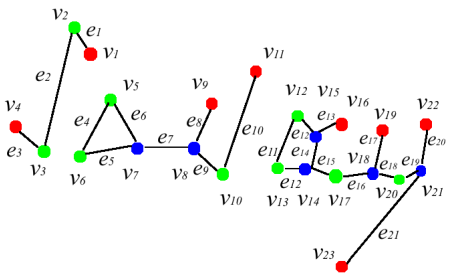


Figure 3.33: Graph representation of a word

Besides the structural signatures of the handwriting, morphological information is also a critical aspect in handwritten shape description. In order to integrate morphological properties in the graph representation model, we assign to each vertex the Shape Context descriptor as the attribute. For the edges, we use the length of strokes (the number of pixels on the skeleton) as the attribute. In this way, both morphological and topological signatures of the handwriting are encoded in the proposed graph. The formal definition of graph in our case is as

Definition 1. (Graph) A graph is a four-tuple $g = (V, E, \mu, v)$, where

- V is the finite set of vertexes, corresponding to the structural points
- $E \subseteq V \times V$ is the set of edges, corresponding to the strokes
- $\mu : V \rightarrow L$ is the vertex labeling function, corresponding to the Shape Context descriptor
- $v : E \rightarrow L'$ is the edge labeling function, corresponding to the length of stroke

3.3.3 Graph representation of handwritten document images

A handwritten word is usually composed of several connected components (CCs). Since one connected component corresponds to one graph, a word is represented as a sequence of graphs. For example, the word "*Savay*" in Fig.18 is represented by three graphs corresponding to "S", "av" and "ay". Meanwhile, it can also be considered as an entire graph for the whole word image without taking the CCs into account. Three different representation methods for word images are proposed. In the following part, we introduce the details of each proposition and the corresponding comparison method. In the end, we show the results of testing the propositions and give the discussion of the selection of the representation method.

Connected component labeling (alternatively connected-component analysis, blob extraction, region labeling, blob discovery, or region extraction) is an algorithmic application of graph theory, where subsets of connected components are uniquely labeled based on a given heuristic.

Connected component labeling is used in computer vision to detect connected regions in binary digital images, although color images and data with higher-dimensionality can also be processed. In the scenario of handwriting, a connected component can be a stroke, a part of a letter, a letter, or several letters.

The connected component labeling method used in our approach is a two-pass algorithm [Shapiro 2002], applied to the binary image. The algorithm makes two passes over the image. The first pass is to assign the temporary labels and record the equivalences. Then in the second pass, each temporary label is replaced by the smallest label of its equivalence class. This two-pass algorithm is relatively simple to implement and understand.

A brief introduction to two-pass connected-region extraction is presented below.

1. On the first pass

   Iterate through each pixel of the image by column, then by row (Raster Scanning). If the pixel is not the background, get the neighboring elements of the current pixel. If there are no neighbors, uniquely label the current element and continue. Otherwise, find the neighbor with the smallest label and assign it to the current element. Store the equivalence between neighboring labels.

2. On the second pass

   Iterate through each element of the data by column, then by row. If the element is not the background, relabel the element with the lowest equivalent label.

Figure 3.34: Example of connected component labeling

- Based on connected components

  In this method, we consider that each word image is represented as a sequence of graphs. Each graph corresponds to a connected component. The word image comparison is converted to the comparison of sequences of graphs. Four different distance measures are tested. Their definitions are as follows.

  Given A and B two images, image A contains N_a Connected Components (CCs), and image B contains N_b connected components.

  – Hausdorff Distance (HD)

$$D(A,B) = \max\left\{\max_{a\in A}\min_{b\in B} ged(a,b), \max_{b\in b}\min_{a\in A} ged(a,b)\right\} \tag{3.15}$$

– Modified Hausdorff Distance (MHD)

$$D(A,B) = \max\left\{\frac{1}{N_a}\sum_{a\in A} min_{b\in B} ged(a,b), \frac{1}{N_b}\sum_{b\in B} min_{a\in A} ged(a,b)\right\} \tag{3.16}$$

– Dynamic Time Warping (DTW)

$$D(A,B) = \frac{\sum_{a\in A, b\in B} ged(a,b)}{\max(N_a, N_b)} \tag{3.17}$$

– Mine

$$D(A,B) = \min\left\{\frac{1}{N_a}\sum_{a\in A} min_{b\in B} ged(a,b), \frac{1}{N_b}\sum_{b\in B} min_{a\in A} ged(a,b)\right\} \tag{3.18}$$

where $ged(\cdot)$ means the graph edit distance function.

Due to handwriting variation, the amount of connected components within the same word can be different with different instances. For example, the instances Q11, Q12, Q13, Q17, Q18 in Fig.3.35 respectively have 4, 3, 2, 3, 2 CCs. Hence, the graphs of different instances of the same word can be different. In this case, the comparison based on connected components does not work well.

- Based on entire graph

  In order to overcome the problem caused by the variation of CCs, instead of comparing each pair of connected component graphs, we represent a word with an entire graph. Then we calculate the graph edit distance based on entire graphs without considering CCs. It is less computational than the comparison method based on CCs, but the problem is that it does not take the order information into consideration while comparing. Some errors are caused by this omission.

- Based on the merging of connected components

  Since the previous two methods have each its own drawback, we are trying to find a method which can avoid both problems. First of all, it has been proved that the order information is crucial. It should be included into the comparison procedure. In this way, DTW (Dynamic Time Warping) becomes a good option. Thus, we use DTW to find the assignments among connected components using the graph edit distance. Secondly, we need to make the similarity measure robust to the variation of CCs. Based on the DTW matching results, a two-direction merging operation is executed. The graphs assigned to the same connected components are considered as one entire graph. In this way, both the sequences of graphs represent the query and the test image might change. Afterwards, with the new sequences of graph representation,

graph matching is performed again. The average distance among corresponding graph matchings is considered as the final distance between two word images.

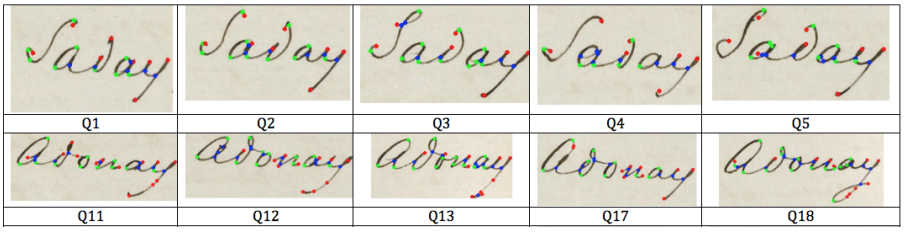


Figure 3.35: Examples used in the tests

Exploratory tests are performed on a small set of data, which contains 19 queries from 3 word classes (Table 3.1), and 237 regions of interest (equivalent to segmented words). The mean Average Precision is used as the evaluation criteria of the performance. The results are presented in Table 3.2.

Words	Amount of instances	Amounts of CCs
Savay	5	3, 3, 3, 3, 3
Avouay	10	4, 3, 2, 1, 3, 4, 3, 2, 2, 3
Otheos	4	5, 2, 1, 1

Table 3.1: Detailed information of queries used in the tests

	Savay	Avouay	Otheos	mAP
HD	15.06	30.98	19.77	21.94
MHD	44.77	52.40	34.68	43.95
DTW	51.98	52.52	27.59	44.03
Mine	40.05	56.25	40.29	45.53
Entire	29.22	59.71	82.08	57.00
Merge (single)	10.42	15.14	45.86	23.81
Merge (double)	41.27	50.54	53.70	48.50

Table 3.2: The results of tests on each proposition

It can be seen that entire graph matching achieves the best performance in terms of mAP. However, if we check the result for each word class, merge in double direction performs well in each word class in general. Since, for the word class "*Savay*", all the instances have the same amount number of CCs and all the CCs are the same, DTW method gains the best performance. Meanwhile, since the amount of CCs

varies a lot in word class "*Avouay*" and "*Otheos*", entire graph matching perform best among all the comparison methods.

In addition, the complexity of the graph edit distance approach used in our proposition is $O(n^3)$, while the complexity of the classic graph edit distance approach is $O(n!)$. The complexity of the DTW distance measure is $O(pq)$. Thus, the total word comparison of merging in double direction costs $O(pqn^3)$. n is the amount of vertexes in the graph. p, q are respectively the amount of CCs in the query and the test image.

3.3.4 Approximate graph edit distance

Graph edit distance is one of the most widely used methods to compute the difference between two graphs [Hero 2006, Bunke 2006, Raphael 1968]. The basic idea of graph edit distance is to define the difference of two graphs as the minimum amount of edit operations required to transform one graph into the other. Namely, we compute the number of edit operations needed, which is composed of insertion, deletion, and substitution of nodes and edges. Two graphs G_1 and G_2 have the edit path $h(G_1, G_2) = (e_{d1}, \cdots, e_{dk})$ (each e_{di} indicates the edit operation) to convert G_1 completely into G_2 using specific editing. Fig. 19 shows an example of the edit path between G_1 and G_2.There exist a number of different edit paths for transforming G_1 to G_2.To find out the most suitable edit path, one introduces a cost for each edit operation, measuring the strength of the corresponding operation. The idea of such a cost function is to define whether or not an edit operation represents a strong modification of the graph. The graph edit distance between graphs G_1 and G_2 is computed as

$$d(G_1, G_2) = min_{e_{d1},\cdots,e_{dk} \subseteq h(G_1,G_2)} \sum_{i=1}^{k} c(e_{di}) \tag{3.19}$$

There are several classic algorithms to solve graph edit distance. The computation

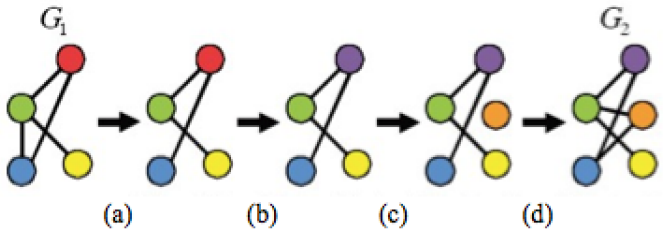


Figure 3.36: An edit path between G_1 and G_2:(a) edge deletion (b) node substitution (c) node insertion (d) edge insertion

of the edit distance is usually carried out by means of a tree search algorithm, which explores the space of all possible mappings of the nodes and edges of the first graph to the nodes and edges of the second graph. A widely used method is based on the A^* algorithm [Raphael 1968] which is a best-first search algorithm. The basic

idea is to organize the underlying search space as an ordered tree. The root node of the search tree represents the starting point of our search procedure, inner nodes of the search tree correspond to partial solutions, and leaf nodes represent complete - not necessarily optimal - solutions. Such a search tree is constructed dynamically at runtime by iteratively creating successor nodes linked by edges to the currently considered node in the search tree. In order to determine the most promising node in the current search tree, i.e. the node that will be used for further expansion of the desired mapping in the next iteration, a heuristic function is usually used. Formally, for a node p in the search tree, we use *g(p)* to denote the cost of the optimal path from the root node to the current node p, i.e. *g(p)* is set equal to the cost of the partial edit path accumulated so far, and we use *h(p)* to denote the estimated cost from p to a leaf node. The sum of *g(p)* and *h(p)* gives the total cost assigned to an open node in the search tree. One can show that, given that the estimation of the future costs *h(p)* is lower than, or equal to, the real costs, the algorithm is admissible, i.e. an optimal path from the root node to a leaf node is guaranteed to be found [Raphael 1968].

In our approach, we choose to use a suboptimal graph edit distance algorithm in order to reduce the computational complexity of the approach. This suboptimal algorithm in [Riesen 2009] is based the decomposition of graphs into sets of subgraphs. These subgraphs consist of a node and its adjacent structures. Instead of using dynamic programming, bipartite matching is adapted to find the optimal match.

The general idea of this algorithm is to use assignment algorithm finding the optimal assignments for nodes. To take the local structure of graph into consideration, the cost of node substitution is composed of two parts. First is the cost calculated from the attributes of two nodes. The second comes from the local structure comparison, as shown in Eq. 4.20. We assign different weights to the two parts according to the case.

$$C_{substitution} = w_1 C_{SC} + w_2 C_{local\_structure} \tag{3.20}$$

$$C_{deletion} = d \tag{3.21}$$

$$C_{insertion} = a \tag{3.22}$$

where a and d are constant values manually chosen. $C_{local\_structure}$ is the edit operation cost on the edges. It follows the same principle as the operation for vertexes. The aim is to find out an optimal edit path for the adjacent edges of two vertexes. The costs of deletion and insertion of edge are constant values, while the substitution cost equals $1 - e_{short}/e_{long}$, where e_{short} means the length of the shorter edge, e_{long} is the length of the longer edge.

The assignment algorithm used in this approach is Munkres' (Hungarian) algorithm [Munkres 1957]. It can solve the bipartite matching problem in polynomial time. The input of Munkres' algorithm is a cost matrix defined as follows. Let $g_1 = (V_1, E_1, \mu_1, v_1)$ be the source and $g_2 = (V_2, E_2, \mu_2, v_2)$ be the target graph with

$V_1 = (u_1, \cdots, u_n)$ and $V_2 = (v_1, \cdots, v_m)$, respectively.

$$C = \begin{bmatrix} c_{1,1} & c_{1,2} & \cdots & c_{1,m} & c_{1,\varepsilon} & \infty & \cdots & \infty \\ c_{2,1} & c_{2,2} & \cdots & c_{2,m} & \infty & c_{2,\varepsilon} & \cdots & \infty \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{n,1} & c_{n,2} & \cdots & c_{n,m} & \infty & \infty & \cdots & c_{n,\varepsilon} \\ c_{\varepsilon,1} & \infty & \cdots & \infty & 0 & 0 & \cdots & 0 \\ \infty & c_{\varepsilon,2} & \cdots & \infty & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \infty & \infty & \cdots & c_{\varepsilon,m} & 0 & 0 & \cdots & 0 \end{bmatrix} \tag{3.23}$$

where $C_{(i,j)}$ denotes the cost of a node substitution, $C_{(i,\varepsilon)}$ denotes the cost of a node deletion $C(u_i \rightarrow \varepsilon)$, and $C_{(\varepsilon,j)}$ denotes the cost of a node insertion $C(\varepsilon \rightarrow v_j)$. The Munkres' algorithm first finds the $min|V_1|, |V_2|$ vertex substitutions which minimize the total costs. Then the costs of $max\{|V_1|, |V_2|\}$ vertex deletions and $max\{|V_2|, |V_1|\}$ vertex insertions are added to the minimum cost vertex assignment such that all vertexes of both graphs are processed. Using this approach, all vertexes of the smaller graph are substituted and the vertexes remaining in the larger graph are either deleted (if they belong to g_1) or inserted (if they belong to g_2).

- Normalization

  To make sure that the cost from Shape Context comparison and the cost obtained from local structure comparison are in the same scale, we divide the total edge edit distance by the number of operations. In this way, the $C_{local\_structure}$ is also between 0 to 1.

  To normalize the final graph edit distance, the sum of edit costs is also divided by the number of operations.

- Weight selection

  To decide the weights w_1 and w_2, we have tested the performance of graph edit distance with several different sets of weights (w_1=0.2 and w_2=0.8, w_1=0.5 and w_2=0.5, w_1=0.8 and w_2=0.2). The results reveal that in our scenario, the description given by the Shape Context is more discriminant and important than the information depicted by the adjacent structure in terms of the subgraph. Hence, we assign 0.8 to w_1 and 0.2 to w_2.

- Costs for the insertion and deletion operation

  We have carried out an experiment to investigate the optimal costs for the insertion and deletion operation by exhaustively finding the optimal costs for each word in the validation set. The range of the costs are from 0 to 1 and we use 0.1 as the interval. Subsequently, for each sample, we evaluate the performances of 100 cost configurations. Fig. 3.37 gives the results of four samples.

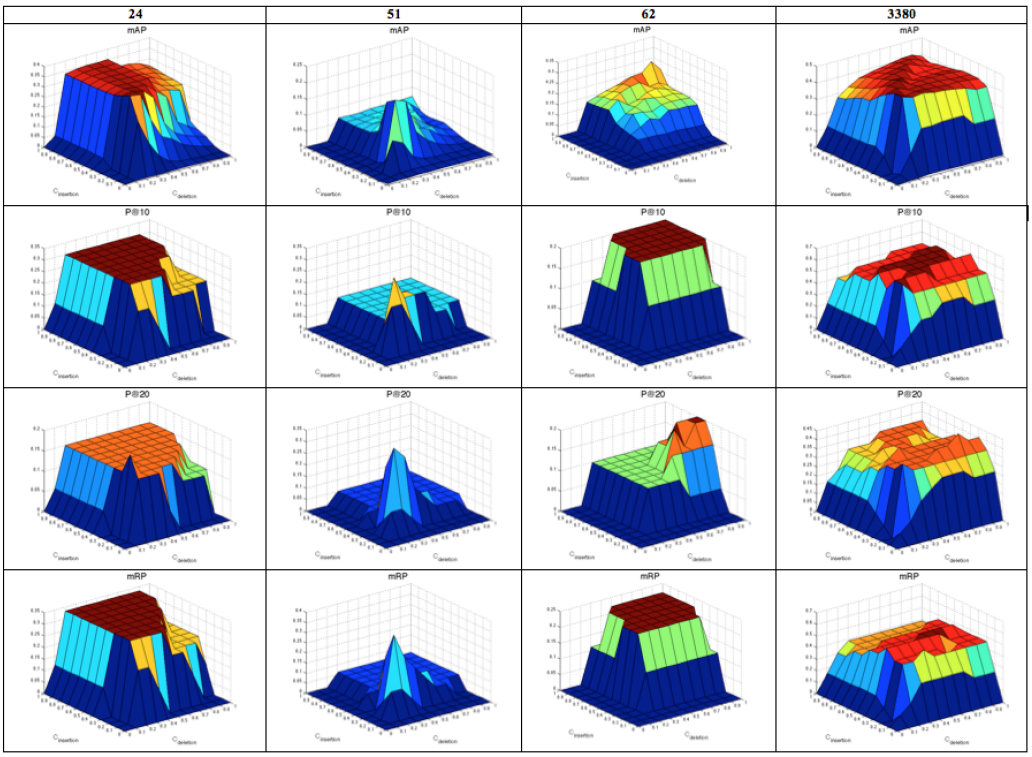


Figure 3.37: Word spotting performance evaluation for four samples in terms of mAP, P@10, P@20, mRP

The results show that the best performance for each training sample is achieved with different configurations of the costs. It indicates that there is no unique cost configuration which is suitable for all samples. Therefore, at this moment, we use 0.5 as the costs for deletion and insertion operations.

In this chapter, we have presented one of the contributions of this PhD work, the graph-based comprehensive representation model dedicated to historical handwritten document images. We have shown the details of how to construct the graph and to use the graph to describe the images as well as the selection and extraction of the features adapted in the model. Besides, some basic preprocessing techniques used in our algorithm are also introduced. Our work on the study of interest points and point descriptors for word spotting in the historical handwritten document images has been published in *the 15th International Conference on Computer Analysis of Images and Patterns, 2013*. In the following chapter, we discuss existing handwriting retrieval approaches and give the methodology of our word spotting propositions, which are based on the novel representation model we propose.

Munkres' algorithm for the assignment problem

Input: A cost matrix C with dimensionality n
Output: The minimum cost node or edge assignment
1. For each row r in C, subtract its smallest element from every element in r
2. For each column c in C, subtract its smallest element from every element in c
3. For all zeros z_i in C, mark z_i with a star if there is no starred zero in its row or column
4. **STEP 1:**
5. **for** each column containing a starred zero
6. do cover this column
7. **end for**
8. **if** n columns are covered then **GOTO** DONE
9. **else GOTO** STEP 2
10. **end if**
11. **STEP 2:**
12. **If** C contains an uncovered zero then
13. Find an arbitrary uncovered zero Z_0 and prime it
14. **if** There is no starred zero in the row of Z_0 then
15. **GOTO** STEP 3
16. **else**
17. Cover this row, and uncover the column containing the starred zero **GOTO** STEP 2
18. **end if**
19. **else**
20. Save the smallest uncovered element e_{min} **GOTO** STEP 4
21. **end if**
22. **STEP 3:** Construct a series S of alternating primed and starred zeros as follows:
23. Insert Z_0 into S
24. **while** In the column of Z_0 exists a starred zero Z_1 **do**
25. Insert Z_1 into S
26. Replace Z_0 with the primed zero in the row of Z_1. Insert Z_0 into S
27. **end while**
28. Unstar each starred zero in S and replace all primes with stars. Erase all other primes and uncover every line in C. **GOTO** STEP 1
29. **STEP 4:** Add e_{min} to every element in covered rows and subtract it from every element in uncovered columns. **GOTO** STEP 2.
30. **DONE:** Assignment pairs are indicated by the positions of starred zeros in the cost matrix

Chapter 4

Word Spotting Approach Based on a Comprehensive Model

Contents

4.1 State of the art 79
4.1.1 Appearance-based and structure-based approaches 80
4.1.2 Learning and learning-free approaches 85
4.1.3 Segmentation-based and segmentation-free approaches 88
4.2 Our contribution 93
4.2.1 Introduction 93
4.2.2 Coarse selection 96
4.2.3 Fine selection 108
4.3 Evaluation 127
4.4 Other potential applications 131
4.4.1 Symbol classification 131
4.5 Conclusion 132

4.1 State of the art

Handwritten word spotting is an alternative solution when the OCR technique fails in handwritten document images. It offers users a fast access to the untranscribed content of manuscript collections. The objective of word spotting is to find all instances of a given word in a potential dataset of document images. As introduced in chapter 1, there are two types of word spotting tasks. One is referred to query by example (QBE), when the query word is provided as an image of a handwritten word, while the other one is named query by string (QBS), in which case the query word may be a text string. The initiation of handwritten word spotting is to solve the QBE retrieval problem under the assumption that there is no transcription of the dataset. Many state-of-the-art works have focused on word spotting in the QBE scenario [Rath 2007, Leydier 2009, Rodríguez-Serrano 2009, Rusiñol 2011, D. Fernández 2011, Almazán 2012b, Lladós 2012, Rothacker 2013, Toselli 2014] while recently some efforts have been made for the QBS case [Almazán 2013, Fischer 2010a, Aldavert 2013]. Usually, the QBE approaches first

create character or lexicon prototypes manually or by some other schemes. Afterwards, when the user types the query, such character templates are put together to synthetically generate an example of the query word.

Before presenting our proposed QBE approaches, we first give a brief state-of-the-art on word spotting works in terms of some key aspects. Usually, a QBE word spotting approach includes at least two main parts: a representation model that describes the word images with the essential information, and a similarity measure that provides the comparison between the query image and target images. Sometimes, a word spotting mechanism can also include a learning step to train the system to recognize either word prototypes, individual characters or groups of characters (grams). First of all, the overview of existing works is made with respect to the representation model, and then we give a discussion of the usage of learning. In the end, considering that the feasibility of the approach is also important, the previous works are summarized under two categories, depending on whether the segmentation of words needs to be performed or not.

4.1.1 Appearance-based and structure-based approaches

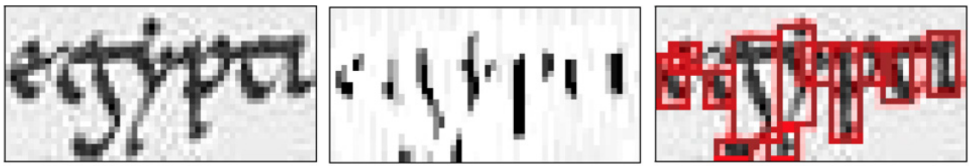
Different representations can be found in the literature for the description of word images. Concerning the representation models used in handwriting retrieval and recognition, the existing approaches can be classified into two categories. One is dedicated to the appearance-based model, which represents the image as an n-dimensional feature vector. The other uses the structure-based model, which usually uses a set of geometric and topological primitives and relationships among them to represent the image. The current word spotting approaches that belong to the first category are predominant.

4.1.1.1 Appearance-based model

Appearance-based representation model is the most common model used in word spotting application, whether for printed text or handwriting. Many features including both global and local features, have been used in this approach category. For example, the global features such as the width, height, aspect ration, number of pixels, describe the images as a whole, whereas local features are independently relevant to different regions of the images or primitives extracted from it, for instance position/number of loops, key points or regions.

In the beginning of the handwritten retrieval researches, Rath and Manmatha [Rath 2007] proposed the well-known word spotting approach based on columnwise sequential feature alignment. After normalizing the inter-word variations such as skew and slant, three features are extracted and normalized. They are respectively projection profile, word profile and background/ink transitions. Then, they perform dynamic time warping matching to calculate the distance between two word images. The principle of their approach is simple, however, it requires the regular and tidy writing as the condition of a good performance. Afterwards, the usage of

local gradients became popular in word spotting application. Both Y. Leydier et al. [Leydier 2007, Leydier 2009] and J. A. Rodríguez et al. [Rodríguez-Serrano 2009] chose the gradients extracted from the patch as the feature to describe the local region of the handwriting. The difference of these two approaches is that Leydier represents word images as a set of zones of interest (ZOIs) and compares the images by using a cohesive matching between ZOIs, guides and gradient fields, while Rodríguez just concatenates the gradient histograms computed in each cell. In the work of [Leydier 2007], a selection of robust features is carried out. Many invariant features used in image registration and indexation are tested on both the first and the second Gaussian filtering images. In the end, they find that the orientation of the gradient is the most efficient feature for the word shape description, which can be explained by the fact that the orientation of the gradient describes the true local structure of the strokes and the orientation of the characters' contours. Moreover, depending on the language, they use the graphemes along a specific direction (vertical or oblique) as the guides and zones of interest as enlarged bounding boxes of guides for matching (see Fig. 4.1). The main advantage of this approach is its elastic and cohesive matching which helps to overcome the spatial variations of handwriting, while the shortcoming of this approach is that it is not suited to retrieve short words (less than four letters), because the information extracted by this approach for short words is not enough to distinguish the word shapes. This shows that its representation model could be improved by either adding more information or extracting features that are more distinctive.



(a) (b) (c)

Figure 4.1: Guides and zones of interest of the word "egýpti". (a) template (b) guides (c) ZOIs [Leydier 2007]

Compared to Leydier's approach, the unconstrained word spotting solution presented in [Rodríguez 2007] adopts a simpler but denser way to reformulate the gradients of all pixels in the word image. A sliding window centered at each column of word image is used to obtain a feature vector. In addition, at each position, the sliding window is subdivided into rectangular cells with three different strategies. In the end, the feature vectors of all the sliding windows within the same word image are concatenated together as the final representation. According to the experiment results shown in [Rodríguez 2007], we can notice that this approach achieves a better performance than other conventional approaches. However, it is at the expense of a much higher computational cost caused by 128-dimensional feature vectors. Later, M. Rusiñol and his group [Rusiñol 2011] introduced the bag-of-visual-word scheme

into the word spotting task. They first extract SIFT descriptors from the document images, and then use the classified SIFTs to represent patches. In order to refine the representation, they also apply the latent semantic indexing technique, which allows one to retrieve patches even if they do not contain the same exact features as the query sub-image. The evaluation of [Rusiñol 2011] shows very promising results. One drawback, however, is that the window size cannot be trivially adapted to the length of the query. As noted by the authors, the performance of the method is very dependent on the length of the query with respect to the fixed size of the window. Besides, the problem of expensive computation still exists in this algorithm, since it densely calculates the SIFT descriptors all over the page (see Fig. 4.2).

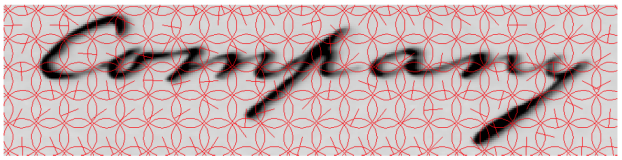


Figure 4.2: Dense SIFT features extracted from a word image [Rusiñol 2011]

Recently, more and more advanced techniques invented on natural images have been brought into document image analysis. By using the exemplar SVM, J. Almazán et al. [Almazán 2012b] realize segmentation-free writer-dependent word spotting based on an unsupervised learning of the query described by the quantized HOGs. This work is remarkable due to two points. First of all, the use of HOG descriptor addresses the problem of fixed window size existing in [Rusiñol 2011]. Secondly, it improves the word spotting method by adapting the unsupervised learning to the particular query with less constraints on the training samples. Both facts make this approach feasible and flexible. The issue unsolved with the approach in [Almazán 2012b] is indexing the query in a multi-writer scenario. However, shortly afterwards, they developed an attribute-based approach to address multi-writer word spotting problem [Almazán 2013]. Moreover, this work bridges the gap between query-by-example and query-by-string word spotting solutions. In this approach, word strings are encoded as a pyramidal histogram of characters, which is called PHOC, inspired by the bag of character string kernels in machine learning. Fig. 4.3 gives an illustration of the PHOC construction of the word "*beyond*". Concerning the representation of word images, Fisher Vector framework based on SIFT features is adopted instead of BoVW framework. In addition, to capture the structure of the word image, a spatial pyramid of 2 x 6 is implemented, leading to a final descriptor of approximately 25,000 dimensions. The most innovative part of this work is the proposed attribute-based representation model, which realizes a unified representation of word images and strings.

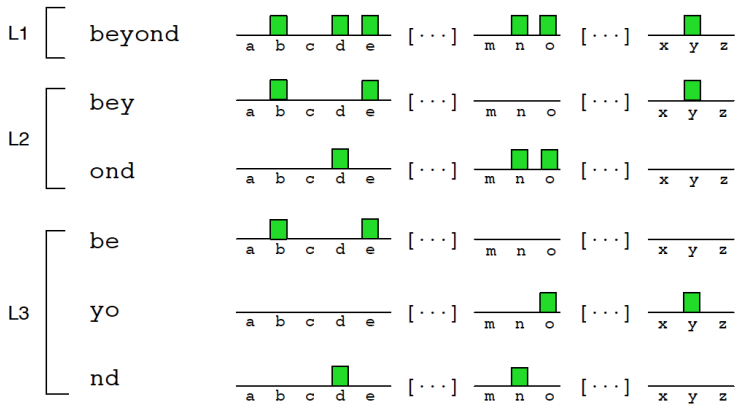


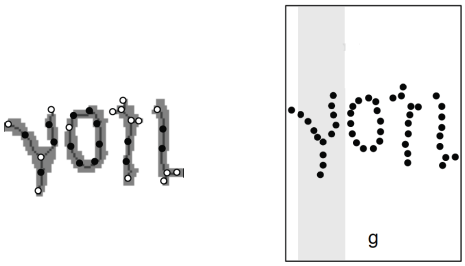
Figure 4.3: PHOC histogram at levels 1, 2, and 3. The final PHOC histogram is the concatenation of these partial histograms [Almazán 2013]

4.1.1.2 Structure-based model

The use of the structure-based model for handwriting analysis is still not as fashionable as the appearance-based model. Some existing works use graph representations because of its speciality of capturing and modeling the structural properties of the objects. Even though the graph representation is very popular in molecule compound recognition in chemistry and symbol recognition in document analysis, up to now, there are only a few attempts made in handwriting retrieval and recognition researches with graphs or related concepts.

In the literature, there are some early works using graph representations for isolated digits [Filatov 1995, López 1998], or in the context of single character recognition, such as the graph-based recognition of Chinese characters reported in [Lu 1991, Hsieh 1995, Suganthan 1998, Zaslavskiy 2009]. Even though the hierarchical model proposed in [Lu 1991] reflects the hierarchical nature of handwritten characters, especially of Chinese characters, which have a more complex structure than Latin letters, in general, a more sophisticated representation mechanism is required to distinguish similar characters. Later, Fischer et al. developed a graph representation model for handwriting recognition based on the skeleton of word image, which contains only vertexes without edges [Fischer 2013] as the example shown in Fig. 4.4. By adding enough intersection points among keypoints, the structural information is still preserved and meanwhile graph representation is robust to the bad quality of skeleton. However, it also leads to the fact that graphs may contain too many vertexes, leading to the increase of the computation.

Recently, due to the expensive computational complexity of graph matching, the transformation of graphs into feature vectors has attracted attention. In [Chherawala 2011], first the topological and geometrical information of subword shapes is extracted from the skeleton. Then, a directed acyclic graph is constructed for each subword, and is converted into a topological signature vector that preserves



(a) (b)

Figure 4.4: Graph representation in the approach [Fischer 2013] (a) original word and keypoints detected on the skeleton (b) the corresponding graph

the graph structure. The work of D. Fernández et al. [D. Fernández 2011] is also a structure-based word spotting approach, which is inspired by characteristic Loci features. As Fig. 4.5 shows, for each key point, the numbers of intersections along 8 directions are calculated and quantized. The extraction of the modified Loci feature is carried out on the skeleton image and the keypoints can be either the background pixels or the foreground pixels. Since the Loci feature are encoded in base 3, the dimension of the feature space is 3^8 (=6,561). If we take the decimal 4-dimensional Loci feature as Locu numbers, then the representation of a word image is a histogram of Locu numbers. The benefit of using a hash table to store the compact signatures of images is that the word can be easily indexed, which saves both space and time.

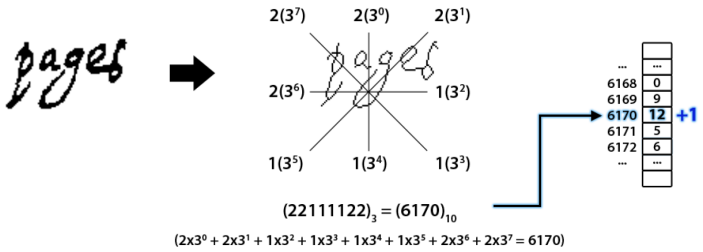


Figure 4.5: Characteristic Loci feature of a single point of the word page [D. Fernández 2011]

Moreover, Lladós and et al. [Lladós 2012] adapt the graph serialization concept to graph matching for handwritten word spotting. By extracting and clustering the acyclic paths of graphs, one-dimensional descriptor, bag-of-paths (BoP), is generated to describe the words. This attempt is interesting but the performance is far from satisfying.

J. Lladós et al. in [Lladós 2012] has made a comparison and analysis of two

structure-based handwriting representation models and two appearance-based models in the context of word spotting application in historical documents. They are respectively the works of [D. Fernández 2011], [Dutta 2012], [Rath 2007] and [Rusiñol 2011]. The result of [Lladós 2012] is that the appearance-based approach [Rusiñol 2011] achieves the best performance, while structural methods do not perform as well as the appearance-based methods, especially the bag-of-paths based approach. The reason is that it needs a preliminary step of transforming the raster image into a vectorial image in order to build a graph representing the words. This vectorization process is very easily influenced by slight degradations when we are dealing with historical documents. However, if we consider the recent progress on the structure-based word spotting approaches and add those works [Fischer 2010b, Fischer 2013] into the competition, structure-based methods and appearance-based methods are neck and neck. They all have their respective strengths and weaknesses. Generally, the appearance-based methods occupy a lot of memory and time complexity is high, which makes them deficient when space and time are limited. On the other hand, most structural methods face the problem caused by binarization, which is applied as a preliminary process. Depending on the condition of the document images, the performance can be severely degraded due to the loss of information after binarization. In this sense, the appearance-based approaches that do not require binarization and extract the information directly from grey-level images are less constrained.

Finally, some existing works on word spotting application cannot be simply classified as appearance-based nor structure-based approaches, because they adopt both statistical and structural descriptors. Kessentini et al. [Kessentini 2010] propose a multi-stream HMM-based approach for off-line handwritten recognition. They combine two different types of features: directional density features and density features. The authors conclude that the combination of these types of features obtain the best results. We have also oriented our researches to the approaches based on the combination of the structure and the appearance which seem to be promising. The appearance-based shape context descriptor and the structure-based topological node feature are used separately in the early stage to describe word images, and then are integrated in the similarity measure to compare the samples. Using both morphological and topological aspects is the choice that we made for yielding a comprehensive and complete description for word images. We will present the details in the next section of this chapter.

4.1.2 Learning and learning-free approaches

From another point of view, the approaches that address the handwritten word spotting problem can generally be divided into two classes by the employment of learning. The learning-based approaches use machine learning techniques in order to train models of the characters or the sought words [Fischer 2010a, Frinken 2010, Rusiñol 2011, Almazán 2012b, Almazán 2013, Rothacker 2013, Aldavert 2013, Toselli 2014], while the counterpart is called example-based

(learning-free) [Rath 2007, Lladós 2007, Leydier 2009, Rodríguez-Serrano 2009, D. Fernández 2011, Lladós 2012], which is retrieval-oriented with dedicated matching scheme based on image sample comparisons without any necessary associated training process.

In the work of [Rusiñol 2011], latent semantic indexing (LSI) technique [Deerwester 1990] is used to retrieve relevant patches even if they do not contain the exact same features. Originally, LSI is designed for the text retrieval task. Its principle is to use a mathematical technique called "singular value decomposition (SVD)" to identify patterns in the relationship between the terms and concepts contained in an unstructured collection of texts. In the scenario of the work of M. Rusiñol et al. [Rusiñol 2011], it assumes that patches that appear in the same contexts tend to have similar meanings. In other words, there exists some underlying semantic structure in the descriptor space. Hence, instead of comparing patch features directly after obtaining them with the BoVW model and SPM (spatial pyramid matching) strategy, they first transform all patch features into a space where the patches with similar topics but with different descriptors lie close and calculate the cosine distance as the similarity measure in this space. In this way, words are modeled in relation to the topics of each single document. A similar idea is also used for addressing query-by-string word spotting problem [Aldavert 2013]. A word snippet has two representations: textural representation and visual representation. Latent semantic analysis (LSA) is used to transform these two representations into a common space. Several abstract topics are designed and each topic is a representative distribution of textural and visual features. To obtain the transformation matrix, the textural descriptor and visual descriptor are concatenated together as a single descriptor. With all the words in the training set, a descriptor-by-word matrix is built and then decomposed by SVD to generate the transformation matrix. The final similarity distance is calculated in the space where transformation matrix projects. As a conclusion, the employment of LSA allows the query-by-string word spotting of Latin languages. The experimental results shown in [Aldavert 2013] illustrate its strength against other state-of-the-art approaches. However, we can also find that the performance of this approach decreases a lot when it copes with a query that does not appear in the training set.

Hidden Markov Models (HMM) is a classic and effective technique often used for handwriting recognition. Many works on handwritten word spotting are derived from it [Fischer 2010a, Kessentini 2010, Rothacker 2013]. For example, in [Fischer 2010a], three different HMMs are designed (see Fig. 4.6) for retrieving keywords in a multi-writer handwriting scenario. Each text line is represented as a sequence of letter models. Based on the likelihood ratio between a keyword model and a filler model, each text line gets a score. With the certain threshold, the text line is decided as a positive match or negative one. Another instance of using HMM for handwritten word spotting is the work of [Toselli 2014]. In this work, after extracting sequential morphological features for text lines, HMM model is used to estimate the likelihood of characters and collaborates with N-gram language model to predict words. Word graph (WG) is adapted to represent word sequence hypothe-

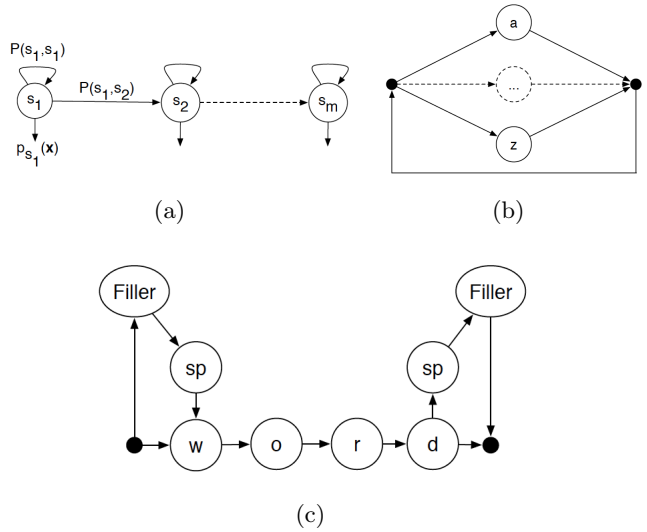


Figure 4.6: Hidden Markov Models designed in [Fischer 2010a]: (a) letter HMM (b) filler HMM (c) keyword HMM

ses as well as alternative word segmentations and word decoding likelihoods (see Fig. 4.7). Compared to other learning approaches, HMM needs a large amount of training samples to estimate the HMM models, especially in the scenario of multi-writing styles.

Neural network is a popular learning technique in recent decades in the domain of machine learning. Due to its strength, it has been adopted to a broad range of computer vision applications, and handwritten word spotting is no exception. The work of [Frinken 2010] has employed bidirectional long short-term memory (BLSTM) neural network as the recognizer, which is a recently developed recurrent neural network. It produces a sequence of probabilities for each letter and each position in the text line. An illustration of this network can be seen in Fig. 4.8.

One common limit to the learning techniques introduced above is their deficiency in new queries that are not included in the training dataset. To solve this problem, J. Almazán et al. [Almazán 2012b, Almazán 2014] adopt the exemplar learning technique for query word images which can highlight the regions that are relevant to the query in the images. By means of examplar SVM framework [Malisiewicz 2011], a new representation of each query is built. The new representation can be understood as weighting the dimensions of the region descriptors that are relevant to the query.

In a word, for both learning and learning-free approaches, there are advantages and disadvantages. Although there are extensive example-based approaches existing in the literature, their performance in the multi-writer scenario is disappointing. Moreover, from the application point of view, such methods are not practical enough, if the user just wants to type the word in the keyboard and search

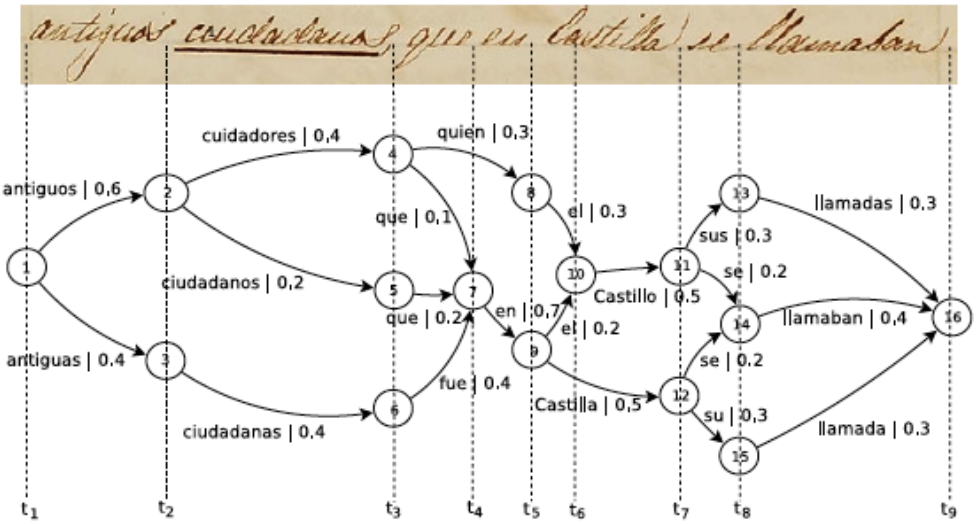


Figure 4.7: A simplified example of a normalized WG that would be obtained from the decoding of a line image [Toselli 2014]

the existence of this word. Nevertheless, compared to learning-based approaches, example-based approaches do not require the training process, which can be performed even with a very small quantity of images and without ground truth. Since most of learning-based approaches rely on the extraction or estimation of characters [Fischer 2010a, Aldavert 2013, Almazán 2013, Toselli 2014], such character extraction is manually done [Fischer 2010a, Toselli 2014] or by means of some clustering scheme [Almazán 2013]. In this way, if some specific characters or N-grams only appear a few times throughout the collection, the performance of such methods is influenced due to the lack of training samples. Besides, without the annotation of the words, the learning-based approach cannot be conducted. Additionally, the framework of this kind of approach is difficult to apply to glyph languages in an unconstrained situation, since there are countless characters existing in such languages. For example, it needs a great effort to create the prototypes of Chinese characters, since there are around 50,000 characters in total.

4.1.3 Segmentation-based and segmentation-free approaches

As research on word spotting develops, two dominant categories of word spotting algorithms have been created. One is the segmentation-based method, as the first attempts to solve the word spotting problem relying on an initial layout analysis step devoted to performing word segmentation. Since many unsolved issues remain in segmenting a line into individual words, there are as yet no satisfactory word-segmentation methods. Therefore, segmentation-free word spotting methods are introduced. The segmentation-free method attempts to perform spotting and seg-

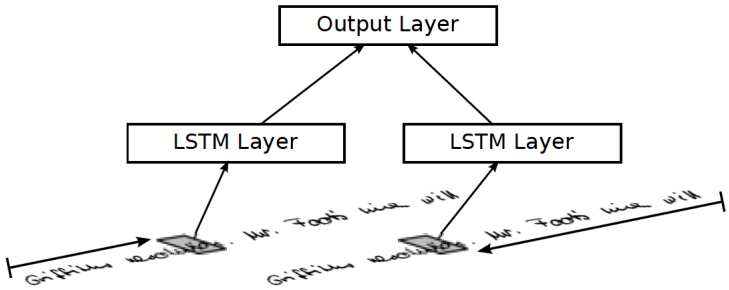


Figure 4.8: The entire BLSTM Neural Networks [Frinken 2010]

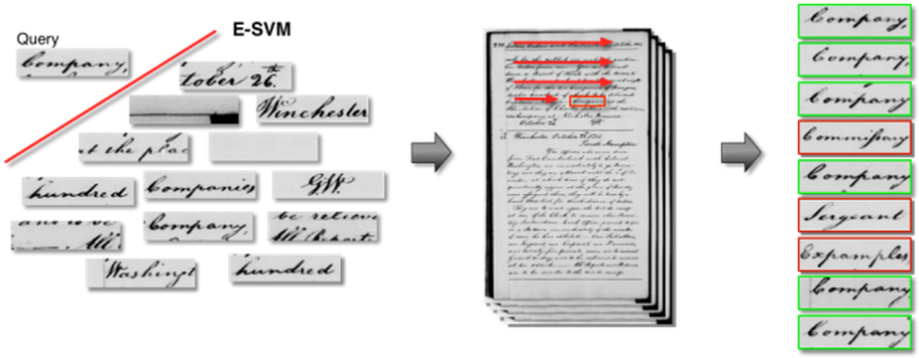


Figure 4.9: Illustration of exemplar SVM training and sliding window search [Almazán 2014]

mentation concurrently. Rather than a candidate word image, the whole page image acts as input.

In word spotting, algorithms often incorporate a method to segment lines into words. After segmentation, these words are represented with shape signatures to be treated as one-dimensional signals. Early work in segmentation-based word spotting was done by Rath and Manmatha in [Rath 2007]. By using dynamic time warping (DTW) based distance, the query word image is finally matched against the whole word corpus. One of the drawbacks of this approach, however, is that it is heavily dependent on the precise segmentation of words. Its tolerance to segmentation error restricts its performance. The approach of D. Fernández et al. [D. Fernández 2011] based on characteristic Loci Features stored in a hash structure is also highly dependent on an accurate word segmentation.

In Fig. 4.10, a classic process of segmentation-based word spotting is illustrated. Document is first segmented and the distances between the word images are calculated. After clustering the word images, some clusters are manually labeled and can

be used as index terms.

Figure 4.10: An illustration of the segmentation-based word spotting idea [Rath 2007]

With respect to the performance, segmentation methods play an important role in the segmentation-based word spotting approaches. However, there is no existing segmentation method that can achieve 100% accuracy for all kinds of handwriting. Because handwriting skews and slants can be random and irregular, depending on the characteristics of the writing style, the performances of the segmentation method can be very different. If the target images are not well segmented, it directly results in a reduction of the spotting rate of the algorithm. We have also studied different segmentation methods on words or lines. S. N. Srihari's group [Srihari 2005] realized automatic word segmentation, which is based on taking several features on either side of a potential segmentation point and using a neural network for deciding whether or not to segment between two distinct words. In V. Malleron's PhD work [Malleron 2009], a new approach for line segmentation is proposed. After extracting connected components, they compute for each component a neighborhood-fan which contains Euclidean distance and orientation of the component and nearest neighbors in 18 directions of space in order to know the component neighborhood structure. With the analysis from the corner and border extraction and the line orientation estimation, adjacent connected components are linked together to construct the text line.

Nowadays, the most recent trends in word spotting research are focusing on trying to propose methods that do not need a perfect word segmentation step, or in some cases, no segmentation at all. The recent works of [Fischer 2010a, Frinken 2010] propose methods that relax the segmentation problem by requiring segmentation only at the text line level. In [Fischer 2010a], a handwritten word spotting method that just needs to segment the text into lines is presented. Given a text line, it is first normalized. The trained Hidden Markov Model (HMM) is able

to perform the segmentation of the word within a line and the word spotting in a single step. In a similar fashion, Frinken et al. present a method in [Frinken 2010] that uses a Neural Network (NN) to perform the spotting at the line level. In both cases, the advantage of the presented models is that they can perform the spotting independently of the different writing styles, while their drawback is that they still rely on good line segmentation and line normalization steps in order to be able to process lines as one-dimensional signals. [Leydier 2007] presents a word spotting methodology based on local gradients. In this work, neither line nor word segmentation is required. For a given query image, zones of interest are detected and the orientations of gradients are encoded by a simple descriptor. The word spotting is performed by trying to locate zones of the document images with similar zones of interests. These retrieved zones are then filtered and only the ones sharing the same spatial configuration with the query model are returned. As Fig. 4.11 shows, a cohesive distance is defined to locate local zones of words corresponding to the query.

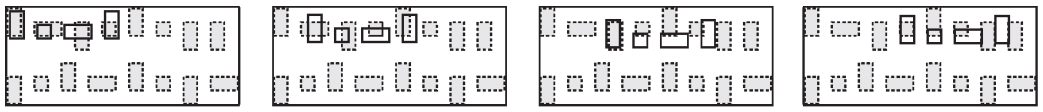


Figure 4.11: An example of template matching. The framed transparent boxes are the template's ZOIs, the grey dotted-framed boxes represent the image's guides. Only the first template ZOI has to be propped up on an image guide [Leydier 2007]

The methods developed by M. Rusiñol and his team in [Rusiñol 2011, Aldavert 2013, Rothacker 2013] are also segmentation-free. They can be applied to both handwritten and typewritten historical document images, meanwhile they also work on non-Latin scripts. The patch-based framework where patches are represented by the well-known bag-of-visual-words model powered by SIFT descriptors renders this approach feasible.

In general, the segmentation-free approaches can be better adapted to different contexts and reduce the complexity of preprocessing. However, up to now, almost all segmentation-free approaches are purely appearance-based. Since handwriting is inherently structured, it would be interesting to develop a segmentation-free approach based on the mixture of the appearance and the structure.

At the end of this section, we would like to draw a conclusion on the state of the art related to handwritten word spotting approaches. No matter appearance-based approaches or structure-based approaches, each of the approaches presented above has its strength and weakness. To some extent, they are complementary. Subsequently, we propose several multi-model approaches based on the combination of the appearance and the structure to address word spotting problem in different scenarios. Details are given in the next section.

Approach	App.-based	Struct.-based	Seg.-based	Seg.-free	Learning	Learning-free
[Rath 2007]	columnwise features		word-level			X
[Lladós 2007]	Shape Context		word-level			X
[Leydier 2009]	Gradients			X		X
[Rodríguez-Serrano 2009]	Gradients		word-level			X
[Fischer 2010a]	Sequential features		line-level		HMMs	
[Fischer 2010b]		Graph-based	word-level		HMMs	
[Frinken 2010]	Sequential features		line-level		BLSTM NN	
[Rusiñol 2011]	BoVW (SIFT)			X	LSI	
[D. Fernández 2011]		Pseudo-structure (Loci)	word-level			X
[Almazán 2012b]	HOG			X	Exemplar SVM	
[Almazán 2013]	Fischer vector (SIFT)		word-level		CCA	
[Aldavert 2013]	BoVW (SIFT)		word-level		LSA	
[Rothacker 2013]	Bag-of-features (SIFT)			X	HMMs	
[Toselli 2014]		Word graph	line-level		HMMs	

Table 4.1: Summary of attributes of each handwritten word spotting approach

4.2 Our contribution

After an exhaustive study of conventional approaches for characterizing handwritten document images, we realize the fact that the description of handwriting needs to be comprehensive, i.e. the sole usage of homogeneous features is not enough to distinguish different handwritten patterns. Inspired by this, we decide to build a representation model for handwriting with a complete description, including multi-facets of handwriting, such as topological and morphological aspects. Subsequently, based on the proposed representation model, a set of word spotting approaches are developed regarding different scenarios. Since the objective of the work is to focus on the single or similar writing style, we do not involve any learning process to address multi-writing style issue. However, as a perspective, we plan to adaptat current approaches with learning techniques to tackle manuscripts written by different hands.

4.2.1 Introduction

In our approach, given a dataset of handwritten document images, the pages are first segmented into lines. Afterwards, we start to build the graph-based representation model introduced in chapter 3 for the images. First of all, we extract the skeleton of the handwriting and detect three types of structural points based on it. Taking the structural points as the vertexes and the strokes as the edges, the handwriting is represented as graphs that comprise the 2-dimensional spatial relations between handwritten strokes. In order to encode the contextual information, which is considered powerful in handwriting description, into the graph-based representation, the Shape Context descriptor is used as the attributes of vertexes. In this way, the representation model preserves both topological and morphological signatures of the handwriting.

To make the approach more efficient, the retrieval process is executed in two steps. They are respectively called the coarse selection and the fine selection. The coarse selection is designed to realize a fast comparison between the query and the test images. A sliding window is adapted to scan the text line connected component by connected component. By using the explicit graph embedding approach [Gibert 2012], 2-dimensional handwriting is converted into a 1-dimensional scalar feature vector in terms of single vertexes and bipartite relations between vertexes. Based on the comparison of graph embedding vectors, regions of interest are selected (see section 4.2.2.2). Most of the area on the test page is ignored in the next comparison. The objective of this step is to eliminate the regions that are not similar to the query according to the distance calculated based on the graph embedding representation so as to reduce further comparisons. Consequently, the recall must be as high as possible after the coarse selection. With regions of interest, the attribute of vertexes in the graph is recalculated with respect to the entire contour inside the region of interest. A more precise and structure-based comparison between the query and the regions of interest is then performed on the graph edit distance (see

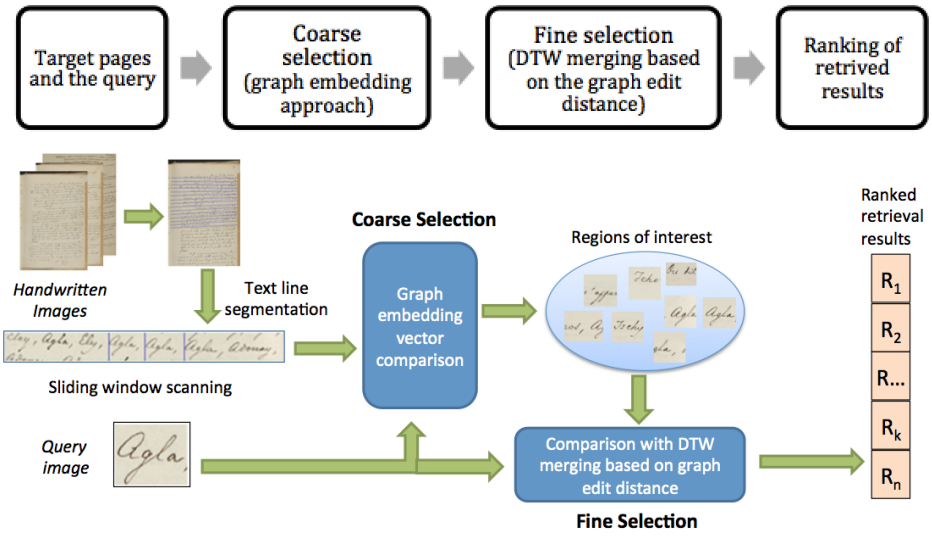


Figure 4.12: General workflow of the proposed coarse-to-fine word spotting approach

section 4.2.3.3). The similarity distance obtained at this step is used as the final score to rank the candidates. Fig. 4.12 shows a general workflow of the proposed approach.

Before we present the details of the methodology of our approaches, we would like to introduce the datasets and criteria used for the performance evaluations in next sections.

There are mainly four datasets used in the experiments and three of them are public ones. They are respectively the George Washington dataset [Rath 2007], the IAM off-line dataset [Marti 2002] and the BH2M (the Barcelona Historical Handwritten Marriages) dataset [Fernández 2014]. The fourth dataset, which is related to the CITER project, is offered by Prof. Antony McKenna.

- George Washington dataset

  It consists of 20 pages from a collection of letters by George Washington. It was accurately segmented into words and transcribed. There are 4860 segmented words with 1124 different transcriptions. We have used it for our segmentation-based word spotting approach evaluation in the fine selection section.

- IAM off-line dataset

  The IAM off-line dataset forms of unconstrained handwritten text, which were scanned at a resolution of 300dpi and saved as PNG images with 256 grey levels. We select part of the IAM off-line dataset, which is the biggest collection of a unique writing style for the segmentation-based word spotting approach evaluation. It contains 59 pages of modern handwritten English with 3760 segmented words of 1399 transcriptions.

- BH2M dataset

  BH2M (the Barcelona Historical Handwritten Marriages) dataset is composed of Marriage Licenses from 1451 to 1905 conserved at the Archives of the Cathedral of Barcelona. The resolution of images is around 20MB per page. It is used for both segmentation-based and segmentation-free word spotting approaches evaluation. For the segmentation-based part, the evaluation corpus contains 27 pages from a collection of BH2M dataset which was written by the same writer. There are 6544 segmented words with 1751 transcriptions. Concerning segmentation-free word spotting proposition evaluation, the ground-truth used in the experiments contains 50 pages from one volume written by the same writer.

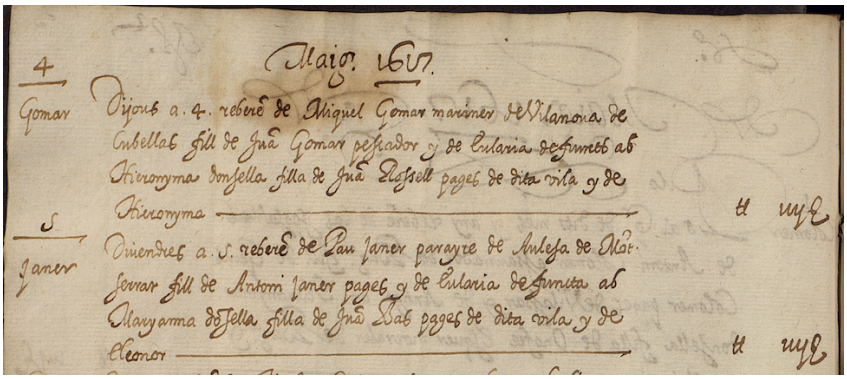


Figure 4.13: Sample of the Barcelona Cathedral dataset [Fernández 2014]

- Dataset of the CITER project

  It is composed of letters written by different French philosophers. The corpus that we select constitutes four collections written by four different writers. There are 11 pages containing approximately 2000 words. It is used for both segmentation-based and segmentation-free proposition validation.

In order to evaluate the performance, we choose three performance assessments based on precision and recall measures. Given a query, let us label *rel* the set of relevant objects with regard to the query and as *ret* the set of retrieved elements from the dataset.

- *P@n* — the precision at n is obtained by computing the precision at a given cut-off rank, considering only the n topmost results returned by the system. In the evaluation, we provide the results at the 10 and 20 ranks.

- *R-Precision* — the *R-Precision* is the precision computed at the Rth position in the ranking of results, R being the number of relevant words for that specific query.

- *mAP* (mean Average Precision) — *mAP* is computed using each precision value after truncating at each relevant item in the ranked list. For a given query, let *r(n)* be a binary function on the relevance of the nth item in the returned ranked list, the mean average precision is defined as follows:

$$mAP = \frac{\sum_{n=1}^{|ret|} P@n \times r(n)}{|rel|} \tag{4.1}$$

4.2.2 Coarse selection

In order to reduce the computational expense of the entire approach, the coarse selection is introduced to locate some specific regions on the page where there is more probability of matching the query. We call such regions the regions of interest, which are kept as the potential candidates for further comparison. In this way, it is required that the scheme of the coarse selection be not too complex and that the recall of the selection be ensured. To avoid the drawbacks of word segmentation as explained in section 4.1.3, we apply a sliding window with a dynamic width scanning each line.

4.2.2.1 Profile-based coarse selection

The coarse selection scheme for selecting regions of interest is based on the low-level features of text lines. The sliding window moving over the text lines is shifted with certain interval (number of pixels) which is empirically set according to the handwriting resolution.

Image representation As a first attempt, we choose three low-level features to describe text images. They are respectively projection profile, upper border and lower border [Rath 2007], and orientation distribution of skeleton pixels.

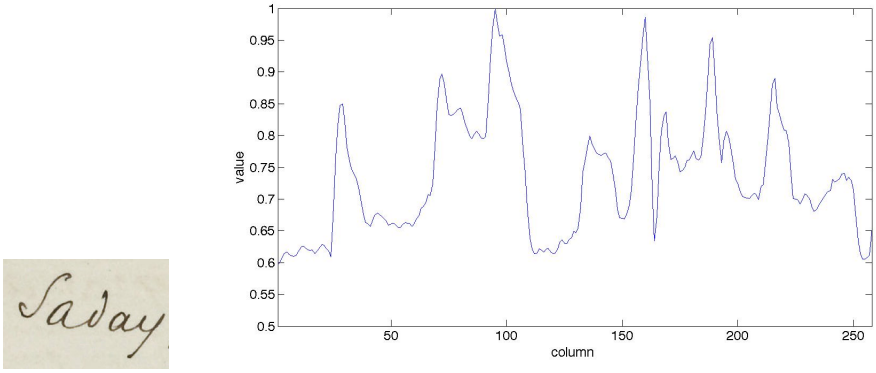
- Projection profile

  Projection profile captures the distribution of the ink along the one of the two dimensions in the handwriting. A vertical projection profile is computed by summing the intensity values in each image column separately:

$$pp(I, c) = \sum_{r=1}^{h} (255 - I(r, c)) \tag{4.2}$$

  Due to the variations in quality (e.g. contrast, faded ink) of the images, different projection profiles do not generally vary in the same range. To make them comparable, the projection profile is normalized to the range [0, 1]. Fig. 4.14 shows an example of the projection profile of a word and the original image that it is extracted from.

  Projection profile, as a holistic feature, compared to other features, is easy to understand and implement. It indicates the ink distribution over the entire



(a) (b)

Figure 4.14: (a) Original image and (b) normalized projection profile feature

word. One limitation of using the projection profile is its incapacity to handle variant scales due to the mechanism. Since the projection profile feature is explicitly calculated column by column (or row by row), which is rigid in its length, when the query image and the candidate image are not in the same scale, it is difficult to match them by using the projection profile. Hence, before comparing the projection profiles, a normalization needs to be realized. In addition, the projection profile is highly dependent on the slants. As a consequence, local pixels in the histogram may not represent the same morphological variations. Therefore, projection profile is not efficient enough to be used alone. More features should be introduced to make the description of handwriting more robust.

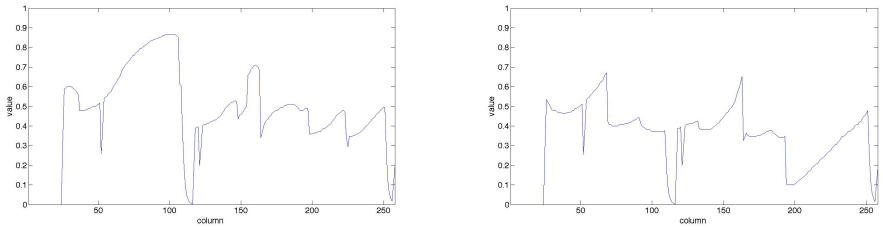
- Upper border and lower border

  Word profiles, including the upper border and lower border, capture part of the outlining shape of a word. Their extraction is conducted on binary images. After binarizing the image [Otsu 1979], if the pixel value is 1, it is a background pixel. Otherwise, it is an ink pixel. Let *is_ink (I, r, c)* be a function that returns 1 if the pixel *I (r, c)* is an ink pixel, and 0 if the pixel is a background pixel. Using *is_ink*, the upper and lower word profiles can be calculated as follows:

$$up(I,c) = \begin{cases} undefined & if\ \forall r \quad is\_ink(I,c,r) = 0 \\ argmin_{r=1,\cdots,h}(is\_ink(I,c,r) = 1) & otherwise \end{cases} \tag{4.3}$$

$$lp(I,c) = \begin{cases} undefined & if\ \forall r \quad is\_ink(I,c,r) = 0 \\ argmax_{r=1,\cdots,h}(is\_ink(I,c,r) = 1) & otherwise \end{cases} \tag{4.4}$$

  If a column does not contain ink pixels, *up* and *lp* of this column will be undefined (no distance to the nearest ink pixel from the top or bottom of



(a) (b)

Figure 4.15: (a) Normalized upper border profile (b) normalized lower border profile

word image bounding box). A number of factors, such as pressure on the writing instrument and fading ink, affect the occurrence of such gaps, which is not consistent for multiple instances of the same word. Therefore, gaps where *up* and *lp* are undefined should be closed by linearly interpolating between the nearest defined values:

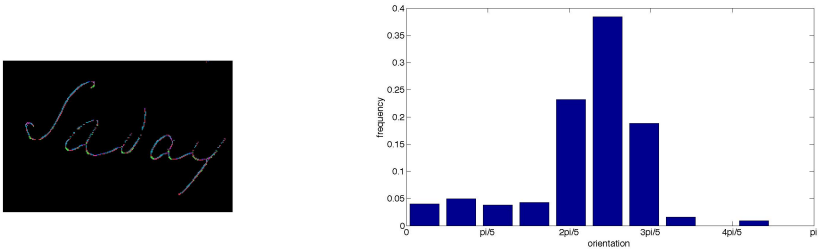
$$up'(I,c) = \begin{cases} interplated \quad value & if\, up(I,c) \quad is \quad undefined \\ up(I,c) & otherwise \end{cases} \tag{4.5}$$

The same for *lp* function. Fig. 4.15 shows the upper border profile and the lower border profile from the original image Fig. 4.14(a).

- Orientation distribution

  Orientation is one of the main visual characteristics involved in the pre-attentive view. The work of [Journet 2008] has proved the usefulness of orientations for the characterization of fonts in old document images. By extracting the features linked to frequencies and to orientations in the different areas of a page, it is possible to extract and compare elements of high semantic level without expressing any hypothesis about the physical or logical structure of the analyzed documents. We are interested in this characterization thus proposing the combination of texture features and the orientation information. As skeleton is considered as a brief representation of the handwritten shapes, we calculate the orientation of each pixel on the skeleton and afterwards summarize the orientation distribution of the entire shape as a frequency histogram. To eliminate scale variance, the histogram is normalized into the same range [0, 1] by dividing by the maximum. Fig. 4.16 shows the distribution histogram of the skeleton of the image in Fig. 4.14 (a).

Similarity measure Since the proposed representation contains both holistic descriptors i.e. projection profile, word border profiles and a visually salient descriptor orientation distribution, so the similarity measure must take into account both of these dimensions. They are considered together as the final distance of the coarse selection in the format of a linear combination.

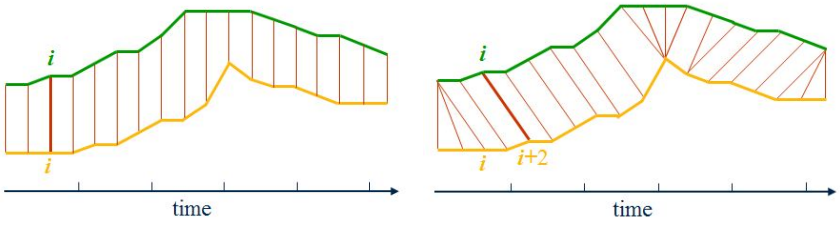


(a) (b)

Figure 4.16: (a) Orientation distribution of each skeleton pixel (b) normalized orientation histogram of skeleton pixels (10 bins)

After extracting the features for the content inside the sliding window, we employ dynamic time warping (DTW) [Rath 2007] as the comparison method to calculate the local optimal distance for the projection profile and word profiles, and the Chi-square distance for the orientation distribution of skeleton pixels.

Dynamic time warping is originally designed to compute the distance between two time series. A time series is a list of samples taken from a signal, ordered by the time when samples are obtained. A naive approach to calculate a matching distance between two time series could be to resample one of them and then compare the series sample-by-sample (see Fig. 4.17(a)). The drawback of this method is that it does not produce intuitive results, as it compares samples that might not correspond well.



(a) alignment after resampling (b) alignment with DTW

Figure 4.17: Different alignments of two similar time series

Dynamic time warping solves this discrepancy and calculates matching distance by recovering optimal alignments between samples in the two time series. The alignment is optimal in the sense that it minimizes a cumulative distance measure consisting of "local" distances between aligned samples. Fig. 4.17(b) shows such an alignment. The procedure is called "time warping", because it warps the time axes of the two time series in such a way that corresponding samples appear at the

same location on a common time axis. The DTW distance between two time series $x_1, \cdots, x_M$ and $y_1, \cdots, y_N$ is D(M,N), which we calculate in a dynamic programming approach using

$$D(i,j) = min \left\{ \begin{matrix} D(i,j-1) \\ D(i-1,j) \\ D(i-1,j-1) \end{matrix} \right\} + d(x_i, y_j) \tag{4.6}$$

The particular choice of recurrence equation and "local" distance function *d(·,·)* is the Euclidian distance in our application. By using the given three values *D(i, j-1)*, *D(i-1, j)* and *D(i-1, j-1)* in the calculation of *D(i, j)*, DTW preserves local continuity, which ensures smooth time warping. While the slant and skew angles at which a person writes are usually constant for single words, the inter-character and intra-character space is subject to large variations. DTW offers a more flexible means to compensate for these variations than linear scaling.

For the orientation feature, since it is represented as the frequency histogram, it is natural to use Chi-square to calculate the distance. Let D_{od} denote the cost of matching the orientation features of two images, p and q.

$$D_{od} = \frac{1}{2} \sum_{k=1}^{K} \frac{(h_p(k) - h_q(k))^2}{h_p(k) + h_q(k)} \tag{4.7}$$

where $h_p(k)$ and $h_q(k)$ denote the K-bin normalized histogram of p and q, respectively.

The final similarity measure for the coarse selection is composed of the distances from all the features. Let $D_{pp}, D_{ub}, D_{lb}, D_{od}$ be respectively the distances calculated from the projection profile, the upper border profile, the lower border profile and the orientation distribution. After normalizing all the distances into the same range, we sum them up as the final distance of the coarse selection D_{coarse}.

$$D_{coarse} = D_{pp} + D_{ub} + D_{lb} + D_{od} \tag{4.8}$$

By setting a threshold for the summed-up distance shown as Eq. 4.8, the regions of interest that have more similarity to the query are selected for further comparison. In this way, the approach is more efficient compared to methods which directly carry out the sophisticated scheme on the whole page.

Evaluation We evaluate the performance of this coarse selection proposition in terms of mean average recall (mAR). Precision at this step is not considered because this step is the selection of regions of interest and precision is intended to be very low. The overall mean average recall of the evaluation is 79.04%. In conclusion, projection profile, word profile and orientation distribution, such holistic features are easy to understand and extract, but as a result of their simplicity, such columnwise features are not robust enough to accommodate handwriting variations. Moreover, the 2-dimensional spatial information of handwriting is lost when it is represented

by 1-dimensional linear sequential vectors. Even though the usage of dynamic time warping can render the matching more flexible compared to other techniques, the scale of context considered in the dynamic time warping is just one-pixel difference, which is not enough. In order to improve the performance of the coarse selection, a less rigid representation model is needed and more contextual information should be taken into account.

4.2.2.2 Coarse selection using graph embedding

After the first attempt, we continue making more efforts to improve the performance of the coarse selection. Inspired by the graph-based representation introduced in chapter 3, we propose a new coarse selection based on the same model with a fast and effective comparison scheme. Compared to the previous low-level-description-based coarse selection proposition, the graph-based approach preserves more structural signatures of handwriting, which is supposed to have more strength in distinguishing the handwritten patterns.

Before introducing the details of the approach, it is necessary to briefly recall the basic definition of our graph representation which has already been presented in Chapter 3. The structural points are used as vertexes, and the strokes are the edges (see Fig. 4.18). We can assign the attributes for both vertexes and edges. However, in the coarse selection, we give labels only to vertexes. The formal definition of graph used in the coarse selection is as

Definition 1. (Graph) A graph is a three-tuple $g = (V, E, \mu)$, where

- V is the finite set of vertexes, corresponding to the structural points
- $E \subseteq V \times V$ is the set of edges, corresponding to the strokes
- $\mu : V \rightarrow L$ is the vertex labeling function, corresponding to the local Shape Context descriptor

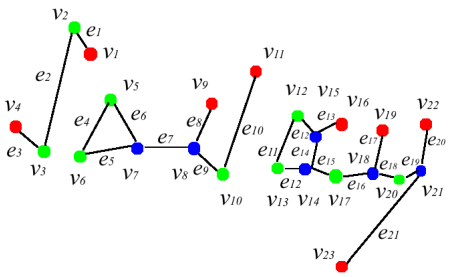


Figure 4.18: Graph representation of a word

To be specific, we use the fuzzy C-means clustering function based on the local Shape Context descriptor as the labeling function of the vertex. Graphs are converted into 1-D vectors with a specific graph embedding approach introduced in

[Gibert 2012]. Instead of moving pixelwise, the sliding window scans lines in terms of connected components, which generates much less sliding windows than the previous version. According to the width of the query, the size of the sliding window is adjusted automatically to contain a suitable amount of the connected components depending on their size. In order to clarify the rules of adjusting the sliding window, we elaborately explain them with the example given in Fig. 4.19: "*fill*" is the query word with the width w. The sliding window has the same size as the query. It starts to scan the text line from position 1. Since it moves connected component by connect component, afterwards, next positions of the sliding window are respectively 2, 3 and so on. When the sliding window comes to the position n_1, it turns out to be the exceeding width of the second connect component w_1 is less than $1/3$ of w (the width of the query). In this way, the second connected component ("*ju*") is also included in the sliding window. The same principle for the position n_2, w_2 is over $1/3$ of w, so the connected component "*pesc*" is not included. Analogously, if the total width of all the connected components inside the sliding window is less than $1/3$ of w, this position is not considered. This approximation considers connected components as a relevant lexical unit, especially in the contemporary handwriting, where it is rare to have merged connected components between words.

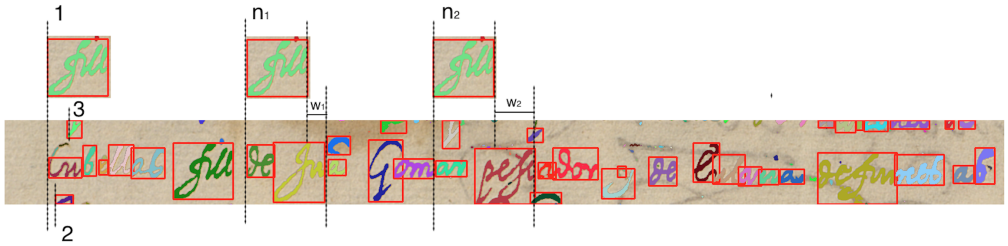


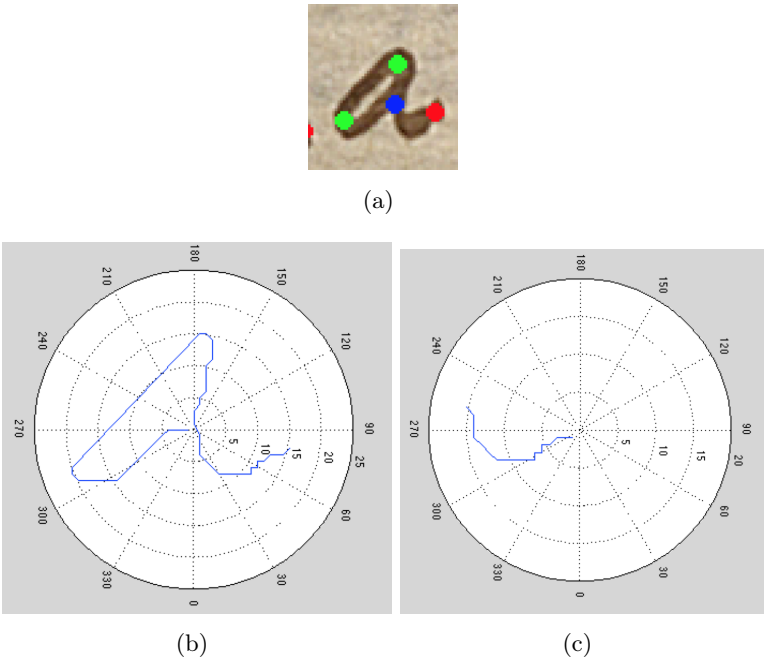
Figure 4.19: Illustration of the movement of the sliding window

The graph embedding feature is recalculated within the sliding window at each potential location. The distance between the sliding window and the query based on the graph embedding feature is used to grade the candidates. Only half the candidates are preserved as regions of interest.

Local Shape Context extraction on the k-Neighborhood of a vertex The Shape Context descriptor is a point descriptor which captures the distribution over relative positions of the shape points and thus summarizes the global shape in a rich, local polar diagram [Belongie 2002]. In order to adapt the Shape Context descriptor to our application, we especially design a dynamic selection of the relative shape points to build the local Shape Context descriptor based on the 1-step neighborhood of the reference point on the graph.

Originally, according to the graph theory [Dahm 2013], a k-Neighborhood (kN) of a vertex v is an induced subgraph formed from all the vertexes that can be

reached within k steps from v. This induced subgraph is centred around the vertex v and contains all vertexes up to k steps away. In our case, we keep using the same definition of the k-Neighborhood but replace the edges by the corresponding skeleton of the handwritten strokes. It means that the Shape Context extracted in the k-Neighborhood of a reference point records the relative positions of the skeleton points located in the corresponding area. In practice, we use k=*1*. Fig. 4.20 shows the *1*-Neighborhood of the blue point and the red point in the example "a". The polar diagram indicates the contribution of all angular sectors within the *1*-Neighborhood of the reference vertex.



(a) (b) (c)

Figure 4.20: (a) An example of the letter "a" with the structural points (b) *1*-Neighborhood of the blue point (c) *1*-Neighborhood of the red point

Vertex labeling with fuzzy C-means clustering Instead of assigning the exact local Shape Context feature as the attribute of vertexes, we decide to build a codebook of the local Shape Context features with the size of n classes and use the representative of the corresponding class as the attribute of the vertexes. The total amount of classes is a tunable parameter influenced by the nature of the dataset. It affects the efficiency of the classification. Moreover, in noisy situations, it can happen quite frequently that a vertex label is between two representatives, and there is no clear rule telling us to which representative the vertex should be assigned. In such case, a soft rather than a hard assignment can be beneficial. Hence, we choose to employ the fuzzy C-means clustering algorithm [Bezdek 1981], which can assign to every vertex a certain degree of belongingness to every cluster, rather than an

exclusive assignment. Let $P \subset R^d$ be the set of all vertex labels in all the graphs of G. Furthermore, let $W = \{w_1, \cdots, w_n\}$ be a set of n representatives of all vectors in P. Its main idea is to assign to a point $x \in P$ a degree of belongingness to each cluster in W, which is inversely proportional to the distance between x and the cluster center. This leads to

$$p_i(x) = \alpha \cdot (\frac{1}{\|x - w_i\|_2})^s \tag{4.9}$$

where α is a constant assuring that $\sum_{i=1}^{n} p_i(x) = 1$ and s is a parameter that controls the amount of fuzziness that the user is giving to the assignment. The larger s is, the more weight is given to points close to the centers.

With the fuzzy C-means assignment algorithm, we define the vertex labeling function

$$v \mapsto \lambda_s(v) = (p_1(v), \cdots, p_n(v)) \tag{4.10}$$

where $p_i(v) = P(v \in w_i)$ is the probability of the vertex v being assigned to the representative w_i, while $p_i(v) \geq 0$. Fig. 4.21 presents the labeling results of a set of corresponding points. As shown, the same branch (blue) points in three instances of "*Bara*" are studied. The *1*-neighborhoods of the reference points are plotted and the classification results with fuzzy C-means are given as the attribute of the vertex. It can be seen that the belongingness to each cluster of the three sample points is very similar.

Graph embedding representation For graph embedding, we adapt J. Gibert's [Gibert 2012] approach based on the vertex attribute statistics. It redefines the appearance of a specific label by checking the probabilities of belongingness for all vertexes in the graph. This is given a graph $g = (V, E, \mu)$ and a representative set W, the frequency of a representative $w_i \in W$ is

$$U_i = \#(w_i, g) = \sum_{v \subseteq V} p_i(v) \tag{4.11}$$

Similarly, when there is an edge between two vertexes in the graph $(u, v) \in E$, the fact that vertexes are assigned to representatives according to Eq. 4.10 makes a contribution to all relations between any two representatives. It is defined as

$$B_{ij} = \#(w_i \longleftrightarrow w_j, g) = \sum_{(u,v) \subseteq E} p_i(u)p_j(v) + p_j(u)p_i(v) \tag{4.12}$$

The formal definition of graph embedding used in our approach is given as follows.
Definition 2. (Graph Embedding) Given a set of vertex representatives $W = w_1, \cdots, w_n$, we define the embedding of a graph g into a vector space as the vector

$$\varphi_w(g) = (U_1, \cdots, U_n, B_{11}, \cdots, B_{ij}, \cdots, B_{nn}) \tag{4.13}$$

where $1 \leqslant i \leqslant j \leqslant n$, $U_i = \#(w_{i,g})$ and $B_{ij} = \#(w_i \longleftrightarrow w_j, g)$.

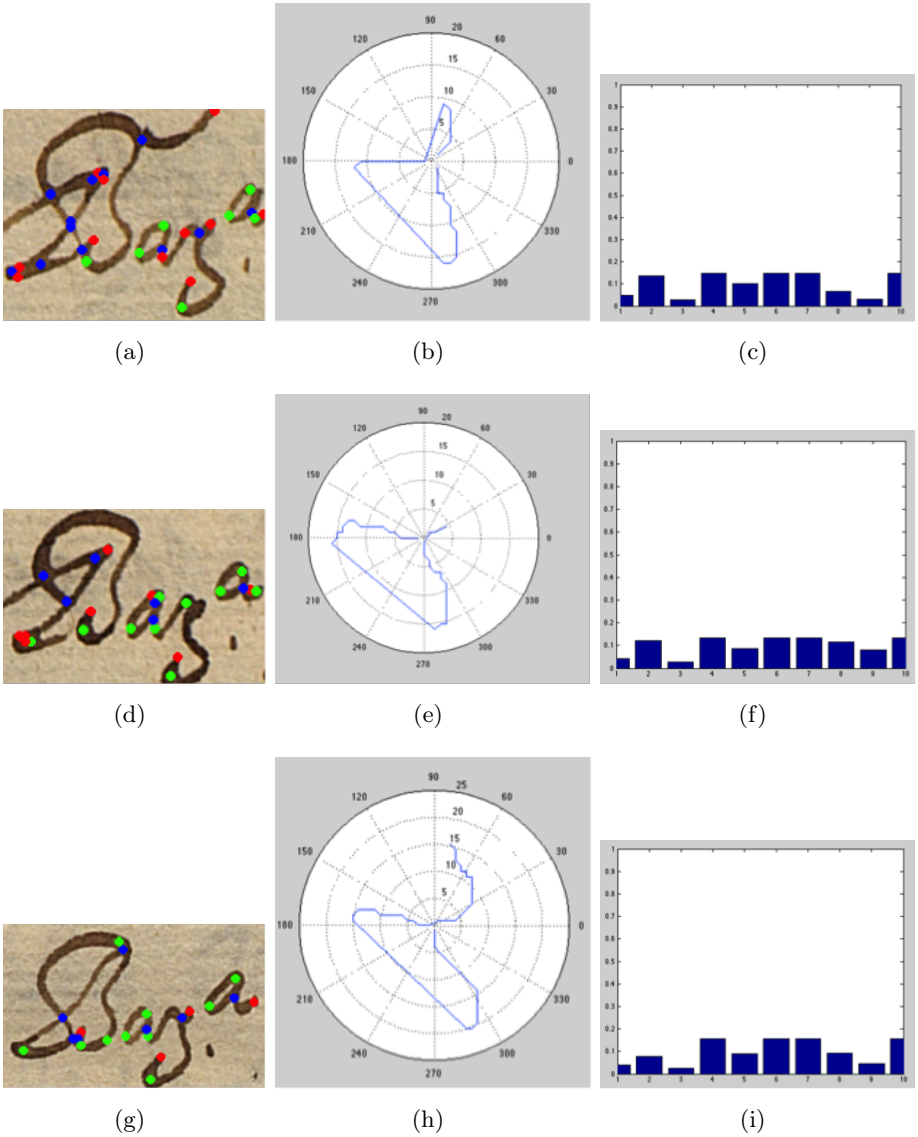


Figure 4.21: Labeling of the blue point in "a" with the fuzzy C-means. (a)(d)(g) original images (three instances of "*Bar.a*") (b)(e)(h) *1*-neighborhood of the blue points (c)(f)(i) labels assigned with the fuzzy C-means (n=10) clustering

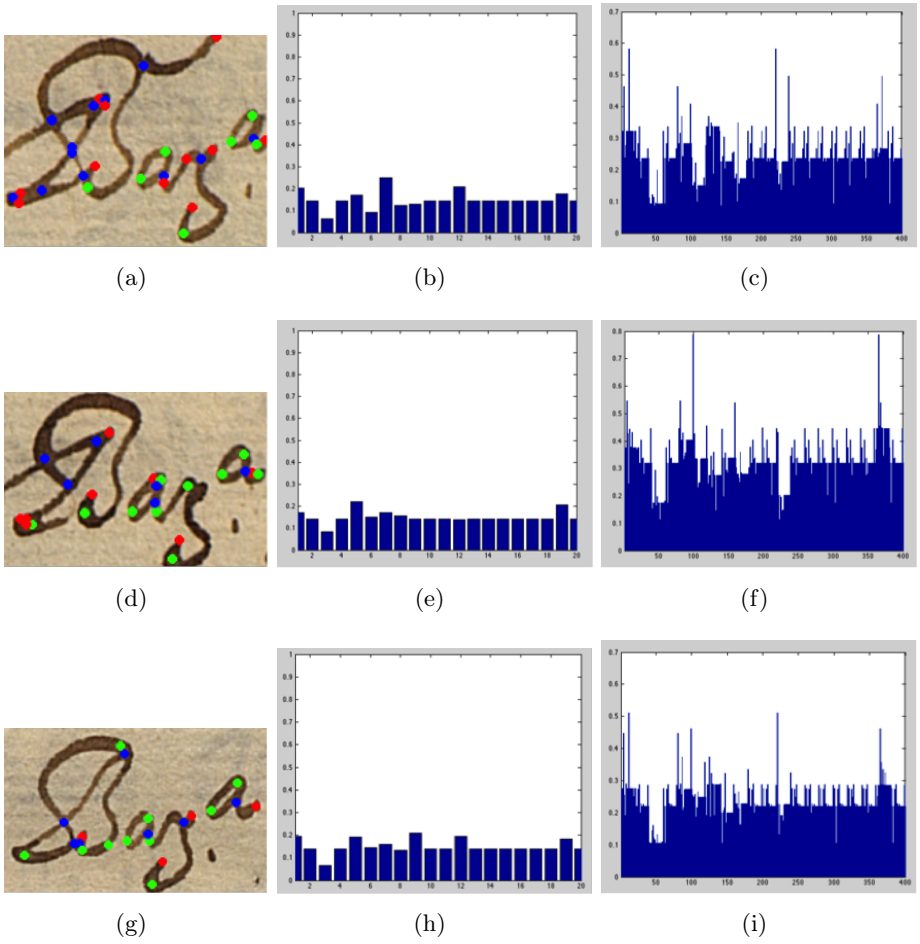


Figure 4.22: Graph embedding representation (a)(d)(g) original images (b)(e)(h) the frequency of 20 representatives ($U_i, i \in [1, 20]$) (c)(f)(i) the relation between two representatives ($B_{ij}, i, j \in [1, 20]$)

One example of graph embedding representation is shown in Fig. 4.22. The figure presents respectively the statistics of vertexes and their bipartite relations.

Since a word image usually contains several connected components, in order to adapt the graph embedding approach mentioned above to represent the entire word image, we sum up all the graph embedding vectors of the connected graphs inside the image. Given there are m connected components in the image, each of them has a graph embedding feature f_i. Let f_{GE} denote the graph embedding feature of the entire image.

$$f_{GE}(k) = \sum_{i=1}^{m} f_i(k) \quad k \in [1, n + n * n] \tag{4.14}$$

where n is the number of representatives.

We choose to use the Chi-square distance to measure the distance between two graph embedding features.

Evaluation In this part, the proposed approach is applied to a large collection of historical documents of the BH2M (the Barcelona Historical Handwritten Marriages) database [Fernández 2014]. The groundtruth used in the experiments contains 50 pages from one volume written by the same writer. The results are compared to the performance of another coarse-to-fine word spotting approach proposed by J. Almazán et al. [Almazán 2012a]. The reference approach first uses a pseudo-structure based representation embedded in a hashing structure to coarsely eliminate word images that are unlikely to be instances of the query. Afterwards, they use a Support Vector Machine to learn a representative model for each query word class based on the HOG appearance description. Differently from the proposed approach, the reference system segments the lines into words using a projection function. Moreover, the reference system needs more than one sample of the query word class and a lot of the negative samples to train the representative model of each word class. We show the results of the evaluation and the discussion in the latter part of this section.

As for the number of classes used for vertex label clustering, we empirically choose $n = 10$ in our experiment which gives the best performance on the selected dataset. The parameter s in Eq. 4.9 is set to 2.

Table 4.2 gives the average recall for the coarse selection corresponding to each query word class. In practice, there are around 800 sliding windows per page on average that are compared to the query with the graph embedding approach. Only half of the regions (around 400) are preserved as the regions of interest for the fine selection. Compared to the reference system, which selects approximately 40,000 regions in a single page after the coarse selection, the proposed approach is more efficient and feasible for large-scale collections, especially when the computational time of the fine selection linearly increases with the number of regions of interest.

Queries	Coarse selection (%)
barna	77.51
fill	77.54
filla	82.18
habitant	82.81
pages	85.06
viuda	67.21
viudo	82.47
reberé	65.30
mAR	77.51

Table 4.2: Average recall of the coarse selection

4.2.3 Fine selection

In the fine selection, a more accurate and precise comparison between the query and regions of interest is performed. Three propositions for the fine selection have been successively made. In the beginning, we tested a linearly combined representation based on complementary features. Afterwards, we proposed an automatically weighted distance computed on both appearance-based and structure-based representation. Finally, we develop a more compact graph-based representation including the morphological property as the attribute, and adapt a sub-optimal graph edit distance with merging operations to measure the similarity between word images.

4.2.3.1 A diversified selection using low-level features and dominant primitives

The idea of the first attempt for fine selection is inspired by the fact that neither the morphological properties nor the topological properties alone can give a precise and sufficient description of the handwriting for word spotting application, as has been revealed in the literature. In consideration of this, we establish a novel representation model comprising both morphological and topological properties of handwriting. In each part, different features are adopted to describe the characteristics of the writing.

Image representation With the regions of interest obtained from the coarse selection and the query, we first extract the skeleton and the contour of the text, and then detect structural points. Taking the structure points as reference points, the SC descriptions for the query and regions of interest are built. Afterwards, loop detection is applied based on the construction of graph. Ascenders and descenders are identified based on baseline detection. Moreover, all the holistic features used in the coarse selection, such as the projection profile, are also taken into account in the final similarity measure. An illustration of the representation model is given in Fig. 4.23. Detailed description of the method is illustrated as follows.

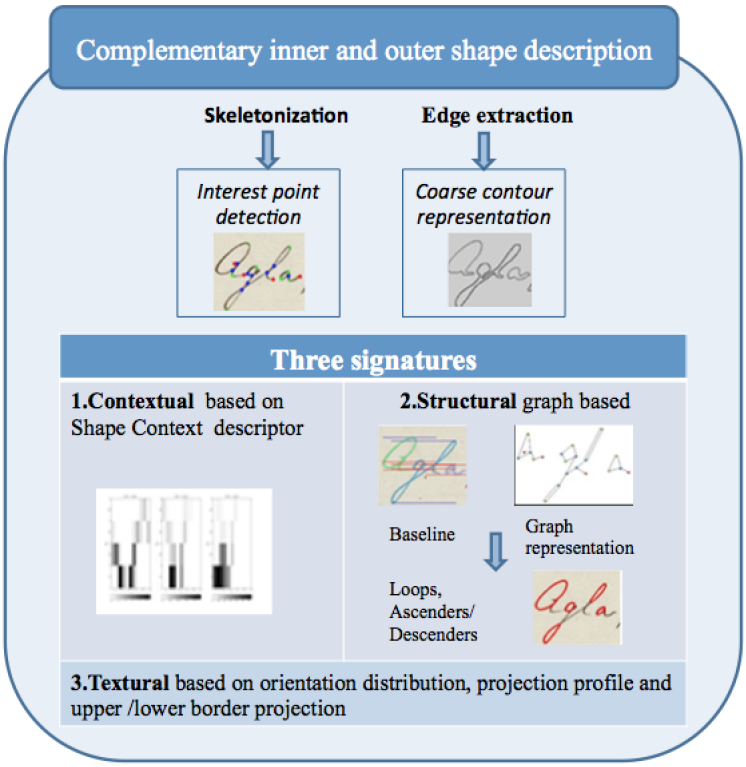


Figure 4.23: Illustration of the comprehensive image representation

Shape Context description As introduced and explained in chapters 2 and 3, Shape Context is a powerful local descriptor: even the sole use of Shape Context description can achieve a good performance in object recognition tasks [Belongie 2002]. J. Lladós et al. [Lladós 2007] have adapted Shape Context for keyword spotting in archive documents. The promising result achieved by them proves the robustness of Shape Context to noises and distortions on document images. However, since their approach is designed to spot keywords in a relatively simple scenario (archive documents), where most texts are printed, the representation proposed in [Lladós 2007] is not able to cope with complex handwritten materials. In the context of handwritten word spotting, with the consideration of the specific nature of manuscripts and the complexity of the approach, we choose to build the Shape Context descriptors on the structural points of the scripts instead of a set of random points down-sampled from the contour. The advantages of this is to obtain less redundancy in the description and meanwhile to target the regions of interests more directly. The entire contour detected inside the sliding window or the query is considered while the Shape Context descriptors are built. In this way, the image is interpreted as a set of summaries of the overall shape of the word from different points of view.

Ascender/descender detection Ascenders are strokes above the upper baseline (e.g. lower part of '*p*'). In the same manner, we refer to strokes below the lower baseline as descenders. The use of ascenders/descenders can first be found in the works on document image categorization [Myers 1995, Spitz 1997]. The main idea of these works is to convert the characters into a shape code by using different features. Ascender/descender is one of the most popular features used in such applications. Recently, many significant works dedicated to word spotting and handwriting recognition have also used strokes, especially ascenders and descenders, as part of their description of the handwriting. The work of [Lavrenko 2004a] offers an approach for historical handwriting recognition, considering severely degraded situations. It adapts both scalar and profile-based features. The number of ascenders and the number of descenders are two out of six scalar features used in this approach. Not only for the handwritten text, ascender/descender is also interesting for the printed document. S. Bai and et al. [Bai 2009] develop a fast keyword spotting approach for printed document images without performing OCR. They extract 7 different features (including ascenders/descenders) and convert the word into a Word Shape Code (WSC) that describes the features from left to right. Later, a similar idea is adapted to word spotting on cursive handwritten documents with a modified character shape code by S. Sarkar [Sarkar 2013]. Moreover, with the work of [Pourasad 2012], it turns out that ascenders/descenders are discriminant characteristics not only for Latin language but also for Farsi and Arabic scripts.

Ascenders and descenders are identified based on the baseline detection method introduced in Chapter 3 (see Fig. 4.24). This method is applied to connected components. Instead of using the same baselines for the entire line, each connected component has its own baselines. The reason of doing this comes mainly out of the consideration that some scripts can be very cursive for the content within the same line. What is more, the baselines can also vary in the same word itself in handwriting. In such a context, a uniform baseline system is definitely not appropriate.

For comparison, we calculate the difference in the numbers of ascenders/descenders between the query and the regions of interest.

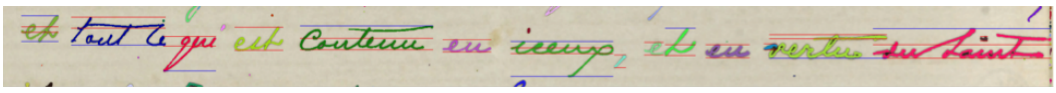


Figure 4.24: Ascender/descender detection (the top and bottom baselines are blue, and the two middle baselines are red.)

Loop detection and identification In the study of handwriting representation, loop is found to be one of the most dominant characteristics available in cursive handwriting processing. Several previous researches like [Blankers 2007, Steinherz 2009] have shown the importance and usefulness of loops. In particular, loops are often considered to be the key to successful offline and online word

recognition systems. As a prominent pattern, loop is very distinguishing no matter in term of letters or writing styles [Blankers 2007].

A loop is a handwritten pattern, made of several strokes formed when the writing instrument returns to a previous location while touching the pad continuously, giving a closed outline with a "hole" in the center [Steinherz 2009]. In the context of handwriting, there are two types of loops. One is obvious loops, which can be visually perceived, while the other type is hidden, such as the loop in letter "e" in Fig. 4.25. In our case, we consider only the obvious loops like the one letter "h" or "g" in Fig. 4.25.

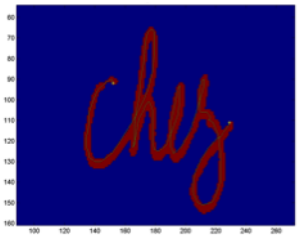


Figure 4.25: Obvious and hidden loops

Taking structural points as vertexes, strokes as edges, we simplify scripts into undirected graphs without labels, which is a classic and compact representation method containing all the structural information of the text in its two-dimensional configuration. To detect loops (cycles) in graph representation, we mainly use a modified depth first search (DFS) method, called colored DFS. The original DFS is an algorithm for traversing or searching tree or graph data structures. The colored DFS method performs on the basis that an undirected graph has a cycle (loop) if and only if a DFS finds an edge that points to an already-visited vertex (a back edge). To distinguish visited vertexes from non-visited ones, we give them different colors. All vertexes are initially marked white. When a vertex is encountered, it is marked grey, and when its descendants are completely visited, it is marked black. If a grey vertex is encountered, then there is a cycle.

In the literature of writer identification, the exploration of loops is very fruitful. The work of [Blankers 2007] reveals that there is a similarity of different loops produced by a single writer, which means that the loops occurring in the "l"s, will be very similar every time this writer writes the letter "l", and may even bear resemblance to the loops they produce in other letters containing ascenders, such as "h", "k". Inspired by this discovery, we build a loop dictionary for the collections of the same writer by clustering loops into classes. The size of the dictionary varies as the writing style changes. Usually, we cluster the loops into 2 classes. One class is elongated loops usually appearing in ascenders or descenders, such as the loops in "l", "h", "y" and "j". The other class represents the loops appearing in the base of the word, like the loop in letters "a", "d", "e" and "o". Sometimes, according to writing style differences and application requirements, the number of classes can be

3, 4 or more. After several experiments, the features chosen for clustering are the perimeter and area of loop and the ratio of the area of the loop to the area of the bounding box. With the loop dictionary, each loop is assigned a label.

A. Perimeter

The perimeter of the loop is computed as the total amount of pixels on the trajectory.

B. Area

A loop can be considered as a polygon, of which the area can be computed as follows:

$$Area = \frac{1}{2}\sum_{i=0}^{N-2}(x_i y_{i+1} - x_{i+1} y_i) \tag{4.15}$$

C. Area of loop/area of bounding box

The area of bounding box is calculated as the product of the height and width of the loop.

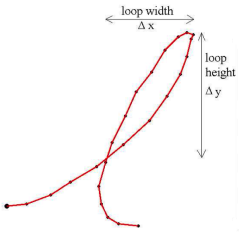


Figure 4.26: Width and height of a loop

Fig. 4.27 gives an example of loop detection and identification.

To compare the script by checking the loop pattern, we encode labels of loops into a sequence following the direction of the trajectory. To measure the differences between query and test instances, a string editing distance is employed.

Global properties Features like loops, ascenders/descenders are categorized into high-level feature group, which are regarded as more semantic features in relation to the meaning of handwritten shapes. The presence of certain strokes or loops can be very useful for distinguishing word shapes. However, the drawback of such features is their instability. It might be caused by the inability of the detection approach to cope with the ambiguity of handwriting. Or it is because of the inconsistency of the writer. To handle this problem, we also include global properties obtained in the coarse selection (Section 4.2.2.1) in the final description of handwriting as a complement. After all, as low-level features, the projection profile, the word profile and the orientation distribution are relatively stable.

In this way, the word image representation is complete, containing both high-level and low-level, morphological and topological descriptions.

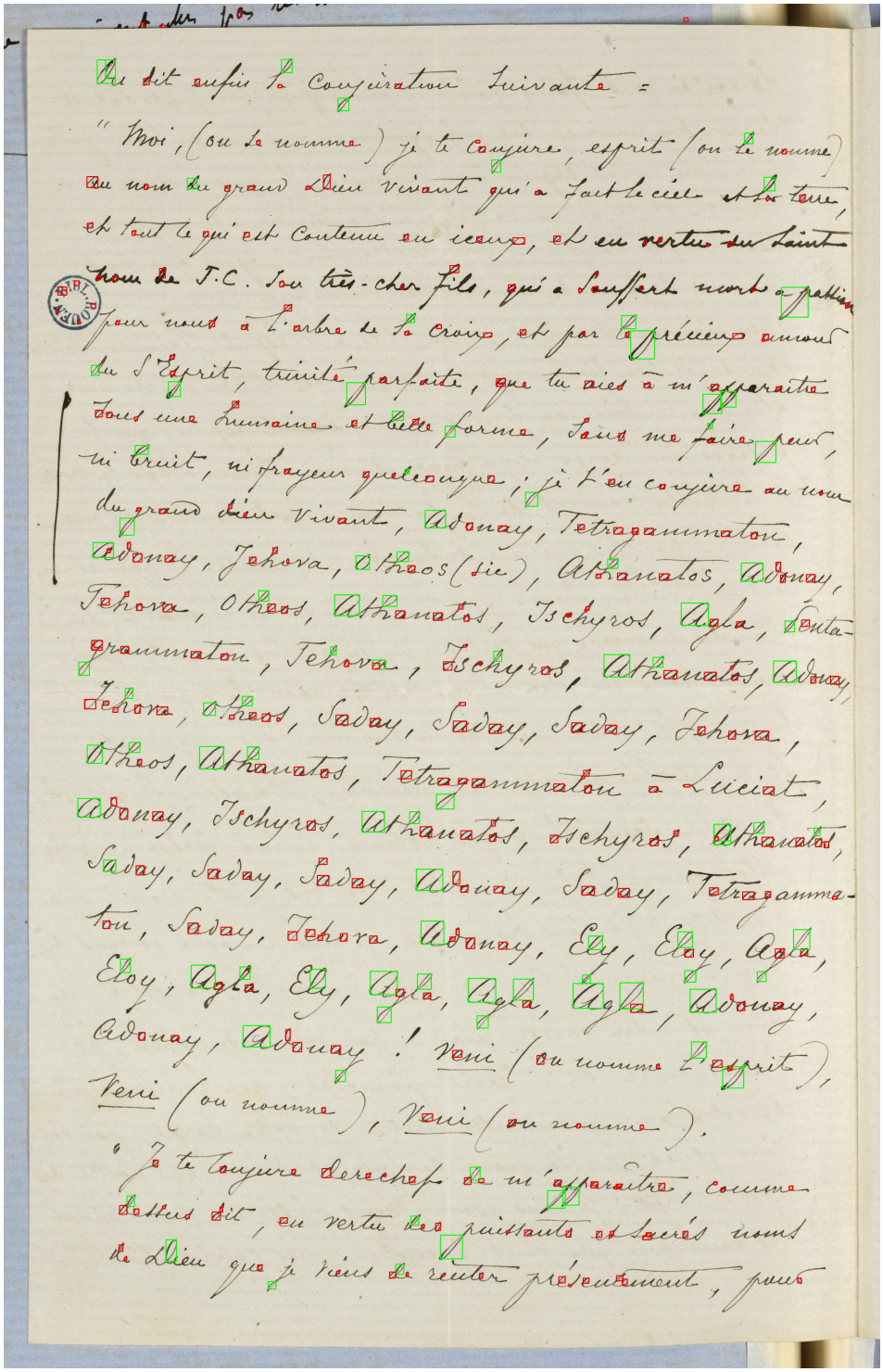


Figure 4.27: Identification and classification of loops (colors indicate classes)

Mixed similarity measure In order to use descriptions of different aspects together to measure the similarities between the query and the regions of interest, we choose first to calculate the distance in each description and sum them up. We design the final normalized distance function for similarity comparison as Eq. 4.16. It is composed of four parts, taking into consideration contextual information (by the SC description), topological and structural information (through loops and strokes) and textural information (the result obtained in the coarse selection (Section 4.2.2.1)).

$$D = w_{SC}D_{SC} + w_{ad}D_{ad} + w_{loop}D_{loop} + w_{texture}D_{texture} \tag{4.16}$$

where D_{SC} is defined as Eq. 3.14 in chapter 3, $D_{texture}$ is defined as Eq. 4.8, and D_{ad} and D_{loop} are the edit distances of the corresponding loop label and ascender/descender sequences.

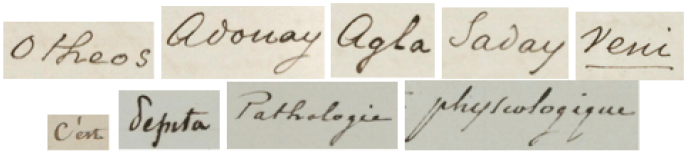
After obtaining the different distance values relative to each part of the shape description, we proceed to a normalization step so as to ensure a final normalized D value. The weights for each part are experimentally chosen so as to highlight visually the strongest properties of the handwriting (cursive, elongated, round ...). They can also be adjusted by users for particular purposes or by taking into account specific knowledge on the writing styles. Afterwards, an automatic weighting procedure is developed in Section 4.2.3.2

Evaluation In order to show that the proposed representation model with mixed similarity measure overcomes the drawbacks of using the information from a single shape aspect, we compare our results with the approach that uses only the Shape Context descriptor.

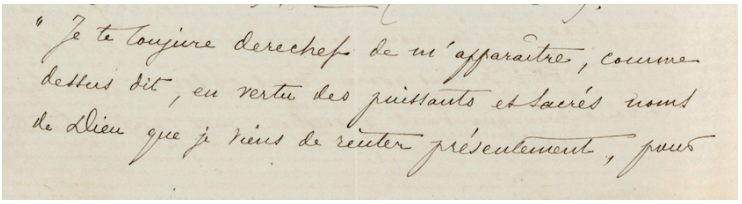
The evaluation dataset is related to the CITRE project. It is composed of letters written by different French philosophers and constitutes 4 collections. There are 11 pages containing approximately 2000 words. From all the word classes in the dataset, we select a subset of 9 classes (a class = a set of the same word instances). The number of samples per class varies from 2 to 12. Each sample in 9 classes is used as a query and this makes a total of 51 queries (see Fig. 4.28).

The evaluation protocol is the following: for each query, we rank all the regions that are first selected after the coarse orientation and projection profile based selection (around 250-350 regions on average). A region is classified as positive if it overlaps by more than 50% with the annotated bounding box in the ground truth, and negative otherwise. We combine the retrieved regions of all the documents and rerank them according to their distances. For each word class, we report the Average Precision (AP) and Average Recall (AR) for 5 ranks (rank 3, rank 5, rank 10, rank 20 and rank 30), which are standard measures in retrieval systems. Overall results are evaluated by computing the mean Average Precision (mAP) and mean Average Recall (mAR) over the 9 classes.

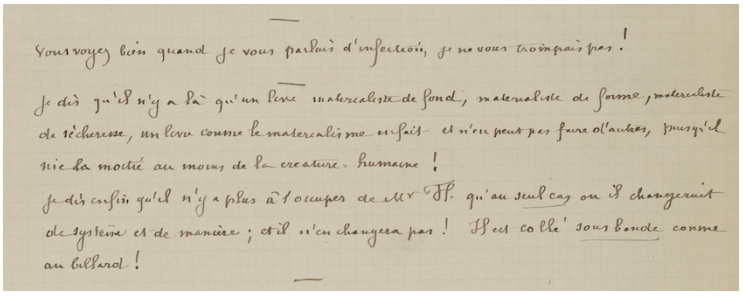
In Table 4.3 and 4.4, we show respectively the results of using only Shape Context descriptor and the results of using our mixed approach. We must emphasize that the results presented in this part are the performance of the global approach. Since the



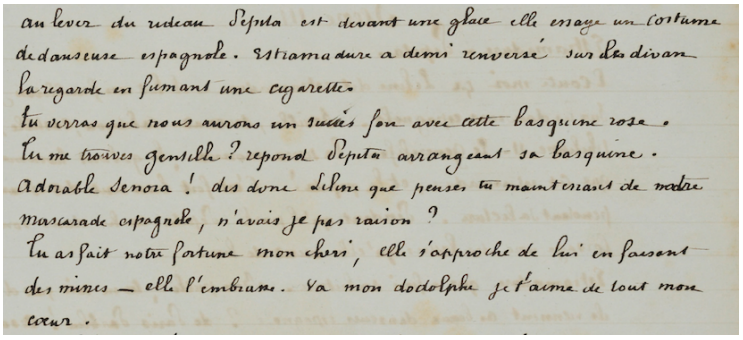
(a)



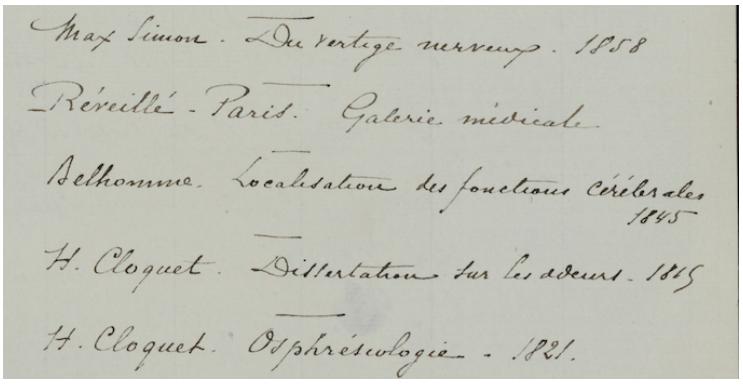
(b)



(c)



(d)



(e)

Figure 4.28: (a) Examples of query words (b) an example of collection 1 (c) an example of collection 2 (d) an example of collection 3 (e) an example of collection 4

procedure in the coarse selection step is the same, the comparison of SC approach with our approach can be indicated by the global performance comparison. We present the results here according to writing styles. The word classes from the same collection are summarized together. Collection (Col.) 1 contains word classes "*Savay*", "*Alga*", "*Avouay*", "*Otheos*" and "*reui*"; Col.2 is the word class "*Sepita*"; Col.3 contains word classes "*Pathologie*" and "*physiologique*"; Col.4 is the word class "*c'est*". It can be seen that from the overall mAR and mAP, our approach gives better results than the use of Shape Context descriptor, which takes into account only contour and contextual information of the words but nothing about their inner structures.

%	AR(Average Recall)					AP
Shape Context	Rank3	Rank5	Rank10	Rank20	Rank30	(Average Precision)
Col.1	39.17	49.28	59.50	70.33	72.33	37.99
Col.2	65.56	74.45	77.78	77.78	77.78	85.50
Col.3	48.71	57.95	66.76	77.76	83.05	75
Col.4	0	2.5	12.5	17.5	32.5	8.22
mAR/mAP	40.48	49.76	58.53	65.98	69.68	51.68

Table 4.3: The average recall and average precision for the different classes using only the Shape Context descriptor

%	AR(Average Recall)					AP
Proposed approach	Rank3	Rank5	Rank10	Rank20	Rnak30	(Average Precision)
Col.1	43.61	49.25	60.83	70.08	74.33	48.58
Col.2	71.12	76.67	76.67	76.67	76.67	84.72
Col.3	48.71	57.95	66.76	77.76	83.05	73.76
Col.4	15	15	18	35	45	16.4
mAR/mAP	47.11	52.50	60.25	68.50	72.56	55.87

Table 4.4: The average recall and average precision for different classes using our proposed approach

It has been observed that the results of the Col. 4 are much worse than the others. This is mainly because the writing style of the Col. 4 is very compressed and dense as shown in Fig. 4.29 and also because the query "*c'est*" used in Col. 4 is very short. The inexistence of any very significant strokes or loops, and the lack of contour specificities are the explanation of such results. With such a limited amount of information, the distinguishing capabilities based on the Shape Context and elongated strokes decrease a lot. Fig. 4.30 illustrates one reason for the decrease of mAP, that is the confusion with similar shaped words. This phenomenon cannot be considered as an error of the system but as an enlargement of the set of possible solutions to be proposed to users.

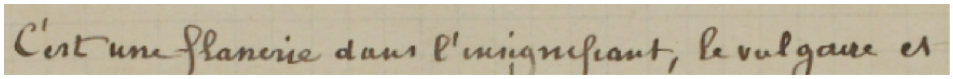


Figure 4.29: Example of compressed and dense writing style (Col. 4)

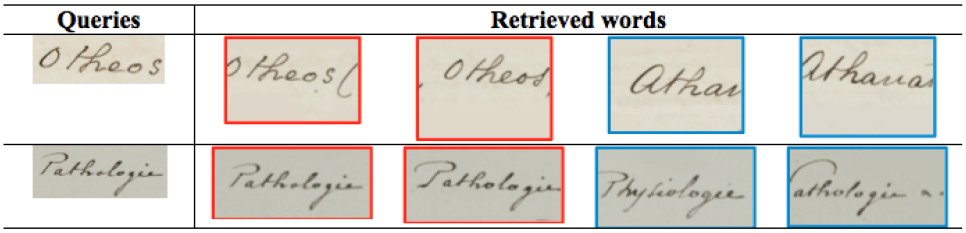


Figure 4.30: Examples of failure cases of the proposed method: words with very similar shapes

4.2.3.2 A selection based on the comprehensive description of the handwriting

After the first attempt, we continue to work on developing a comprehensive representation model fusing the properties from both morphology and topology of handwriting. This time, we keep the identical SC description for the morphological part and make a deeper exploration on graph representation and extract a purely topological descriptor, a topological node feature (TNF) that describes the local topological information in the neighborhood of the vertex in the graph. The details are given below.

Image representation

- Morphological description

  To describe the contour, we maintain the Shape Context (SC) descriptor according to the signatures of the text. To take advantage of the structural characteristics of the text, we use the structural points instead of a large set of contour points as interest points so as to preserve more specific salient information in the formation of the handwriting trajectory. The number of points considered is relatively small but it is compensated by their structural significance and precise location in the word. The similarity distance based on SC description is calculated as Eq. 3.14 in chapter 3.

- Topological description

  Graph representation is a very powerful method of representing images in terms of preserving structural properties. In our scenario, the graph-based representation model is established on the skeleton of the handwriting. The

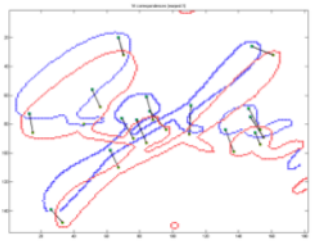


Figure 4.31: Word matching using Shape Context

vertexes of the graph correspond to structural points and strokes connecting them are considered as the edges.

Topological Node Feature The Topological Node Feature (TNF) is a point feature that describes the local topological information about a vertex (node), which is calculated solely on graphs. It was originally used for identifying vertex compatibilities, which is also known as subgraph isomorphism. We adapt it into our application in order to make comparison of word images in terms of the topology. The influence of local neighborhood is noteworthy in the context of word comparison because the local handwritten information about the shape is very poor. The relation that a point or a stroke has locally with its neighbor reinforces the contextual deterministic description.

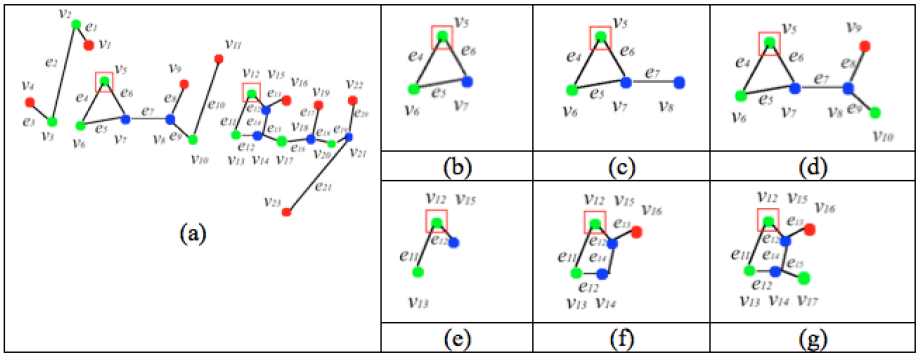
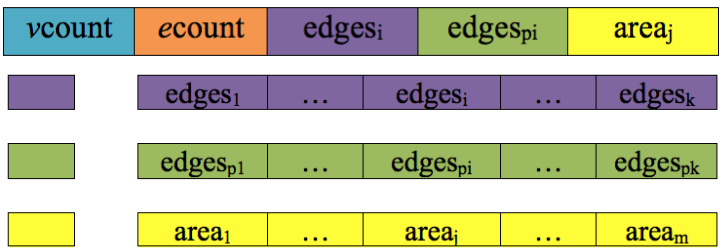


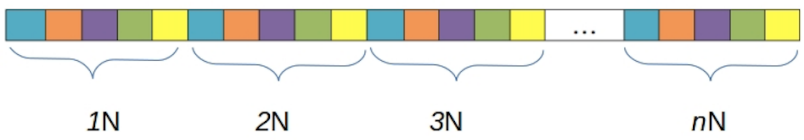
Figure 4.32: (a) Graph representation of the word "*Savay*". (b) (c) (d) the 1- , 2- , 3- neighborhood of the vertex v_5 (e)(f)(g) the 1- , 2- , 3- neighborhood of the vertex v_{12}.

The traditional TNFs are composed of features such as degree, clusterc, $n$cliques$_k$, $n$walks$_{pk}$, and so on. In our case, we use the topological n-neighborhood features proposed by N. Dahm et al. [Dahm 2013]. An n-neighborhood of a vertex v (denoted as $n\mathrm{N}(v,\ n)$) is an induced subgraph

formed from all the vertexes that can be reached within n steps from v. This induced subgraph is centered on the vertex v and contains all vertexes up to n steps away, and all edges between those vertexes. Fig. 4.32 gives an illustration of the n-neighborhood concept by showing the example of the neighborhoods of three levels of two vertexes. For any single vertex v, a unique nN may be created for each value $n = 1, 2, \cdots, m$, where nN(v, m)=G (the entire graph can be reached in m steps). The TNFs extracted from different size n-neighborhoods can give different performances. Within a certain-step neighborhood, we calculate and record five properties of the neighborhood as the TNFs used in our case. These five properties are respectively $v$count (the number of the vertexes in the nN), $e$count (the number of edges in the nN), edges$_i$ (the number of edges belonging to the ith class in the nN), edges$_{pi}$ (the number of edges belonging to the ith class in the nN, which pass through the main vertex), and area$_j$ (the location distribution of the edges in area j). It should be noted that we cluster the edges into k classes first according to their lengths. In practice, we use $k = 5$ to distinguish the edges. Other values for k have been tested on the validation dataset. It turns out that $k = 5$ is the best option. For the property "area$_j$", we take the reference vertex as the centroid of the polar diagram, with the neighborhood evenly divided into m (m=8) areas. We calculate the distribution of the edges in each area. Fig. 4.33 presents the composition of the TNF in our application.



(a)



(b)

Figure 4.33: The illustration of the TNF vector in our case (a) a unit of features calculated for a single level neighborhood (b) the composition of an nN TNF descriptor, which contains the unit of features in (a) for each level of neighborhood

After obtaining the TNFs from the graph representation, we first adapt the popular Bag-of-Visual-Words (BoVW) framework to classify all occurrences of TNFs over all word images. The TNFs are quantized into visual words by using a codebook that is obtained by clustering the feature space into n_{cb} clusters, i.e. the visual word that is assigned to a TNF feature corresponds to the index of the cluster that the TNF belongs to. In order to efficiently select the best size n_{cb} of the codebook, we use X-means [Pelleg 2000] algorithm to automatically find out the optimal amount of clusters. For the similarity measure, we adapt the same distance calculation as is used in the baseline framework of the BoVW approach. Considering a query image P and a test image Q, Eq. 4.17 gives the distance definition for the TNF-based BoVW representation.

$$D_{TNF} = \sum_{i=1}^{n_{cb}} (h_P(i) - h_Q(i))^2 \tag{4.17}$$

where $h(\cdot)$ denotes the frequency of the class.

Hybrid weighted similarity measure based on LDA optimization With various descriptions of handwriting, we choose to adapt the late fusion strategy [Moulin 2014] to fuse all the information. In other words, each feature is processed independently and the distance is calculated separately. Then the final distance is computed as the overall shape distance (Eq. 4.18), i.e. the weighted sum of individual distances given by all independent measures: the Shape Context distance, and the TNF-based BoVW comparison. A linear combination strategy, as one of the simplest and most widely used solutions for fusing dissimilarity information from different aspects, is employed. For such a combination, it is obvious that the choice of the fusion parameters for the final distance expression is very sensitive to the nature of descriptors (e.g. morphological and topological) in regard to writing styles. It seems that the weights assigned to different descriptions should not be the same because the effectiveness of each description is not equivalent for all writing styles (and even for all sizes of queries). It is not easy to set these parameters, even for an expert.

$$D = w_{SC} D_{SC} + w_{TNF} D_{TNF} \tag{4.18}$$

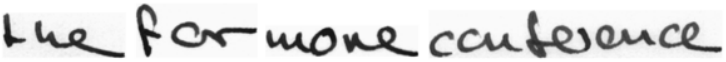
To solve this problem, we choose to use the Linear Discriminant Analysis (LDA) for optimizing the weights in Eq. 4.18 by using only a certain amount of training data. The works presented in [Moulin 2014] and [Zhu 2009] have proved that the LDA can provide a nearly optimal learning of the weights with an efficient computation. We reformulate the learning of the weights as a dimensionality reduction problem in a binary classification context. With the LDA, this problem can be solved analytically. It has to be pointed out that the LDA in our application is not used to build a classifier but to find the best linear combination, which separates positive and negative samples for all the training queries. For this reason, this analysis does not require any hypothesis on the distribution of the variables. In the end,

the final distance is identified as the combination of the distances of the SCs and the TNFs with the optimal weights.

Evaluation In this part, two issues concerning the performance are investigated in the experiments: the performance of the combination of the SCs and TNFs instead of the single one, and the effect of using different weights for different distances. The experiment data and protocol are introduced in the first part of the section. The results and discussion are given in the latter part.

For the experimental evaluation, the proposed approach is performed on two public datasets, including both historical handwriting and modern handwriting: the BH2M dataset and the IAM off-line dataset. Both corpus are accurately segmented into words and transcribed. All the words containing at least 3 letters and appearing at least 10 times in the collection are selected as queries. In this way, there are 514 queries corresponding to 32 different words for the BH2M dataset and 765 queries from 27 different word classes for the IAM off-line dataset. Some examples are shown in Fig. 4.34.

(a)



(b)

Figure 4.34: Examples of query words used in the experiments. (a) the BH2M dataset (b) the IAM off-line dataset

Concerning the LDA learning process for the optimal weights, we use 3-fold cross-validation. In order to ensure the diversity of the queries and the fairness of the experiment, the procedure is repeated 10 times. Each time, one third of the queries for each dataset are used for the estimation of the weights, i.e. 171 queries for BH2M dataset and 255 queries for IAM dataset. The final evaluation is calculated as the mean performance of 10. Additionally, we also manually set different weights and test the performance.

We present in Fig. 4.35 some qualitative results of the method. Concerning the quantitative evaluation, Table 4.5 and 4.6 respectively give the retrieval results for the BH2M dataset and the IAM off-line dataset by using only the SCs, or the TNF-based BoVW, and using the proposed approach (with optimal weights). For the topological part, we empirically use a codebook of 50 classes ($n_{cb} = 50$) for the TNFs of *2*N (the BH2M dataset) and 40 classes ($n_{cb} = 40$) for the TNFs of *5*N (the IAM dataset). It can be seen that the proposed approach outperforms the approaches of using each single property for the BH2M dataset, which indicates the significance of employing a comprehensive model comprising properties of handwriting from

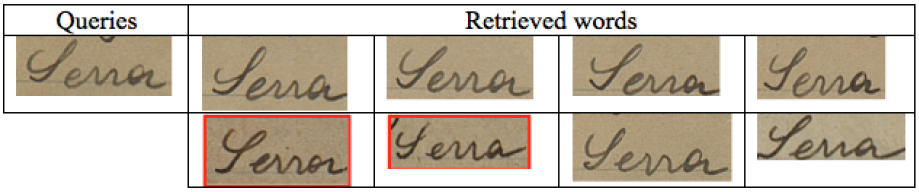


Figure 4.35: Qualitative results: one query and top eight retrieved words (the responses with the red bounding box are failures)

different aspects. For the IAM off-line dataset, our approach does not achieve as good a performance as using only the SCs. This is mainly caused by the simplicity of the structure in IAM handwriting. It can be observed that the handwriting in the IAM dataset is hastier and less decipherable than the BH2M one (see Fig. 4.35). Consequently, the structural points are less numerous and inconsistent, which explicitly changes the graph representation and makes the BoVW representation very sensitive to distortion. In this situation, the SCs prove their powerful strength by including the entire contour of the word in a more global point of view. Fig. 4.36 shows one example of such a case. For five instances of "*to*", we choose one high-curved point that appears in all five instances to study respectively its Shape Context descriptor and the five neighborhoods that are used for calculating the TNF features. It can be seen that the Shape Context descriptors of five corresponding points are very similar. However, the neighborhoods of different levels are dissimilar due to different graph configurations of each instance. Besides, the maximal number of structural points of five instances is only 13. Compared to the other scenarios where the BoVW model is used, the points are very few, which leads to the fact that the wrong classification or mis-detection of a point has a big influence on the final representation of the image.

Moreover, we also make the test to compare the retrieval performance of using optimal weights in the distance measure to the one without giving any weights, which can be also understood as w_{SC}=1 and w_{TNF}=1. It can be noticed in Table 4.7 that the performance is much improved by using optimal weights learned by the LDA. For the IAM off-line dataset, our approach achieves better performance considering the mAP, and is less competent for P@10 and P@20. Our linear combination approach can be used as a flexible tool, which allows the user to check the best distance depending on his requirements.

4.2.3.3 Fine selection using sub-optimal graph edit distance with merging operation in a compact structural way

The final approach that we develop for the fine selection is inspired by the structure-based approaches introduced in section 4.1.1.2. It is based on a novel combination of the Shape Context and the skeleton. Both the morphology and the topology of

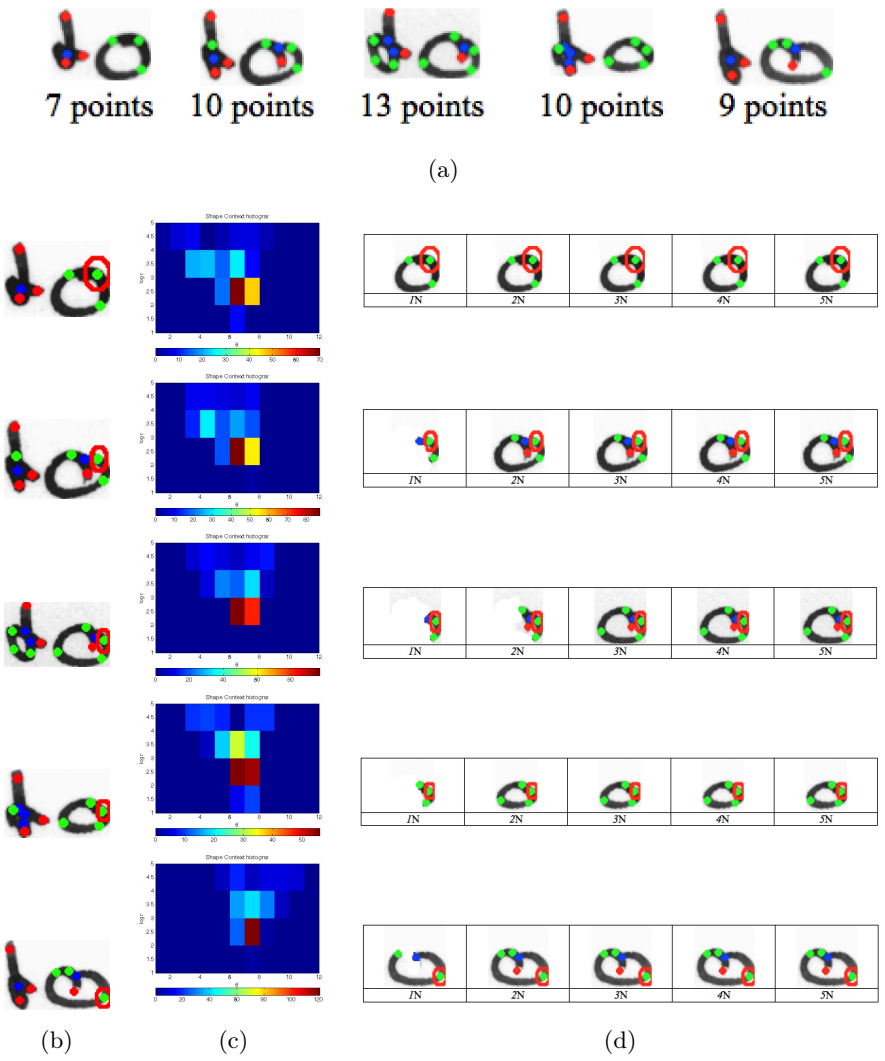


Figure 4.36: The ambiguity of the scripts in the IAM dataset (a) instances of the same word with different amounts of structural points (b) the selected points (c) the corresponding SC descriptors (d) the neighborhoods of five levels of each instance

	P@10	P@20	mAP
SC	0.428	0.298	0.329
TNFs-BoVW	0.141	0.084	0.095
Proposed (no weights)	0.416	0.275	0.305
Proposed (opt. weights)	0.492	0.333	0.376

Table 4.5: Retrieval result for the BH2M dataset (validation set)

	P@10	P@20	mAP
SC	0.680	0.558	0.433
TNFs-BoVW	0.193	0.134	0.084
Proposed (no weights)	0.548	0.435	0.289
Proposed (opt. weights)	0.447	0.418	0.301

Table 4.6: Retrieval result for the IAM off-line dataset (validation set)

the handwriting are integrated into a labeled graph representation. Moreover, a specialized similarity measure based on the graph edit distance is designed. DTW and block merging techniques are introduced in order to make the comparison robust to the variations of the handwriting.

Image representation Since previous works have demonstrated the benefits of combining the topological and morphological description together in the representation of word images, by using labeled graph model, we find a more compact way to represent handwriting images. In this section, we adapt our proposed graph-based representation model previously introduced. The details can be found in chapter 3.

Similarity measure

Connected component comparison Since connected components are represented as graphs, the comparison among words can be solved based on graph matching. To avoid the computational burden, we choose to use an approximate graph edit distance algorithm proposed by K. Riesen and H. Bunke [1]. The edit distance between two attributed graphs is defined by the minimum cost edit path between two graphs.

Definition 3. (Graph edit distance) Let $g_1 = (V_1, E_1, \mu_1, v_1)$ be the source and $g_2 = (V_2, E_2, \mu_2, v_2)$ be the target graph. The graph edit distance between g_1 and g_2 is defined by

$$d(G_1, G_2) = min_{e_{d1},\cdots,e_{dk} \subseteq h(G_1,G_2)} \sum_{i=1}^{k} c(e_{di}) \tag{4.19}$$

	P@10	P@20	mAP
Proposed (opt. weights)	0.492	0.333	0.376
Graph-based approach [Wang 2014a]	0.342	0.241	0.246

Table 4.7: Comparison with the approach presented in [Wang 2014a] on the BH2M dataset

where $\gamma(g_1, g_2)$ denotes the set of edit paths transforming g_1 into g_2, and $c(e_i)$ denotes the cost function measuring the strength of edit operation e_i (insertion, deletion and substitution of vertexes and edges).

This suboptimal algorithm is based on the decomposition of graphs into sets of subgraphs. These subgraphs consist of a vertex and its adjacent structures. In this case, the graph edit distance is approximately reformulated as the optimal match between vertexes and their local structures of two graphs. Instead of using dynamic programming, bipartite matching is adapted to find the optimal match. Subgraph matching is treated as an assignment problem. The Munkres (Hungarian) assignment algorithm [Munkres 1957] is used to find an optimal assignment of all subgraphs, which minimizes the sum of the assignment cost.

To take into account local structure of graph, the cost of vertex substitution is composed of two parts. The first one is the cost calculated from the attributes of two vertexes. The second part comes from the local structure comparison, as shown in Eq. 4.20. After normalizing both of them into [0, 1], each part is given a different weight. We have tested the performance of graph edit distance with several different sets of weights ($w_1 = 0.2$ and $w_2 = 0.8$, $w_1 = 0.5$ and $w_2 = 0.5$, $w_1 = 0.8$ and $w_2 = 0.2$). The results reveals that in our scenario, the description given by the Shape Context is more discriminant and important than the information depicted by the adjacent structure in terms of the subgraph. Hence, we assign 0.8 to w_1 and 0.2 to w_2.

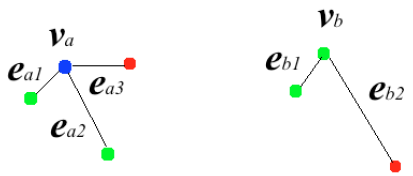
$$C_{substitution} = w_1 C_{SC} + w_2 C_{local\_structure} \tag{4.20}$$

$$C_{deletion} = d \tag{4.21}$$

$$C_{insertion} = a \tag{4.22}$$

where a and d are constant values set experimentally. Since the range of substitution cost is from 0 to 1, we use 0.5 as the insertion and deletion cost. C_{SC} is the cost of the Shape Context. $C_{local\_structure}$ is the graph edit distance calculated on the adjacent edges. The costs of the deletion and insertion of edge are constant values, while the substitution cost equals $1 - \frac{e_{short}}{e_{long}}$, where e_{short} denotes the length of the shorter edge and e_{long} denotes the length of the longer edge. Take two vertexes v_a and v_b shown in Fig. 4.37 for example. When we calculate $C_{local\_structure}$ between v_a and v_b, we also need to build a cost matrix like Eq. 3.23, and input the cost matrix to the Hungarian algorithm to obtain the optimal edit operations for edges. Similarly, in order to build the cost matrix, we need to calculate the costs for

insertion, deletion and substitution. The costs for insertion and deletion are constant values as mentioned before. For the substitution cost, if we want to replace e_{a1} by e_{b2}, then e_{a1} corresponds to the e_{short} and e_{b2} corresponds to e_{long}. The substitution cost between e_{a1} and e_{b2} equals $1 - \frac{e_{a1}}{e_{b2}}$.

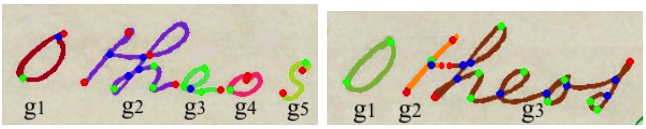


(a) (b)

Figure 4.37: Illustration of edge substitution (a) a branch point (b) a high-curved point

Typically, the graph edit distance is calculated with a tree search of an exponential time complexity. However, the chosen approximate graph matching approach can solve the bipartite matching problem in polynomial time.

Word Image Matching Since a word image is represented as a sequence of connected components, DTW is adapted in order to find the assignments among connected components using the graph edit distance introduced in the previous part. As a major variation of the handwriting, the number of connected components of the same word can vary a lot in different instances, even in the same writing style. It potentially results in n to m mapping from one sequence of graphs to the other (n and m are respectively the numbers of graphs in two words). Fig. 4.38 gives an example where two instances of one word have respectively 5 and 3 connected components.

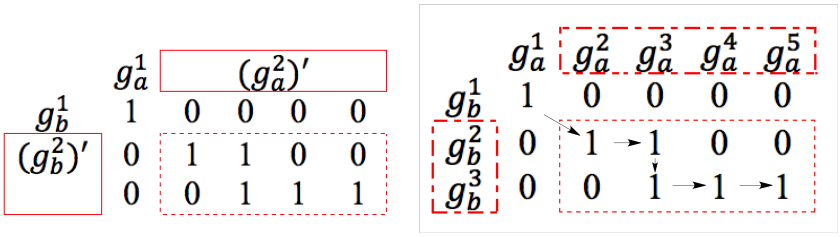


(a) (b)

Figure 4.38: Two instances of the word "*Otheos*" with different numbers of connected components

In order to make the similarity measure robust to such variation, based on the matching results obtained in the previous step, an exhaustive merging operation is performed. The graphs assigned to the same connected component are considered as one entire graph. In this way, both the sequences of graphs representing the query and the test image might change. Afterwards, with the new sequences of graph representation, the graph edit distance is calculated again between new corresponding

pairs of graphs. The average distance between the new corresponding graph matchings is considered as the final distance between two word images. Take two instances of the word "*Otheos*" in Fig. 4.38 for example. Given $a = g_a^1, \cdots, g_a^5$, represents the word in Fig. 4.38 (a) and $b = g_b^1$, g_b^2, g_b^3, represents the word in Fig.4.38(b). After the DTW alignment, it shows that g_a^1 corresponds to g_b^1, g_a^2 is assigned to both g_b^2 and g_b^3, g_a^3 , g_a^4, g_a^5 correspond to g_b^3. Therefore, in the next step, g_a^2, g_a^3, g_a^4, g_a^5 are merged together as an entire graph $(g_a^2)'$, because all of them are matched to g_b^3. g_b^2 and g_b^3 are merged also as $(g_b^2)'$, because both of them are matched to g_a^2. In this way, both image (a) and image (b) are reformulated as the sequences of two connected graphs. The graph edit distance is calculated again between g_a^1 and g_b^1, $(g_a^2)'$ and $(g_b^2)'$, which are respectively noted as $ged(g_a^1, g_b^1)$ and $ged((g_a^2)', (g_b^2)')$. The final distance between two words *d(a,b)* is the average of $ged(g_a^1, g_b^1)$ and $ged((g_a^2)', (g_b^2)')$. Fig. 4.39 visually presents the procedure explained above.



(a) (b)

Figure 4.39: Illustration of the block merging process based on the DTW assignment results (the arrows indicate the optimal route of matching)

4.3 Evaluation

In this part, the proposed graph-based word spotting approach is applied to two datasets. One is the George Washington dataset and the other is the BH2M dataset. All the words containing at least 3 letters and appearing at least 10 times in both datasets are selected as queries. In this way, there are 1847 queries corresponding to 68 different words for the George Washington dataset and 514 queries from 32 different word classes for the BH2M dataset.

The results are compared to the performance of five other word spotting approaches. They are respectively sequence alignment using DTW with Manmatha's features [Rath 2007], a bag of visual word approach as statistical model [Lladós 2012], a pseudo-structural model based on a Loci feature representation [Lladós 2012], a graph-based approach using the bag-of-paths descriptor [Lladós 2012] and Leydier's approach [Leydier 2009] based on the gradients and cohesive matching.

Fig.4.40 presents the plots of precision-recall curves for both the George Washington dataset and the BH2M dataset. In table 4.8 and 4.9 the evaluation of the

performance for both datasets is given.

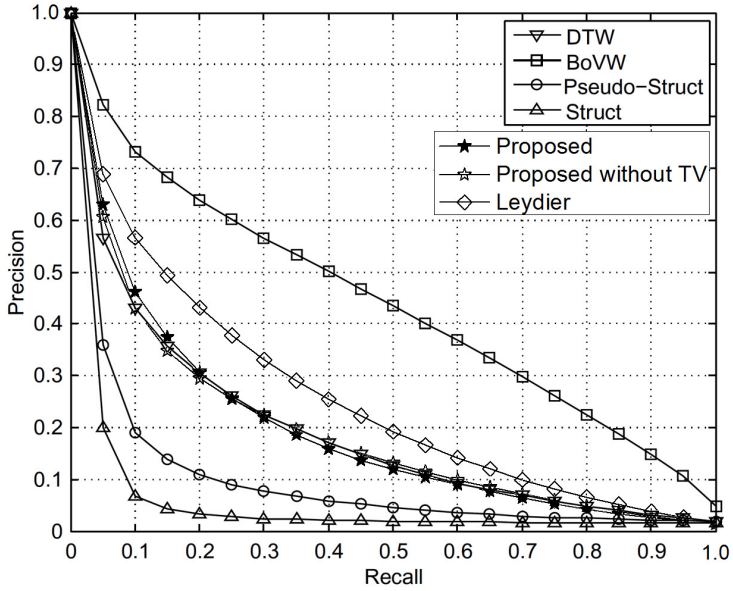
	P@10	P@20	Rprecision	mAP
Proposed approach	0.393	0.329	0.203	0.175
Proposed approach (without TV)	0.372	0.312	0.206	0.175
Manmatha et al. [Rath 2007]	0.346	0.286	0.191	0.169
BoVW [Lladós 2012]	0.606	0.523	0.412	0.422
Pseudo-Struct [Lladós 2012]	0.183	0.149	0.096	0.072
Structural [Lladós 2012]	0.059	0.049	0.036	0.028
Leydier et al. [Leydier 2009]	0.449	0.359	0.269	0.238

Table 4.8: Retrieval result for the George Washington dataset (TV = tensor voting)

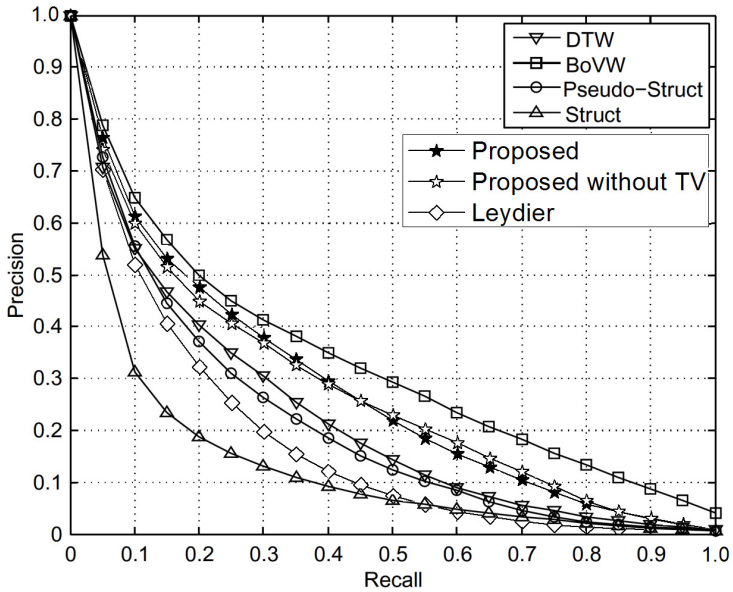
	P@10	P@20	Rprecision	mAP
Proposed approach	0.347	0.246	0.273	0.247
Proposed approach(without TV)	0.342	0.241	0.270	0.246
Manmatha et al. [Rath 2007]	0.288	0.201	0.214	0.192
BoVW [Lladós 2012]	0.378	0.289	0.303	0.3
Pseudo-Struct [Lladós 2012]	0.273	0.189	0.199	0.178
Structural [Lladós 2012]	0.155	0.12	0.118	0.097
Leydier et al. [Leydier 2009]	0.235	0.153	0.170	0.145

Table 4.9: Retrieval result for the BH2M dataset (TV = tensor voting)

We can see that in both scenarios, the proposed approach outperforms the DTW-based, the pseudo-structural and the structural methods and it also gets better results than Leydier's approach on the BH2M dataset. The BoVW method achieves the best performance on both datasets. This is mainly because the BoVW method extracts word description directly from the grey-level image without any binarization process. All the other methods except Leydier's approach in the comparison establish the representation based on the binarized images. The problem caused by the ink degradation is clearly magnified by the binarizing process. In our approach, the discontinuity of the skeleton caused by the binarization directly leads to the deformation of the graph representation and increases the unstability of connected components (see Fig.4.41). With the tensor voting technique, the discontinuity appearing in the skeleton can be recovered to some extent. However, when the gap is too large, we still cannot join two broken parts. Even though the specific similarity measure designed for the graph-based representation can tolerate this distortion, the final distance between images can still be slightly impacted. This is why the appearance-based approaches (BoVW, Manmatha's approach and Leydier's approach) generally perform better than the structure-based approaches (the proposed approach, Pseudo-struct and structural). Moreover, compared to other methods, the BoVW method produces a more dense description by extracting visual words based on SIFT detection and represents the handwriting in a more statistical way. How-



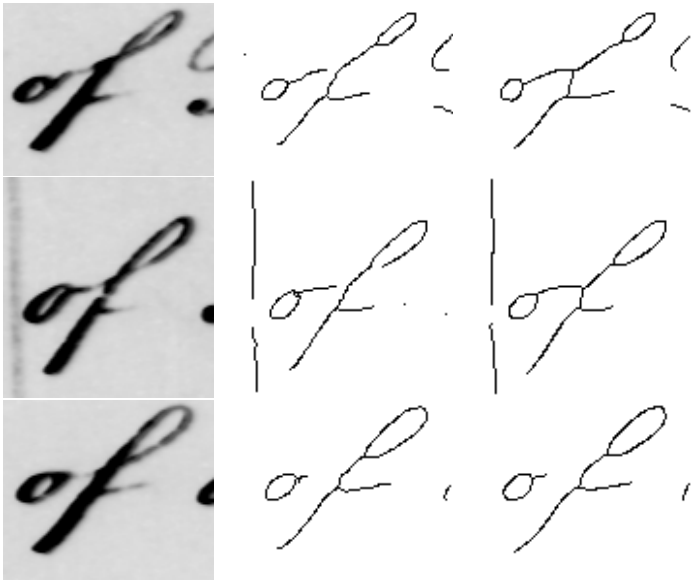
(a)



(b)

Figure 4.40: Precision and recall curves for (a) the George Washington dataset and (b) the BH2M dataset

ever, the classification of visual words and the final high-dimensional feature vector of the image in the BoVW method greatly increase the complexity and the computational cost of the approach. It is interesting to point out that the performance of the proposed approach is better on the BH2M dataset than the George Washington dataset. The reason is that most word images in the George Washington dataset have more degradation than the images in the BH2M dataset. As a conclusion, the main constraint of the proposed approach is the quality of the skeleton. It can be solved by adapting a more effective approach or adding some intersection points to increase the stability of the skeleton.



(a) (b) (c)

Figure 4.41: Skeleton examples for different instances of the same word class for the George Washington dataset (a) the original images (b) skeletonization without applying the tensor voting technique (c) skeletonization with application of the tensor voting technique

Apart from the BoVW method, the proposed approach is proved to be more competent than four other methods in the comparison on the BH2M dataset. Especially, the fact that the proposed approach greatly outperforms the pseudo-structural and the structural methods demonstrates that using structural points to construct the graph is very informative, and furthermore, the integration of contextual information depicted by the Shape Context in the graph representation is necessary and promising.

4.4 Other potential applications

4.4.1 Symbol classification

In this part, we present the application of the proposed approach to handwritten symbol classification. A public database of handwritten musical scores has been used, which mainly contains clefs and accidentals. The database of musical scores is obtained from a collection of modern and old musical scores (19th century) of the Archive of the Seminar of Barcelona. The database contains a total of 4098 samples between seven different types of clefs and accidentals from 24 different authors. The main difficulty of this data set is the lack of a clear class separability because of the variation of writer styles and the absence of a standard notation. A pair of segmented samples for each of the seven classes showing the high variability of clefs and accidental appearance from different authors can be observed in Fig. 4.42.

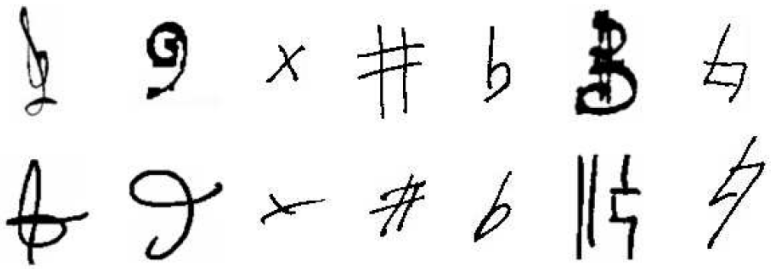


Figure 4.42: Examples of handwritten musical symbols [Escalera 2009]

The objective of this experiment is to evaluate our graph-based approach in terms of classifying the clef and accidental dataset comparing with different state-of-the-art approaches. One of them is an appearance-based approach proposed by S. Escalera et al. [Escalera 2009]. It is based on the Blurred Shape Model (BSM) descriptor (see section 2.2.5), which is invariant to rotation and reflection. Besides the BSM-based approach, we also compare with the approaches using Zoning, SIFT, CSS, and Zernike descriptors. 3-Nearest Neighbor classifier is used for the categorization of symbols. The results are given in Table 4.10.

Approaches	Proposed	BSM	Zernike	Zoning	CSS	SIFT
Accuracy (%)	94.00	73.92	54.12	69.29	61.28	57.39
(confidence interval)	(0.73)	(8.21)	(9.10)	(10.12)	(8.92)	(9.18)

Table 4.10: Classification accuracy and confidence interval on the clefs and accidental categories for the different representations using 3-Nearest Neighbour [Escalera 2009]

Looking at Table 4.10, one can observe that the graph-based approach we pro-

pose achieves much better results than the other state-of-the-art approaches in the comparison. As the only structure-based approach, the outstanding performance of our approach demonstrates the importance of topological information in the symbol classification, especially dealing with handwritten symbols with high variability of appearance.

4.5 Conclusion

We address two important issues concerning word spotting task. First of all, we made our choice of the structure-based approach instead of the popular appearance-based approach. Graph-based representation is adapted in different ways to model handwritten shapes. Secondly, we have developed specific similarity measure dedicated to the graph-based representation with consideration of the specificity of handwriting. Different word spotting system propositions have been introduced in this chapter. In terms of segmented word images, we proposed three methods (fine selection) with a great flexibility in word analysis and comparison. In the scenario of entire pages, the coarse-to-fine scheme is adapted to overcome the constraint of word-segmentation as well as to save the computational time. Two coarse selection approaches have been developed by taking into account of different aspects of handwriting. The evaluation of the propositions has testified that our idea of using a complete description of handwriting is very useful for distinguishing handwritten shapes. The extensional application on musical symbol classification further proves the significance of using the structural information. Additionally, we would like to highlight several key points in our work.

- THE CHOICE OF STRUCTURE-BASED REPRESENTATIONS

  In the domain of graphical pattern recognition, many techniques based on the structural information used for symbol recognition or chemical formula recognition were developed in recent years and achieved a great success. However, we notice that structural representations remain insufficiently exploited in the field of handwriting recognition, because they cannot support rigid comparisons and require adjustments to increase the tolerance of internal variations of handwriting. Our motivation to develop a structural representation based on graphs, rather than on an appearance-based model, is in consideration of the structural nature of words. There exist common characteristics for the character execution, such as the presence of junction points, ending and starting points. The topological signatures of handwriting and its two-dimensional nature are our reasons of considering structural-based representations rather than only one-dimensional scalar ones.

- LEARNING-FREE METHODOLOGY AND ADAPTATION TO CONTEXT

  We were motivated in this work by the intention to produce a complete description of handwriting based on complementary dimensions (outlines, skeleton,

etc) and to manage it through flexible metrics correlated to acceptable approximations. These approximations have been established to compensate the absence of learning. Considering that there is no guarantee of enough priori knowledge, we chose to solve the questions related to the matching of words with adaptations to the context of local connected objects (it can be also seen as an adaptation to the scale of handwriting). On the other hand, some approximations with the DTW and the approximate graph edit distance and the optimization using LDA for the weights in the final distance compensate the absence of learning.

In the end, we summarize word spotting approaches that we propose with the state of the art in the Table. 4.11 in terms of different attributes.

The work concerning word spotting propositions illustrated in this chapter has been published in

- *the 21st International Conference on Image Processing*, 2014
- *the 22nd International Conference on Pattern Recognition*, 2014
- *the 11th IAPR International workshop on Document Analysis System*, 2014
- *Colloque International Francophone sur l'Écrit et le Document*, 2014
- *the 12th International Conference on Document Analysis and Recognition*, 2013

Approach	App.-based	Struct.-based	Seg.-based	Seg.-free	Learning	Learning-free
[Rath 2007]	columnwise features		word-level			X
[Lladós 2007]	Shape Context		word-level			X
[Leydier 2009]	Gradients			X		X
[Rodríguez-Serrano 2009]	Gradients		word-level			X
[Fischer 2010a]	Sequential features		line-level		HMMs	
[Fischer 2010b]		Graph-based	word-level		HMMs	
[Frinken 2010]	Sequential features		line-level		BLSTM NN	
[Rusiñol 2011]	BoVW (SIFT)			X	LSI	
[D. Fernández 2011]		Pseudo-structure (Loci)	word-level			X
[Almazán 2012b]	HOG			X	Exemplar SVM	
[Almazán 2013]	Fischer vector (SIFT)		word-level		CCA	
[Aldavert 2013]	BoVW (SIFT)		word-level		LSA	
[Rothacker 2013]	Bag-of-features (SIFT)			X	HMMs	
[Toselli 2014]		Word graph	line-level		HMMs	
* [Wang 2013]	Textual/Shape Context/ dominant patterns		line-level			X
* [Wang 2014a]		Graph-based	word-level			X
* [Wang 2014b]		Graph-based	line-level			X
* [Wang 2014c]	Shape Context	Graph-based	word-level		LDA	

Table 4.11: Summary of attributes of each handwritten word spotting approach including our contribution (marked by *)

Chapter 5

Conclusion and Perspectives

Contents

5.1 Conclusion 135
5.2 Perspectives 137
5.2.1 Segmentation-free 137
5.2.2 Multi-writing styles 138

5.1 Conclusion

The objective of this research is to establish an effective representation model for the handwriting, especially historical manuscripts. The proposed model should be able to help to address the problem of the query-by-example word spotting within the same writing style, dedicated to Latin languages. In other words, it requires the representation model to be robust to intra-variations of handwriting and to tolerate specific distortions existing in historical document images.

After a long-term study of the state of the art, we discover that there is always some drawback in using single type of features to represent handwriting. A comprehensive model integrating different attributes of handwriting is desirable. Motivated by this, we start to explore a representation model which can effectively combine the characteristics of different aspects of handwriting without redundancy. Existing researches on handwriting recognition and analysis have revealed the importance of the morphology and the topology of handwritten shapes in characterizing handwriting. Therefore, we attempt to develop a hybrid representation model containing both morphological and topological signatures of handwriting. Different proposals have been put forward and evaluated.

First of all, a representation model based on the combination of the Shape Context description, dominant patterns and textural properties is established. For the Shape Context description, structural points are detected and used as the reference points instead of down-sampled contour or skeleton points. Concerning the strength of dominant patterns in distinguishing handwritten characters, loops and ascenders/descenders are identified with several attributes. Moreover, to make the representation more reliable, low-level textural features, i.e. projection profile, upper/lower border profile and orientation distribution, are included. A late fusion strategy with different weights is adopted to integrate these descriptions into an

entirety at the stage of similarity measure. Even though the evaluation on the performance dedicated to the application of word spotting is not outstanding, it still reveals the potential of such a hybrid model with a superior performance compared to the only appearance-based model.

Subsequently, the exploration on representing handwriting extends to the usage of graphs. Contrary to the fashionable appearance-based approaches, graph-based approaches are still rarely used in handwriting recognition and analysis. Graphs, as a well-known model for characterizing the structure of objects, can contain much concrete information with its labeled vertexes and edges. As the first attempt, we use structural points as vertexes, and strokes obtained by segmenting the ink trace with structural points as edges. In order to avoid the expensive computation caused by graph matching, we adapt the bag-of-visual-word method powered by the topological node features (TNFs), a descriptor used for depicting the local configuration of graph from the point of view of a single vertex. In this way, topological signatures are transferred from graphs to a sole one-dimensional vector. Concerning the morphological part, we keep using the Shape Context description following the same process as the first proposition. In the end, the descriptions of two facets of handwriting are summed up as the final representation. Concerning the word spotting application, a weighted similarity measure is designed for comparing segmented word samples. The optimal weight for each part can be automatically learned by linear discriminant analysis. The experiments carried out on two public handwriting datasets show the outstanding performance of the proposed approach.

After the success of using graphs, we continue to work on the graph-based model. A novel graph-based hybrid representation model is developed for handwritten word spotting. The same construction of graphs as proposed in the previous proposition is adapted. In order to generate a compact representation, both vertexes and edges are labeled with morphological information. Based on the proposed representation, a coarse-to-fine word spotting approach is developed. In the coarse selection, graphs of handwriting are converted into a one-dimensional vector with the graph embedding algorithm based on the statistics of vertex labels and the edges between them so as to realize a fast comparison between the query and test images. Since the text lines are not segmented into words, the local Shape Context extracted within the neighborhood of the reference vertex is employed as the label of the vertex in this stage. The length of the stroke is used as the label of edge. In this way, the proposed model preserves the global structural properties of the handwriting, and also includes detailed information on local regions. After the regions of interest are selected, the global Shape Context is computed in the scope of the entire region and replaces the local Shape Context as the label of the vertex. A more sophisticated similarity measure using graph edit distance based on the DTW alignment is applied to the query and the regions of interest in the fine selection. The proposed word spotting method has been compared with five state-of-the-art approaches on two benchmark historical handwritten document datasets. The results of the experiments are promising. They demonstrate the competence of our graph-based approach.

As indicated in chapter 1, one of the objectives of this PhD work is to develop flexible and adaptive methods dedicated to handwritten document retrieval. Therefore, the successive propositions that we make for word spotting can be chosen depending on the dataset as well as the objective of users. They must not be considered as an "incremental" way of improving word spotting but as different propositions depending on requirements.

In addition to word spotting application in Latin language, the extension at the end of chapter 4 has also shown the effectiveness of our representation model and the corresponding similarity measure on other materials with minor adaptations.

5.2 Perspectives

The initiation of our project is to help scholars in literature and humanities to solve the problem of understanding illegible text in historical handwritten documents by offering an efficient indexing tool. Building an entire retrieval system is a big effort that requires making numerous decisions. Due to time constraints, some of our choices in this work are not based on an in-depth investigation of what the optimal alternative would be, but are rather informed "guesses". Moreover, because of the complexity and triviality of the entire approach as well as the numerous challenges the problem poses, various remaining limitations need improvement. Here we propose directions for future work, paying special attention to making our approach more practicable and feasible.

5.2.1 Segmentation-free

One of the shortcomings of our proposed approach is its dependency on segmentation. Although we need no word segmentation, precise text line segmentation is still required. Because of the flexibility existing in the layout of handwriting, it is difficult to segment the text line without any flaw. The best way to handle this problem is to develop a purely segmentation-free approach.

The key factor that hampers the execution of the proposed approach on the entire page without segmentation is not the representation itself. It is in consideration of the computational expense. It will be very time-consuming if we apply our current approach to the entire collection without any coarse selection (segmentation). Since in practice, computational time is of great importance, we are thinking of simplifying our representation model to reduce the complexity while keeping the useful information.

Moreover, if binarization is also considered as a segmentation in some sense, it is also a process that we should avoid, because of the information loss caused by this operation. In our approaches, binarization is mainly used as a preprocessing of the skeletonization. We have discussed its influence on the skeleton and used the tensor voting technique to tackle this problem. However, another possibility to address this issue is to extract the skeleton directly from grey-scale/color image, by using a median axis retrieval approach, for example.

5.2.2 Multi-writing styles

So far, our assumption has been that the analyzed document collection is written by a single writer or has a unique writing style. Although we have not investigated the performance of our system on a collection with many writers, we expect the retrieval performance to decrease, because of the differences in writing style. As stated in the conclusion of the study of the state of the art, it is not possible to realize handwritten word spotting in the scenario of multi-writing styles without a learning process.

To handle inter-variations of handwriting, the key issue concerning our approaches is to find an appropriate machine learning technique for graph matching. Due to the distinctive two-dimensional nature of the graph representation, many favorable classification approaches, which require one-dimensional vectors as input, are technically not feasible for our representation model, e.g. SVM, AdaBoost, K-means and so on. To realize the idea of introducing graph learning into our approach, an in-depth study on the relevant researches needs to be carried out.

Finally, the author hopes that the content of this thesis will be of valuable help to researchers working on handwritten word spotting or similar problems.

Bibliography

[Abe 1985] K. Abe and S. Suzuki. *Topological structural analysis of digital binary images by border following.* Computer Vision, Graphics and Image Processing, vol. 30, no. 2, pages 32–46, 1985. (Cited on pages 47, 48 and 49.)

[Agam 2007] G. Agam, G. Bal, G. Frieder and O. Frieder. *Degraded document image enhancement.* In Proceedings of the SPIE, Document Recognition and Retrieval XIV, volume 6500, pages 65000C–1–65000C–11, 2007. (Cited on page 42.)

[Aida-zade 2009] K. R. Aida-zade and J. Z. Hasanov. *Word base line detection in handwritten text recognition systems.* International Journal of Electrical and Computer Engineering, vol. 4, no. 5, pages 310–314, 2009. (Cited on pages 45 and 46.)

[Aldavert 2013] D. Aldavert, M. Rusiñol, R. Toledo and J. Lladós. *Integrating visual and textual cues for query-by-string word spotting.* In Proceedings of the 12th Int. Conf. on Document Analysis and Recognition, pages 511–515, 2013. (Cited on pages 21, 28, 31, 79, 85, 86, 88, 91, 92 and 134.)

[Almazán 2012a] J. Almazán, D. Fernández, A. Fornés, J. Lladós and E. Valveny. *A coarse-to-fine approach for handwritten word spotting in large scale historical documents collection.* In Proceedings of Int. Conf. on Frontiers in Handwriting Recognition, 2012. (Cited on page 107.)

[Almazán 2012b] J. Almazán, A. Gordo, A. Fornés and E. Valveny. *Efficient exemplar word spotting.* In Proceedings of the British Machine Vision Conference, pages 67.1–67.11, 2012. (Cited on pages 17, 79, 82, 85, 87, 92 and 134.)

[Almazán 2013] J. Almazán, A. Gordo, A. Fornés and E. Valveny. *Handwritten word spotting with corrected attributes.* In Proceedings of the Int. Conf. on Computer Vision, pages 1017–1024, 2013. (Cited on pages 79, 82, 83, 85, 88, 92 and 134.)

[Almazán 2014] J. Almazán, A. Gordo, A. Fornés and E. Valveny. *Segmentation-free word spotting with exemplar SVMs.* 2014. (Cited on pages 21, 28, 87 and 89.)

[Aslan 2008] C. Aslan, A. Erdem, E. Erdem and S. Tari. *Disconnected skeleton: shape at its absolute scale.* IEEE Trans. Pattern Anal. Machine Intell., vol. 30, no. 12, pages 2188–2203, 2008. (Cited on page 46.)

[Bai 2008] X. Bai and L. Latecki. *Path similarity skeleton graph matching.* IEEE Trans. Pattern Anal. Machine Intell., vol. 30, no. 7, pages 1282–1292, 2008. (Cited on page 46.)

[Bai 2009] S. Bai, L. Li and C. L. Tan. *Keyword spotting in document images through word shape coding.* In Proceedings of Document Analysis and Recognition, pages 331–335, 2009. (Cited on pages 23 and 110.)

[Bay 2008] H. Bay, Tuytelaars T and L. V. Gool. *Surf: speeded up robust features.* Journal of Computer Vision and Image Understanding, vol. 10, pages 346–359, 2008. (Cited on pages 22, 59 and 60.)

[Belongie 2002] S. Belongie, J. Malik and J. Puzicha. *Shape matching and object recognition using shape contexts.* IEEE Trans. Pattern Anal. Machine Intell., vol. 24, no. 4, pages 509–522, 2002. (Cited on pages 20, 46, 60, 102 and 109.)

[Benjelil 2009] M. Benjelil, S. Kanoun, R. Mullot and A. M. Alimi. *Arabic and Latin script identification in printed and handwritten types based on steerable pyramid features.* In Proceedings of 10th Int. Conf. on Document Analysis and Recognition, 2009. (Cited on pages 15 and 16.)

[Benjelil 2012] M. Benjelil, R. Mullot and A. M. Alimi. *Language and script identification based on steerable pyramid features.* In Proceedings of Int. Conf. on Frontiers in Handwriting Recognition, 2012. (Cited on page 15.)

[Bensefia 2005] A. Bensefia, T. Paquet and L. Heutte. *A writer identification and verification system.* Journal of Pattern Recognition Letters, vol. 26, no. 13, pages 2080–2092, 2005. (Cited on page 14.)

[Bezdek 1981] J. C. Bezdek. Pattern recognition with fuzzy objective function algoritms. Plenum Press, 1981. (Cited on page 103.)

[Biglari 2014] M. Biglari, F. Mirzaei and J. G. Neycharan. *Persian/Arabic handwritten digit recognition using local binary pattern.* Journal of Digital Information and Wireless Communications, vol. 4, no. 4, 2014. (Cited on page 20.)

[Blankers 2007] V. Blankers, R. Niels and L. Vuurpijl. *Writer identification by means of explainable features: shapes of loop and lead-in strokes.* In Proceedings of the 19th Belgian-Dutch Conference on Artificial Intelligence, pages 17–24, 2007. (Cited on pages 23, 110 and 111.)

[Blum 1967] H. Blum. *A transformation for extracting new descriptors of shape.* Models for the Perception of Speech and Visual Form, vol. 19, pages 362–380, 1967. (Cited on page 50.)

[Bozinovic 1987] R. Bozinovic and S. Srihari. *Multi-level perception approach to reading cursive script.* Artificial Intelligence, vol. 33, no. 1, pages 217–255, 1987. (Cited on page 43.)

[Bronstein 2008] A. M. Bronstein, M. M. Bronstein, A. M. Bruckstein and R. Kimmel. *Analysis of two-dimensional non-rigid shapes.* Int. Journal of Computer Vision, vol. 78, no. 2, pages 67–88, 2008. (Cited on page 46.)

[Bucalu 2005] M. Bucalu and L. Schomaker. *A Comparison of Clustering Methods for Writer Identification and Verification.* In Proceedings of the 3rd Int. Conf. on Document Analysis and Recognition, volume 2, pages 1375–1279, 2005. (Cited on page 27.)

[Bulacu 2007] M. Bulacu and L. Schomaker. *Text-Independent writer identification and verification using textural and allographic features.* vol. 29, no. 4, pages 701–717, 2007. (Cited on pages 11 and 12.)

[Bunke] H. Bunke. Structural pattern recognition (tutorial material). (Cited on page 65.)

[Bunke 2006] H. Bunke, M. Neuhaus and K. Riesen. *Fast suboptimal algorithms for the computation of graph edit distance.* In Joint IAPR international workshops, SSPR and SPR2006. Lecture Notes in Computer Science, volume 4109, pages 163–172, 2006. (Cited on page 74.)

[Calonder 2010] M. Calonder, V. Lepetit, C. Strecha and P. Fua. *BRIEF: binary robust independent elementary features.* In Proceedings of the European Conf. on Computer Vision, volume 6314, pages 778–792, 2010. (Cited on page 60.)

[Çeliktutan 2013] O. Çeliktutan, C. B. Akgul, C. Wolf and B. Sankur. *Graph-Based Analysis of Physical Exercise Actions.* In Proceedings of the ACM Multimedia Workshop on Multimedia Indexing and Information Retrieval for Healthcare, pages 23–32, 2013. (Cited on page 67.)

[Chherawala 2011] Y. Chherawala, R. Wisnovsky and M. Cheriet. *Tsv-lr: Topological signature vector-based lexicon reduction for fast recognition of premodern Arabic subwords.* In Proceedings of the Workshop on Historical Document Imaging Processing, pages 6–13, 2011. (Cited on pages 32, 68 and 83.)

[Costagliola 2011] G. Costagliola, M. de Rosa and V. Fuccella. *Improving shape context matching for the recognition of sketched symbols.* In Proceedings of Int. Conf. Distributed Multimedia Systems, 2011. (Cited on page 20.)

[D. Fernández 2011] J. Lladós D. Fernández and A. Fornés. *Handwritten word spotting in old manuscript images using a pseudo-structural descriptor organized in a hash structure.* Journal of Pattern Recognition and Image Analysis, vol. 6669, pages 628–635, 2011. (Cited on pages 79, 84, 85, 86, 89, 92 and 134.)

[Daher 2010a] H. Daher, V. Eglin, S. Bres and N. Vincent. *New approach for centerline extraction in handwritten strokes: An application to the constitution of a code book.* In Proceedings of the Int. Workshop on Document Analysis System, pages 425–432, 2010. (Cited on page 50.)

[Daher 2010b] H. Daher, D. J. Gaceb, V. Eglin, S. Bres and N. Vincent. *Ancient handwritings decomposition into graphemes and codebook generation based on graph coloring.* In Proceedings of the Int. Workshop on Frontiers in Handwriting Recognition, pages 119–124, 2010. (Cited on pages 26 and 27.)

[Dahm 2013] N. Dahm, H. Bunke, T.Caelli and Y. Gao. *A unified framework for strengthening topological node features and its application to subgraph isomorphism detection.* In Proceedings of the IAPR Workshop on Graph-Based Representations in Pattern Recognition, volume 7877, pages 11–20, 2013. (Cited on pages 102 and 118.)

[Dalal 2005] N. Dalal and B. Triggs. *Histograms of oriented gradients for human detection.* In Proceedings of the Int. Conf. on Computer Vision and Pattern Recognition, volume 1, pages 886–893, 2005. (Cited on pages 16 and 17.)

[Dalal 2006] N. Dalal, B. Triggs and C. Schmid. *Human Detection Using Oriented Histograms of Flow and Appearance.* In Proceedings of the European Conf. on Computer Vision, volume 3952, pages 428–441, 2006. (Cited on page 17.)

[Deerwester 1990] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer and R. Harshman. *Indexing by Latent Semantic Analysis.* Journal of the American Society for Information Science, vol. 41, no. 6, 1990. (Cited on pages 32 and 86.)

[Du 2010] L. Du, X. You, H. Xu and Z. Gao. *Wavelet domain local binary pattern features for writer identification.* In Proceedings of Int. Conf. Pattern Recognition, 2010. (Cited on pages 14 and 20.)

[Dutta 2012] A. Dutta, J. Gibert, J. Lladós, H. Bunke and U. Pal. *Combination of product graph and random walk kernel for symbol spotting in graphical documents.* In Proceedings of the Int. Conf. on Pattern Recognition, pages 1663–1666, 2012. (Cited on pages 67 and 85.)

[Eglin 2007] V. Eglin, S. Bres and C. Rivero. *Hermite and Gabor transforms for noise reduction and handwriting classification in ancient manuscripts.* Journal of Document Analysis and Recognition, vol. 9, pages 101–122, 2007. (Cited on pages 12, 14 and 15.)

[El-Yacoubi 1999] A. El-Yacoubi, R. Sabourin, M. Gilloux and C.Y. Suen. Knowledge-based intelligent techniques in character recognition, chapitre Off-line handwritten word recognition using Hidden Markov Models, pages 191–229. CRC Press, 1999. (Cited on page 31.)

[Escalera 2009] S. Escalera, A. Fornés, O. Pujol, P. Radeva, G. Sánchez and J. Lladós. *Blurred shape model for binary and grey-level symbol recognition.* Journal of Pattern Recognition Letters, vol. 30, no. 15, pages 1424–1433, 2009. (Cited on pages 32, 33 and 131.)

[Felzenszwalb 2007] P. Felzenszwalb and J. Schwartz. *Hierarchical matching of deformable shapes.* In Proceedings of the Int. Conf. on Computer Vision and Pattern Recognition, pages 1–8, 2007. (Cited on page 46.)

[Fernández 2014] D. Fernández, J. Almazán, N. Cirera, A. Fornés and J. Lladós. *BH2M: the Barcelona Historical Handwritten Marriages database.* In Proceedings of Int. Conf. Pattern Recognition, 2014. (Cited on pages 94, 95 and 107.)

[Filatov 1995] A. Filatov, A. Gitis and I. Kil. *Graph-based handwritten digit string recognition.* In Proceedings of the 3rd Int. Conf. on Document Analysis and Recognition, pages 845–849, 1995. (Cited on page 83.)

[Fischer 2010a] A. Fischer, A. Keller, V. Frinken and H. Bunke. *Hmm-based word spotting in handwritten documents using subword models.* In Proceedings of the Int. Conf. on Pattern Recognition, pages 3416–3419, 2010. (Cited on pages 32, 68, 79, 85, 86, 87, 88, 90, 92 and 134.)

[Fischer 2010b] A. Fischer, K. Riesen and H. Bunke. *Graph similarity features for HMM-based handwriting recognition in historical documents.* In Proceedings of Int. Conf. Frontiers in Handwriting Recognition, 2010. (Cited on pages 85, 92 and 134.)

[Fischer 2013] A. Fischer, C. Y. Suen, V. Frinken, K. Riesen and H. Bunke. *A fast matching algorithm for graph-based handwriting recognition.* In Lecture Note in Computer Science, volume 7887, pages 194–203, 2013. (Cited on pages 32, 68, 83, 84 and 85.)

[Frinken 2010] V. Frinken, A.Fischer and H. Bunke. *A novel word spotting algorithm using bidirectional long short-term memory neural networks.* In Proceedings of the 4th IAPR Workshop on Artificial Neural Networks in Pattern Recognition, volume 5998, pages 185–196, 2010. (Cited on pages 85, 87, 89, 90, 91, 92 and 134.)

[Ghiasi 2010] G. Ghiasi and R. Safabakhsh. *An efficient method for offline text independent writer identification.* In Proceedings of the Int. Conf. on Pattern Recognition, pages 1245–1248, 2010. (Cited on pages 25 and 26.)

[Gibert 2012] J. Gibert, E. Valveny and H. Bunke. *Graph embedding in vector spaces by node attribute statistics.* Journal of Pattern Recognition, vol. 45, pages 3072–3083, 2012. (Cited on pages 93, 102 and 104.)

[Gilliam 2011] T. Gilliam, R. C. Wilson and J. A. Clark. *Segmentation and normalization in grapheme codebooks.* In Proceedings of the Int. Conf. on Document Analysis and Recognition, pages 613–617, 2011. (Cited on pages 25 and 27.)

[Grigorescu 2002] S. E. Grigorescu, N. Petkov and P. Kruizinga. *Comparison of texture features based on Gabor filters.* IEEE Trans. Image Processing, vol. 11, no. 10, pages 1160–1167, 2002. (Cited on page 14.)

[Gupta 2009] A. Gupta, A. Kembhavi and L. S. Davis. *Observing human-object interactions: Using spatial and functional compatibility for recognition.* IEEE Trans. Pattern Anal. Machine Intell., vol. 31, no. 10, pages 1775–1789, 2009. (Cited on page 67.)

[Hero 2006] A. Hero and D. Justice. *A binary linear programming formulation of the graph edit distance.* IEEE Trans. Pattern Anal. Machine Intell., vol. 28, pages 1200–1214, 2006. (Cited on page 74.)

[Hsieh 1995] A. J. Hsieh, K. G. Fan and T. I. Fan. *Bipartite weighted matching for on-line handwritten Chinese character recognition.* Journal of Pattern Recognition, vol. 28, no. 2, pages 422–446, 1995. (Cited on page 83.)

[Hu 1997] J. Hu, A. S. Rosenthal and M. K. Brown. *Combining high-level features with sequential local features for online handwriting recognition.* In Proceedings of the Int. Conf. on Image Analysis and Processing, pages 647–654, 1997. (Cited on page 10.)

[Hu 2010] R. Hu, M. Barnard and J. Collomosse. *Gradient field descriptor for sketch based retrieval and localization.* In Proceedings of the 17th Int. Conf. on Image Processing, pages 1025–1028, 2010. (Cited on page 17.)

[Jager 1996] S. Jager. *Recovering writing traces in off-line handwriting recognition: using a global optimization technique.* In Proceedings of the 13th Int. Conf. on Pattern Recognition, volume 2, pages 150–154, 1996. (Cited on page 37.)

[Jain 1988] A. Jain. Fundamentals of digital image processing. Prentice Hall, 1988. (Cited on page 40.)

[Journet 2008] N. Journet, J. Y. Ramel, R. Mullot and V. Eglin. *Document image characterization using a multiresolution analysis of the texture: application to old documents.* Journal of Document Analysis and Recognition, vol. 11, pages 9–18, 2008. (Cited on pages 12 and 98.)

[Joutel 2008] G. Joutel, V. Eglin and H. Emptoz. *Generic scale-space process for handwriting documents analysis.* In Proceedings of Int. Conf. on Image and Signal Processing, 2008. (Cited on page 15.)

[Kan 2002] C. Kan and M. D. Srinath. *Invariant character recognition with Zernike and orthogonal Fourier-Mellin moments.* Journal of Pattern Recognition, vol. 35, 2002. (Cited on page 16.)

[Katz 2003] R. Katz and S. Pizer. *Untangling the blum medial axis transform.* Int. Journal of Computer Vision, vol. 55, no. 2, pages 139–153, 2003. (Cited on page 46.)

[Kégl 1999] B. Kégl and A. Krzyzak. *Piecewise linear skeletonization using principal curves.* IEEE Trans. Pattern Anal. Machine Intell., vol. 24, no. 1, pages 59–74, 1999. (Cited on page 37.)

[Kessentini 2010] Y. Kessentini, T. Paquet and A. B. Hamadou. *Off-line handwritten word recognition using multi-stream hidden Markov models.* Journal of Pattern Recognition Letters, vol. 31, no. 1, pages 60–70, 2010. (Cited on pages 85 and 86.)

[Kim 1996] G. Kim and V. Govindaraju. *Efficient chain code based image manipulation for handwritten word recognition.* In Proceedings of SPIE Symposium on Electronic Imaging Science and Technology, 1996. (Cited on page 29.)

[Kimura 1993] F. Kimura, M. Shridhar and N. Narasimhamurthi. *Lexucin driven segmentation - recognition procedure for unconstrained handwritten words.* In Proceedings of Int. Workshop on Frontiers in Handwriting Recognition, 1993. (Cited on page 29.)

[Koerich 2006] A. L. Koerich, A. S. Britto Jr., L. E. S. Oliveira and R. Sabourin. *Fusing high- and low-level features for handwritten word recognition.* In Proceedings of the 10th Int. Workshop on Frontiers in Handwriting Recognition, pages 151–156, 2006. (Cited on page 9.)

[Latecki 2000] L. Latecki and R. Lakämper. *Shape similarity measure based on correspondence of visual parts.* IEEE Trans. Pattern Anal. Machine Intell., vol. 22, no. 10, pages 1185–1190, 2000. (Cited on page 46.)

[Lavrenko 2004a] V. Lavrenko, T. Rath and R. Manmatha. *Holistic word recognition for handwritten historical documents.* In Proceedings of Document Image Analysis for Libraries, pages 278–287, 2004. (Cited on pages 23 and 110.)

[Lavrenko 2004b] V. Lavrenko, T. Rath and R. Manmatha. *Holistic word recognition for handwritten historical documents.* In Proceedings of Document Image Analysis for Libraries, pages 278–287, 2004. (Cited on page 37.)

[Lebourgeois 2007] F. Lebourgeois and H. Emptoz. *Skeletonization by gradient regularization and diffusion.* In Proceedings of the 9th Int. Conf. on Document Analysis and Recognition, pages 1118–1122, 2007. (Cited on pages 50 and 53.)

[Lecolinet 1995] E. Lecolinet and R.G. Casey. *Strategies in character segmentation: a survey.* In Proceedings of the 3rd Int. Conf. on Document Analysis and Recognition, pages 1028–1033, 1995. (Cited on pages 24 and 43.)

[Leydier 2007] Y. Leydier, F. LeBourgeois and H. Emptoz. *Text Search for Medieval Manuscript Images.* Journal of Pattern Recognition, vol. 40, no. 12, pages 3552–3567, 2007. (Cited on pages 17, 81 and 91.)

[Leydier 2009] Y. Leydier, A. Ouji, F. LeBourgeois and H. Emptoz. *Towards an omnilingual word retrieval system for ancient manuscripts.* Journal of Pattern Recognition, vol. 42, no. 9, pages 2089–2105, 2009. (Cited on pages 17, 79, 81, 86, 92, 127, 128 and 134.)

[Ling 2007] H. Ling and D. Jacobs. *Shape classification using the inner-distance.* IEEE Trans. Pattern Anal. Machine Intell., vol. 29, no. 2, pages 286–299, 2007. (Cited on page 46.)

[Lladós 2001] J. Lladós, E. Martí and J. J. Villanueva. *Symbol recognition by error-tolerant subgraph matching between region adjacency graphs.* IEEE Trans. Pattern Anal. Machine Intell., vol. 23, pages 1137–1143, 2001. (Cited on page 65.)

[Lladós 2007] J. Lladós, P. P. Roy and G. Sánchez J. A. Rodríguez. *Word Spotting in Archive Documents Using Shape Contexts.* In Proceedings of Iberian Conference on Pattern Recognition and Image Analysis, volume 2, pages 290–297, 2007. (Cited on pages 20, 86, 92, 109 and 134.)

[Lladós 2012] J. Lladós, M. Rusiñol, A. Fornés, D. Fernández and A. Dutta. *On the influence of word representations for handwritten word spotting in historical documents.* Journal of Pattern Recognition and Artificial Intelligence, vol. 26, no. 5, pages 1 263 002.1–1 263 002.25, 2012. (Cited on pages 32, 68, 79, 84, 85, 86, 127 and 128.)

[López 1998] D. López and J. M. Sempere. *Handwritten digit recognition through inferring graph grammars.* In Proceedings of Joint IAPR Int. Workshops on Advances in Pattern Recognition, pages 483–491, 1998. (Cited on page 83.)

[Lowe 1999] D. G. Lowe. *Object recognition from local scale-invariant features.* In Proceedings of the 7th Int. Conf. on Computer Vision, pages 1150–1157, 1999. (Cited on pages 18 and 21.)

[Lowe 2004] D. G. Lowe. *Distinctive image features from scale-invariant keypoints.* Int. Journal of Computer Vision, vol. 60, pages 91–110, 2004. (Cited on pages 19, 22 and 60.)

[Lu 1991] S. Lu, Y. Ren and C. Suen. *Hierarchical attributed graph representation and recognition of handwritten Chinese characters.* Journal of Pattern Recognition, vol. 24, no. 7, pages 617–632, 1991. (Cited on pages 32, 68 and 83.)

[Luqman 2011] M. Luqman, J. Ramel, J. Lladós and T. Brouard. *Subgraph spotting through explicit graph embedding: An application to content spotting in graphic document images.* In Proceedings of the 11th Int. Conf. on Document Analysis and Recognition, pages 870–874, 2011. (Cited on page 67.)

[Maarse 1983] F. Maarse and A. Thomassen. *Produced and perceived writing slant: differences between up and down strokes.* Journal of Acta Psychologica, vol. 54, no. 1-3, pages 131–147, 1983. (Cited on page 11.)

[Mahé 2005] P. Mahé, N. Ueda, T. Akutsu, J. Perret and J. Vert. *Graph kernels for molecular structure-activity relationship analysis with support vector machines.* Journal of Chemical Information and Modeling, vol. 45, no. 4, pages 939–951, 2005. (Cited on page 67.)

[Malisiewicz 2011] T. Malisiewicz, A. Gupta and A. Efros. *Ensemble of exemplar-SVMs for object detection and beyond.* In Proceedings of Int. Conf. Computer Vision, 2011. (Cited on page 87.)

[Malleron 2009] V. Malleron, V. Eglin, H. Emptoz, S. Dord-CrouslÃ© and P. Régnier. *Hierarchical decomposition of handwritten manuscripts layouts.* In Proceedings of the 13th International Conference on Computer Analysis of Images and Patterns, pages 221–228, 2009. (Cited on page 90.)

[Marti 2001] U. Marti and H. Bunke. *Using a statistical language model to improve the performance of an HMM-based cursive handwriting recognition system.* Journal of Pattern Recognition and Artificial Intelligence, vol. 15, pages 65–90, 2001. (Cited on pages 11 and 13.)

[Marti 2002] U. Marti and H. Bunke. *The IAM-database: an English sentence database for off-line handwriting recognition.* Journal on Document Analysis and Recognition, vol. 5, pages 39–46, 2002. (Cited on page 94.)

[Medioni 2000] G. Medioni, C. K. Tang and M. S. Lee. *Tensor voting: theory and applications.* In Proceedings of the Int. Conf. RIFA, 2000. (Cited on page 54.)

[Medioni 2002] G. Medioni. *Tensor voting*, 2002. (Cited on pages 54 and 56.)

[Messmer 1996] B. Messmer and H. Bunke. *Automatic learning and recognition of graphical symbols in engineering drawings.* Journal of Graphics Recognition Methods and Applications, vol. 1072, pages 123–134, 1996. (Cited on page 65.)

[Mikolajczyk 2002] K. Mikolajczyk and C. Schmid. *An affine invariant interest point detector.* In Proceedings of European Conf. on Computer Vision, volume 2350, 2002. (Cited on page 18.)

[Mishra] B. K. Mishra. *Molecular (graph) characteristics of some hydrocarbons through graph theory.* Rapport technique. (Cited on page 67.)

[Moulin 2014] C. Moulin, C. Largeron, C. Ducottet, M. Géry and C. Barat. *Fisher linear discriminant analysis for text-image combination in multimedia information retrieval.* Journal of Pattern Recognition, vol. 47, no. 1, pages 260–269, 2014. (Cited on page 120.)

[Munkres 1957] J. Munkres. *Algorithms for the assignment and transportation problems.* Journal of the Society for Industrial and Applied Mathematics, vol. 5, pages 32–38, 1957. (Cited on pages 75 and 125.)

[Myers 1995] G. K. Myers and P. G. Mulgaonkar. *Automatic extraction of information from printed documents.* In Proceedings of 18th Annual ACM SIGIR, pages 328–335, 1995. (Cited on pages 23 and 110.)

[Nguyen 2008] T-O. Nguyen, S. Tabbone and O. R. Terrades. *Symbol descriptor based on shape context and vector model of information retrieval.* In Proceedings of Int. Workshop Document Analysis Systems, 2008. (Cited on page 20.)

[Nicolaou 2013] A. Nicolaou, M. Liwicki and R. Ingold. *Oriented local binary patterns for writer identification.* In Proceedings of the 2nd ICDAR Workshop on Automated Forensic Handwriting Analysis, 2013. (Cited on page 20.)

[Nicolaou 2014] A. Nicolaou, F. Slimane, V. Märgner and M. Liwicki. *Local binary patterns for arabic optical font recognition.* In Proceedings of 11th Int. Workshop Document Analysis System, 2014. (Cited on pages 20 and 21.)

[Niels 2007] R. Niels, L.Vuurpijl and L. Schomaker. *Automatic allograph matching in forensic writer identification.* Journal of Pattern Recognition and Artificial Intelligence, vol. 21, no. 1, 2007. (Cited on page 22.)

[Otsu 1979] N. Otsu. *A threshold selection method from gray-level histograms.* IEEE Trans. on Systems, Man, and Cybernetics, vol. 9, no. 1, pages 62–66, 1979. (Cited on pages 40 and 97.)

[Pelleg 2000] D. Pelleg and A. Moore. *X-means: extending K-means with efficient estimation of the number of clusters.* In Proceedings of the 17th Int. Conf. on Machine Learning, pages 727–734, 2000. (Cited on page 120.)

[Pervouchine 2005] V. Pervouchine, G. Leedham and K. Melikhov. *Three-stage handwriting stroke extraction method with hidden loop recovery.* In Proceedings of the 8th Int. Conf. on Document Analysis and Recognition, volume 1, pages 307–311, 2005. (Cited on page 37.)

[Pourasad 2012] Y. Pourasad, H. Hassibi and A. Ghorbani. *A Farsi/Arabic word spotting approach for printed document images.* Journal of Natural and Engineering Sciences, vol. 6, no. 1, pages 15–18, 2012. (Cited on pages 23 and 110.)

[Puzicha 2000] J. Puzicha, S. Belongie and J. Malik. *Shape context a new descriptor for shape matching and object recognition.* Journal of NIPs, pages 831–837, 2000. (Cited on page 60.)

[Raja 2011] K. Raja, I. Laptevy, P. Perez and L. Oisel. *Joint pose estimation and action recognition in image graphs.* In Proceedings of the Int. Conf. on Image Processing, pages 25–28, 2011. (Cited on page 67.)

[Raphael 1968] B. Raphael, P. Hart and N. Nilsson. *A formal basis for the heuristic determination of minimum cost path.* IEEE Trans. Systems, Science, and Cybernetics, vol. 4, pages 100–107, 1968. (Cited on pages 74 and 75.)

[Rath 2005] Toni Maximilian Rath. *Retrieval of handwritten historical document images.* PhD thesis, University of Massachusetts Amherst, 2005. (Cited on page 6.)

[Rath 2007] T. Rath and R. Manmatha. *Word spotting for historical documents.* In Proceedings of the 9th Int. Conf. on Document Analysis and Recognition, volume 9, pages 139–152, 2007. (Cited on pages 37, 79, 80, 85, 86, 89, 90, 92, 94, 96, 99, 127, 128 and 134.)

[Riesen 2009] K. Riesen and H.Bunke. *Approximate graph edit distance computation by means of bipartite graph matching.* Journal of Image and Vision Computing, vol. 27, no. 7, pages 950–959, 2009. (Cited on page 75.)

[Rodríguez-Serrano 2009] J. Rodríguez-Serrano and F. Perronnin. *Handwritten word spotting using hidden markov models and universal vocabularies.* Journal of Pattern Recognition, vol. 42, no. 9, pages 2106–2116, 2009. (Cited on pages 79, 81, 86, 92 and 134.)

[Rodríguez 2007] J. A. Rodríguez, G. Sánchez, J. Lladós and P. Pratim-Roy. *Word spotting in archive documents using shape context.* Journal of Pattern Recognition and Image Analysis, vol. 4478, no. 3, pages 290–297, 2007. (Cited on pages 17, 60 and 81.)

[Romero 2007] D. J. Romero, L. M. Seijas and A. M. Ruedin. *Directional continuous wavelet transform applied to handwritten numerals recognition using neural networks.* Journal of Computer Science and Technology, vol. 7, no. 1, 2007. (Cited on page 14.)

[Rothacker 2013] L. Rothacker, M. Rusiñol and G. A. Fink. *Bag-of-Features HMMs for segmentation-free word spotting in handwritten documents.* In Proceedings of 12th Int. Conf. Document Analysis and Recognition, 2013. (Cited on pages 21, 31, 79, 85, 86, 91, 92 and 134.)

[Rousseau 2006] L. Rousseau, J. Camillerapp and E. Anquetil. *What knowledge about handwritten letters can be used to recover their drawing order?* In Proceedings of the 10th Int. Workshop on Frontiers in Handwriting Recognition, 2006. (Cited on page 22.)

[Rusiñol 2011] M. Rusiñol, D. Aldavert, R. Toledo and J. Lladós. *Browsing heterogeneous document collections by a segmentation-free word spotting method.*

In Proceedings of the 11th Int. Conf. on Document Analysis and Recognition, pages 63–67, 2011. (Cited on pages 21, 31, 65, 79, 81, 82, 85, 86, 91, 92 and 134.)

[Sarkar 2013] S. Sarkar. *Word spotting in cursive handwritten documents using modified character shape codes.* In Proceedings of the Int. Conf. on Advances in Computing and Information Technology, volume 178, pages 269–278, 2013. (Cited on pages 23 and 110.)

[Schomaker 2004] L. Schomaker and M. Bulacu. *Automatic writer identification using fragmented connected-compenent contours.* In Proceedings of the 9th Int. Workshop on Frontiers in Handwriting Recognition, pages 185–190, 2004. (Cited on page 25.)

[Schomaker 2007] L. Schomaker, K. Franke and M. Bulacu. *Using codebooks of fragmented connected-component contours in forensic and historic writer identification.* Pattern Recognition Letters, vol. 28, no. 6, pages 719–727, 2007. (Cited on page 25.)

[Sebastian 2004] T. Sebastian, P. Klein and B. Kimia. *Recognition of shapes by editing their shock graphs.* IEEE Trans. Pattern Anal. Machine Intell., vol. 26, no. 5, pages 550–571, 2004. (Cited on page 46.)

[Serre 2007] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber and T. Poggio. *Robust object recognition with cortex-like mechanisms.* IEEE Trans. Pattern Anal. Machine Intell., vol. 29, no. 3, pages 411–426, 2007. (Cited on page 33.)

[Shahabi 2009] F. Shahabi and M. Rahmati. *A new method for writer identification of handwritten Farsi documents.* In Proceedings of 10th Int. Conf. on Document Analysis and Recognition, 2009. (Cited on page 14.)

[Shapiro 2002] L. Shapiro and G. Stockman. Computer vision. Prentice Hall, 2002. (Cited on page 71.)

[Shi 2004] Z. Shi and V. Govindaraju. *Historical document image enhancement using background light intensity normalization.* In Proceedings of the 17th Int. Conf. on Pattern Recognition, volume 1, pages 473–476, 2004. (Cited on page 42.)

[Shimoni 1998] L. Shimoni. *Molecular recognition in selected biological systems.* PhD thesis, 1998. (Cited on page 67.)

[Siddiqi 1999] K. Siddiqi and A. Shokoufandeh. *Shock graphs and shape matching.* Int. Journal of Computer Vision, vol. 35, no. 1, pages 13–32, 1999. (Cited on page 46.)

[Siddiqi 2009] Imran Siddiqi. *Classification of handwritten documents : writer recognition.* PhD thesis, Université Paris Descartes, 2009. (Cited on page 30.)

[Simoncelli 1995] E. P. Simoncelli and W. T. Freeman. *The steerable pyramid: a flexible architecture for multi-scale derivative computation.* In Proceedings of 2nd Int. Conf. on Image Process, 1995. (Cited on page 15.)

[Slavik 2000] P. Slavik and V. Govindaraju. *An Overview of Run-length Encoding of Handwritten Word Images.* Rapport technique, CEDAR SUNY Buffalo, 2000. (Cited on pages 29 and 30.)

[Spitz 1997] A. L. Spitz. *Determination of the script and language content of document images.* IEEE Trans. Pattern Anal. Machine Intell., pages 235–245, 1997. (Cited on pages 23, 31 and 110.)

[Srihari 2005] S. N. Srihari, H. Srinivasan, P. Bbu and C. Bhole. *Handwritten Arabic word spotting using the CEDARABIC document analysis system.* In Proceedings of the Symposium on Document Image Understanding Technology, pages 123–132, 2005. (Cited on pages 35 and 90.)

[Steinherz 2009] T. Steinherz, D. Doermann, E. Rivlin and N. Intrator. *Offline loop investigation for handwriting analysis.* IEEE Trans. Pattern Anal. Machine Intell., vol. 31, no. 2, pages 193–209, 2009. (Cited on pages 23, 110 and 111.)

[Su 2011] F. Su, T. Lu and R. Yang. *Symbol recognition by multiresolution shape context matching.* In Proceedings of the Int. Conf. on Document Analysis and Recognition, pages 1319–1323, 2011. (Cited on pages 20 and 60.)

[Suard 2006] F. Suard, A. Rakotomamonjy, A. Bensrhair and A. Broggi. *Pedestrian Detection using Infrared images and Histograms of Oriented Gradients.* In Proceedings of the Intelligent Vehicles Symposium, pages 206–212, 2006. (Cited on page 17.)

[Suganthan 1998] P. N. Suganthan and H. Yan. *Recognition of handprinted Chinese characters by constrained graph matching.* Journal of Image and Vision Computing, vol. 16, no. 3, pages 191–201, 1998. (Cited on page 83.)

[Tahmasbi 2012] A. Tahmasbi. *Zernike moments*, 2012. (Cited on page 17.)

[Tan 2008] C. L. Tan, S. Lu and L. Li. *Document image retrieval through word shape coding.* IEEE Trans. Pattern Anal. Machine Intell., vol. 30, no. 11, pages 1913–1918, 2008. (Cited on page 47.)

[Teague 1980] M. R. Teague. *Image analysis via the general theory of moments.* Journal of Optical Society of America, vol. 70, no. 8, pages 920–930, 1980. (Cited on page 16.)

[Toselli 2014] A. H. Toselli and E. Vidal. *Word-graph based handwriting key-word spotting: impact of word-graph size on performance.* In Proceedings of the 11th IAPR Int. Workshop on Document Analysis Systems, pages 176–180, 2014. (Cited on pages 79, 85, 86, 88, 92 and 134.)

[van der Zant 2008] T. van der Zant, L. Schomaker and K. Haak. *Handwritten-word spotting using biologically inspired features.* IEEE Trans. Pattern Anal. Machine Intell., vol. 30, no. 11, pages 1945–1957, 2008. (Cited on pages 33, 34 and 35.)

[Wang 1990] L. Wang and DC. He. *Texture classification using texture spectrum.* Journal of Pattern Recognition, vol. 23, no. 8, 1990. (Cited on page 20.)

[Wang 2001] Q. Wang and C. L. Tan. *Matching of double-sided document images to remove interference.* In Proceedings of the Int. Conf. on Computer Vision and Pattern Recognition, volume 1, pages 1084–1089, 2001. (Cited on page 42.)

[Wang 2013] P. Wang, V. Eglin, C. Largeron, A. McKenna and C. Garcia. *A comprehensive representation model for handwriting dedicated to word spotting.* In Proceedings of the 12th Int. Conf. on Document Analysis and Recognition, pages 450–454, 2013. (Cited on page 134.)

[Wang 2014a] P. Wang, V. Eglin, C. Garcia, C. Largeron, J. Lladós and A. Fornés. *A novel learning-free word spotting approach based on graph representation.* In Proceedings of the 11th IAPR workshop on Document Analysis System, pages 207–211, 2014. (Cited on pages 125 and 134.)

[Wang 2014b] P. Wang, V. Eglin, C. Largeron and C. Garcia. *A coarse-to-fine word spotting approach for historical handwritten documents based on graph embedding and graph edit distance.* In Proceedings of Int. Conf. Pattern Recognition, 2014. (Cited on page 134.)

[Wang 2014c] P. Wang, V. Eglin, C. Largeron and C. Garcia. *Handwritten word spotting based on a hybrid optimal distance (accepted).* In Proceedings of the Int. Conf. on Image Processing, 2014. (Cited on page 134.)

[Yao 2012] B. Yao and F. Li. *Action recognition with exemplar based 2.5D graph matching.* In Proceedings of the European Conf. on Computer Vision, volume 7575, pages 173–186, 2012. (Cited on page 67.)

[Zaslavskiy 2009] M. Zaslavskiy, F.Bach and J.-P. Vert. *A path following algorithm for the graph matching problem.* IEEE Trans. Pattern Anal. Machine Intell., vol. 31, pages 2227–2242, 2009. (Cited on pages 32, 68 and 83.)

[Zhang 1984] T. Y. Zhang and C. Y. Suen. *A fast parallel algorithm for thinning digital patterns.* Communication of the ACM, vol. 27, pages 236–239, 1984. (Cited on page 51.)

[Zhang 2009] Z. Zhang, L. Jin, K. Ding and X. Gao. *Character-SIFT: A novel feature for offline handwritten chinese character recognition.* In Proceedings of Int. Conf. Document Analysis and Recognition, 2009. (Cited on page 21.)

[Zhao 1991] D. Zhao and D. G. Daut. *Morphological hit-or-miss transformation for shape recognition.* Journal of Visual Communication and Image Representation, vol. 2, no. 3, pages 230–243, 1991. (Cited on page 57.)

[Zhu 1996] S. C. Zhu and A. L. Yuille. *Forms: A flexible object recognition and modeling system.* Int. Journal of Computer Vision, vol. 20, no. 3, pages 1573–1405, 1996. (Cited on page 46.)

[Zhu 2009] G. Zhu, Y. Zheng, D. Doermann and S. Jaeger. *Signature detection and matching for document image retrieval.* IEEE Trans. Pattern Anal. Machine Intell., vol. 31, no. 11, pages 2015–2031, 2009. (Cited on page 120.)

[Zimmermann 2004] M. Zimmermann and H. Bunke. *N-Gram language models for offline handwritten text recognition.* In Proceedings of 9th Int. Workshop on Frontiers in Handwriting Recognition, 2004. (Cited on page 28.)