Universitat Autònoma de Barcelona

Multi-Oriented and Multi-Scaled Text Character Analysis and Recognition in Graphical Documents and Their Applications to Document Image Retrieval

A dissertation submitted by **Partha Pratim Roy** at Universitat Autònoma de Barcelona to ful l the degree of **Doctor en Informatica**.

Bellaterra, September 2010

Director: **Dr. Josep Lladós Canet**
Universitat Autònoma de Barcelona
Dep. Ciències de la Computació & Centre de Visió per Computador
Co-director: **Dr. Umapada Pal**
Indian Statistical Institute
Computer Vision and Pattern Recognition Unit

This document was typeset by the author using LaTeX 2$_\varepsilon$.

The research described in this book was carried out at the Computer Vision Center, Universitat Autònoma de Barcelona.

ISBN: 978–84–937261–7–1

Printed by Ediciones Gráficas Rey, S.L.

This thesis is dedicated to my family

Acknowledgements

It is a pleasure to thank people who have helped and inspired me during my doctoral study.

First of all, I would like to express my deep and sincere gratitude to my supervisors Dr. Josep Lladós and Dr. Umapada Pal. Their understanding, encouraging and personal guidance have provided fundamental basis for the present thesis. Their support from the preliminary to the concluding level enabled me to develop an understanding of the subject.

I am indebted to my colleagues for providing a stimulating and fun environment in which to learn and grow. I sincerely appreciate to Ernest, Gemma, Dimos, Mathieu, Agnes, Joan, Miquel, Oriol, Marçal, Jose, Alicia, Jaume, Albert, Farshad, Antonio, Henry, Ricard, Anjan, David, Jon, Lluis for their ideas and brainstorming to this thesis work. I would like to thank Sebastien Adam for providing his code for our research.

Many others have helped me stay concentrated through these difficult years. I would thank to Ivan, Jose Manuel, Bhashkar, Naveen, Marco, Ariel, Murad and others for their valuable support and care which helped me understanding overcome setbacks and stay focused on my graduate study.

I am grateful to the administrative personnel: Pilar, Ana, Helena, Gigi, Ainhoa, Raquel, Mireia, Claire, Eva for assisting me always. Montse deserves special mention. I am also thankful to the system personnel Raquel, Joan, and others.

Lastly, and most importantly, I wish to thank my parents for their support. Without their blessing, this work could not have been accomplished. To them I dedicate this thesis.

Abstract

With the advent research of Document Image Analysis and Recognition (DIAR), an important line of research is explored on indexing and retrieval of graphics rich documents. It aims at finding relevant documents relying on segmentation and recognition of text and graphics components underlying in non-standard layout where commercial OCRs can not be applied due to complexity. This thesis is focused towards text information extraction approaches in graphical documents and retrieval of such documents using text information.

Automatic text recognition in graphical documents (map, engineering drawing, etc.) involves many challenges because text characters are usually printed in multi-oriented and multi-scale way along with different graphical objects. Text characters are used to annotate the graphical curve lines and hence, many times they follow curvi-linear paths too. For OCR of such documents, individual text lines and their corresponding words/characters need to be extracted.

For recognition of multi-font, multi-scale and multi-oriented characters, we have proposed a feature descriptor for character shape using angular information from contour pixels to take care of the invariance nature. To improve the efficiency of OCR, an approach towards the segmentation of multi-oriented touching strings into individual characters is also discussed. Convex hull based background information is used to segment a touching string into possible primitive segments and later these primitive segments are merged to get optimum segmentation using dynamic programming. To overcome the touching/overlapping problem of text with graphical lines, a character spotting approach using SIFT and skeleton information is included. Afterwards, we propose a novel method to extract individual curvi-linear text lines using the foreground and background information of the characters of the text and a water reservoir concept is used to utilize the background information.

We have also formulated the methodologies for graphical document retrieval applications using query words and seals. The retrieval approaches are performed using recognition results of individual components in the document. Given a query text, the system extracts positional knowledge from the query word and uses the same to generate hypothetical locations in the document. Indexing of documents is also performed based on automatic detection of seals from documents containing cluttered background. A seal is characterized by scale and rotation invariant spatial feature descriptors computed from labelled text characters and a concept based on the Gen-

eralized Hough Transform is used to locate the seal in documents.

Keywords: Document Image Analysis, Graphics Recognition, Dynamic Programming, Generalized Hough Transform, Character Recognition, Touching Character Segmentation, Text/Graphics Separation, Curve-Line Separation, Word Retrieval, Seal Detection and Recognition.

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

In this chapter we put in context the analysis and retrieval of graphical document problems. We start by discussing a general overview of document image analysis and recognition and then we will move to graphics recognition, the more particular domain of this research work. Next, we will briefly talk about some related work that are proposed in the literature towards this direction. Afterwards, the problem statement of this thesis is introduced. Finally, we summarize the objectives and contributions of this work.

1.1 Context

As electronic media becomes more and more prevalent, the need for transferring older documents to the electronic domain grows. Recently, many techniques are available to manage such huge volume of electronic documents and to extract knowledge information from such documents. **Document Image Analysis and Recognition** (DIAR) [BW97, Nag00] is the process that performs the overall interpretation of document images. DIAR is concerned with the global issues involved in understanding of printed/written language in images. It refers to algorithms and methods that are applied to document images to obtain computer readable descriptions. DIAR analyzes different text portions in document layout, understands the relationships among these text blocks and finally converts them to a machine-readable version. It adds to **Optical Character Recognition** (OCR) [Man86] technique a superstructure that establishes the organization of the document and applies outside knowledge in interpreting it. OCR works by scanning source documents and performing character analysis on the resulting images, giving a translation to ASCII text, which can then be stored and manipulated electronically like any standard electronic document.

Nowadays, the technology related to DIAR is used in a broad range of applications where some information has to be extracted from documents existing in different media. Typical applications include, among the others, handwritten character recognition, processing of textual web images, and information extraction from digital libraries. In the **Digital Library** (DL) [Bre02] community a lot of efforts have been

devoted to the digitization of paper collections in order to achieve them as document image collections. Libraries and archives are generally interested in mass-digitization and transcription of their book collections. The objective is not only preservation in a digital format of documents of high value, but also to provide access and retrieval services to wider users, and assist scholarly research activities. The simple application of OCR softwares can only partially solve these problems, both for the difficulty of obtaining clean converted text and for the lack of structural description of the document. To tackle these problems, one of the alternative approach usually taken by researchers is **Document Image Retrieval** (DIR) [Doe98]. The objective of DIR is to find relevant document images from an archive based on image features of user's query. In the following we will discuss in brief the OCR structure and document retrieval process.

1.1.1 Optical Character Recognition

An OCR system can be composed into several parts as shown in Fig.1.1. Here, the document is first scanned and digitized. The noise is removed and the text is separated from non-text portions. In **character segmentation** stage, the characters from string is divided such that each image contains a single character [Lu95]. In this stage, the bounding box for each character image is identified. Touching characters lead to false bounding box detection, and hence false recognition results occur. The isolated character images are sent to the preprocessing stage according to the image quality. The character image is usually normalized to a fixed size with a linear or nonlinear normalization method [YYH93].

The feature selection step and the classifier are the two most important parts in the recognition system [TJT96]. In **feature extraction** [ZL04] step, features are computed from character images. In general, feature descriptor is a vector that is produced to describe a given shape of symbol (character). A suitable feature vector should be compact, complete i.e. general enough to represent symbols from different domains, discriminant i.e. able to maximize the intra-class similarity and the inter-class dissimilarity, computationally manageable, extensible and able to support distortion models [LVSM02].

Finally, **character classification** [LS09] assigns each character image with a character identity. A typical character classifier partitions the feature space into regions associating with each class, labeling an observed pattern according to the class region into which it falls. Hence, for each character class either a prototype or a set of character samples must be known.

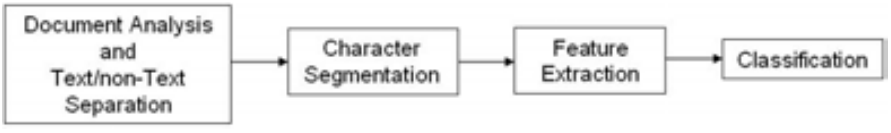


Figure 1.1: Flow chart of general optical character recognition systems.

1.1.2 Document Retrieval

Traditionally, document retrieval process is associated to textual queries. In typewritten and structured documents, the records are indexed with the traditional high performing commercial OCR products. Once the image document is OCRed, the resulting ASCII information is compared with the query using a string search algorithm. In Fig.1.2, a typical document retrieval system is sketched. The digitized document images are first passed through an OCR and the ASCII information corresponding to each document is stored in a database. When a user search a *query* text in this database, the text information goes through a process of text matching and indexing. After matching, the relevant documents appear to the user according to the ranking.

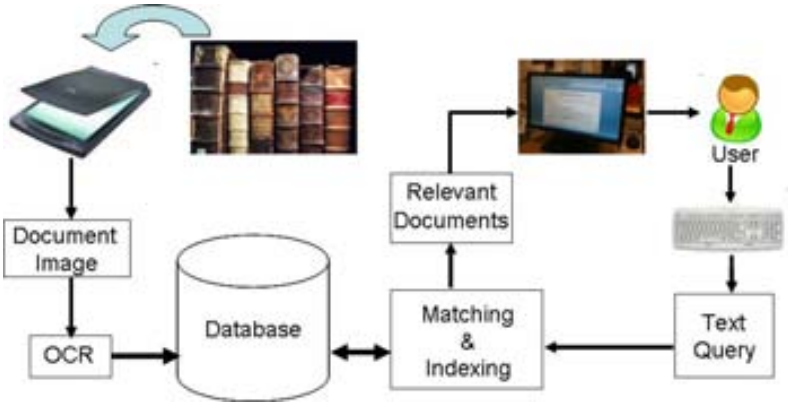


Figure 1.2: Flow chart of a typical document retrieval system.

In the last few years, intensive research has been performed on **Content-based Image Retrieval** (CBIR) [FSN+95] and consequently a wide range of techniques have been proposed. CBIR consists of retrieving visually similar images from an image database based on a given query image. Digitized document is a particular case of image. **Content-based Document Image Retrieval** (CBDIR) [MMS03] is a subdivision of CBIR where large scale document retrieval is performed according to users requests. Documents are containers of information made by humans to be read by humans. CBDIR involves a search process where the user's request is a concept to be found.

In these approaches, query text is considered in image format and retrieval is performed using word spotting approaches [WZH00, RM05]. In word spotting, the query text image is treated as a whole entity. Given a single instance of a query word image, the system looks for different instances of the query image in the document and it returns a ranked list of segmented locations where the queried word is probable to be found. Here, the segmentation and ranking are performed using a matching process between query and target images by image based features.

1.2 Graphical Document Recognition

Besides huge amount of documents having only text strings, some documents contain a large amount of information that is usually neglected and can provide richer knowledge. There are documents comprising mostly **graphical information** such as maps, engineering drawings, diagrams and musical scores [WL06]. Old documents contain decorative initials and separators. Administrative documents contain graphical symbols such as logos, stamps or seals.

For efficient processing and storage of graphical documents, however it is necessary to generate a description of graphical elements in the document rather than a bit-map in order to reduce the storage and processing time. Thus, increasing emphasis is being placed on the need for the realization of computer-based systems which are capable of providing automated analysis and interpretation of paper-based documents [KSM85].

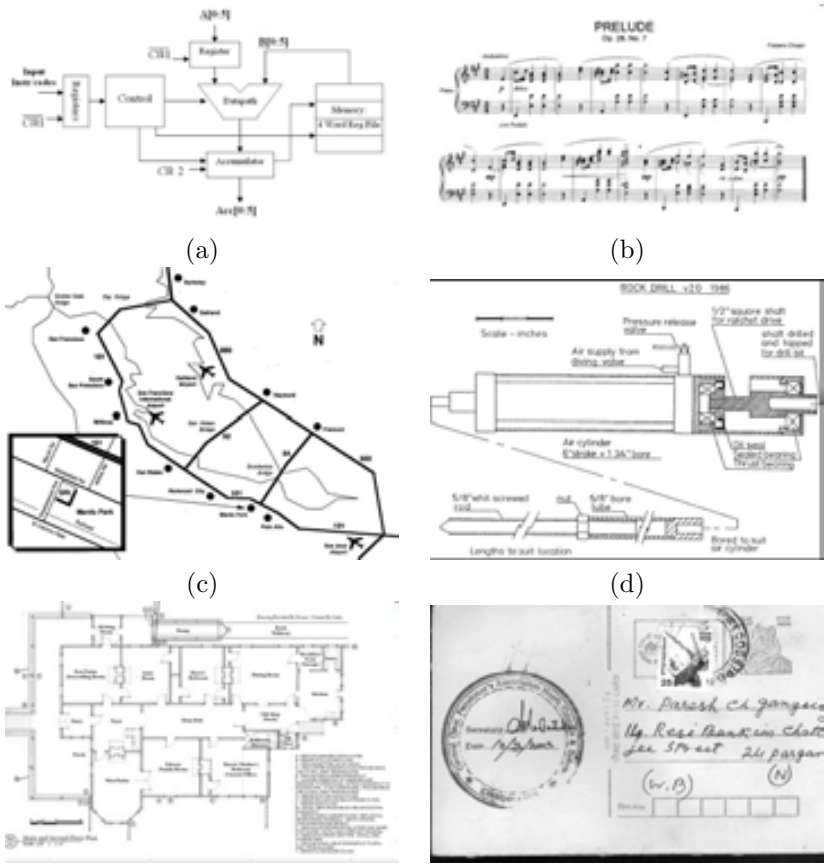
To take care of this problem, **Graphics Recognition** (GR) has become popular in applications such as Optical Music Recognition (OMR) [Bis09], engineering/architectural drawing interpretation [FK88], etc. GR deals with problems such as raster-to-vector techniques, recognition of graphical primitives from raster images of documents, recognition of graphic symbols in charts and diagrams, engineering drawings or diagram analysis of charts, line drawings, tables or forms, 3D model reconstruction from multiple 2-D views, etc.

The graphical symbols are structured into meaningful objects for use in graphical information systems, for example, Computer Aided Design (CAD), Geographical Information Systems (GIS) and multimedia systems. Graphical objects include different symbols e.g. lines, both long and short, and polygons which may be filled or unfilled. Each kind of graphical documents owns a characteristic set of symbols according to their visual properties. In Fig.1.3, we show some examples of graphical documents e.g., logic diagram, musical scores, maps, engineering diagram, architectural diagram and postal document. Apart from text characters, these documents contain different graphical symbols such as musical symbols (Fig.1.3(b)), large graphical objects (Fig.1.3(a), (c), (d), (e)), postal stamp (Fig.1.3(f)).

In graphics rich documents, graphical symbols are often associated with text labels that indicate references or description of component parts of the diagram. Text labels are designed with different fonts (for e.g. “Arial”, “Times New Roman”, etc.) or styles (normal, italics) to annotate the graphical symbol. The text usually appears in the form of strings. The placement and orientation of the text strings are used to indicate the location and extent of features. In maps, curvilinear arrangement of the text letters along the path of river or street is seen frequently.

1.3 Problem Statement

Research in **Analysis and Retrieval of Graphical Documents** is motivated by the large amount of potential applications that the new technologies offer. The access to massive volumes of image data has become easier today. With the progress of research in OCR and DIR, automatic processing of such data is extremely desirable. The users demand for effective tools to manage these large amount of images according to their needs. The problems towards the automation of accessing and indexing the graphical



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

Figure 1.3: Documents containing graphical information. (a) Logic diagram (b) Musical scores (c) Map (d) Engineering diagram (e) Architectural diagram (f) Postal document.

documents are many-fold. Some of them are discussed in the following sections.

1.3.1 Difficulties of OCR in Graphical Documents

Text character detection and recognition are among the main challenges in automation of graphical document processing. The problem for detection and recognition of such text strings is many-folded. In graphical documents, text lines appear frequently in different orientations other than the usual horizontal directions. Sometimes, the text lines in such documents are curvi-linear. The purpose of such orientation and curvi-linearity is to annotate the location of graphical objects. Thus, the inter-character spacing differs according to annotation style. The OCR systems available commercially do not perform well when documents are different in orientations. Hence, the recognition of character strings in such document is difficult due to the usage of

multi-oriented and multi-scale environment.

Text characters are in different orientations due to curved nature of text lines. As a result, it is difficult to detect the skew of such documents and hence **character recognition** of such documents is a complex task [AOC+00]. Due to degradation or noise in documents, characters may touch each other in string. It leads to false character segmentation, and hence false recognition results. **Touching character segmentation** is a major problem for achieving higher recognition rate even in horizontal text OCR [VOB+08]. When touching occurs in multi-oriented documents (e.g. artistic or graphical documents), it is much more difficult to segment such multi-oriented touching than normal horizontal touching [RPLD09].

Separation of text characters and graphical symbols involves many challenges because the text and symbols frequently touch/overlap with graphical components. Automatic **separation of text characters** from graphics in document images is one of the fundamental aims in graphics recognition. It requires proper discrimination of text/graphics [CT01], [FK88], [TTP+02]. Here, the aim is to segment the document into two layers: a layer assumed to contain text characters, and the other one containing the rest of graphical objects representing diagrams, streets, borders of the region, etc.

For the OCR of such documents, we need to extract individual text lines from the documents. As mentioned before, the text lines in graphical documents are not parallel to each other. These text lines may have different orientations or the text lines may be curved shapes. These are annotated in different orientations to illustrate a location or symbol. **Extraction of individual text lines** from multi-oriented and/or curved text documents is a difficult problem, as the text lines may not follow a single direction only [GA99].

We show an example of graphical document in Fig.1.4. A portion of a map is shown to illustrate the problems usually appear in graphical documents. Fig.1.4(a) shows two text characters 't' and 'z' are touched due to degradation of printing and the touching component is in different orientations. In another instance of Fig.1.4(b), due to overlapping, characters 'c' and 'h' are touched with graphical line, making the text/graphics segmentation difficult. It can also be seen from Fig.1.4(c), that the string "Yangtze" is oriented in different directions rather than horizontal direction.

1.3.2 Problems in Graphical Document Retrieval

DIR aims at finding relevant document images based on image features of user's query which is of interest. Indexing of graphical documents can be performed based on text string or graphical entities. We have discussed in Section 1.1.2, how a query text is searched to find relevant documents from a database.

Unfortunately, the OCR process is not perfect, and errors often occur in documents with high degree of degradation such as fax documents due to compression or bilevel conversion, historical documents which are often degraded due to aging or poor typing, or handwritten documents. These errors have an adverse effect on the effectiveness of information retrieval (IR) algorithms that are based on exact matches of query terms and document terms. Searching OCR data is essentially a search of "noisy" or error-filled text [CHTB94].

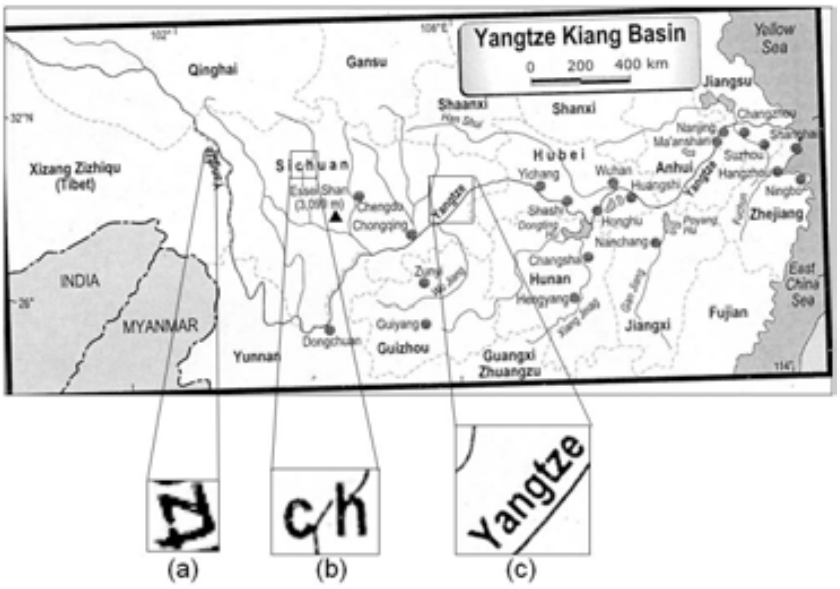


Figure 1.4: Example of some problems typical in a map. (a) Touching characters in multi-oriented direction. (b) Text/Graphics overlapping. (c) Text word not in a horizontal direction.

Sometimes, holistic word spotting [WZH00, RM05] treats each word as a whole entity and thus avoids the difficulty of character segmentation. Word spotting approaches are widely employed in keyword spotting application for degraded document images. This approach is different from the character-level processing strategy taken by OCR, and thus robust to degraded image quality. To avoid the OCR inclusion, query string in ASCII/unicode is converted to image format sometimes and spotting is done accordingly. To cope with the variations of fonts, the textual query is translated into one or more word images (using different fonts/noise models) and later features from each of these images are compared with target images for retrieval [MMS03].

In graphical documents, word-spotting based approaches may not be applicable, because characters may present in different fonts in a single document and their inter-character spacings are not known priori. Also, approaches like template matching, Dynamic Time Warping (DTW) [RM05] may not be useful. Due to curvi-linearity of the characters in a word, generation of such synthetic words may not be possible. Using symbolic encoding of text characters and then use of approximate string matching [Tak01] of such character group can be a good strategy to view ranked queries in unstructured documents.

Searching in terms of symbol queries such as logos or seals will also allow the quick filtering or routing of the document to the right business process, archive or individual. The detection of such symbols in documents undoubtedly increases the performance of document retrieval [UMM98, ZD09]. But, problems like overlapping of graphical object with text, signatures, etc. add difficulty in such object detection.

1.4 Related Work

As we have mentioned, analysis of textual information and graphical objects is a fundamental step in the context of graphical document interpretation. This information is semantically rich and the recognition of this textual information is very important. Literature and commercial products make use of OCR systems in a classical context for horizontal characters and known fonts. However, if we consider the problem of generalization to a different orientation, the number of propositions decreases considerably. There is no reliable industrial product able to recognize words and index them in multi-scale and multi orientation variability.

In graphics rich documents, the presence of graphical objects and their overlapping with text string makes text information retrieval difficult. Text often touch/overlap with long graphical lines in such documents. Thus, interpretation of such documents requires taking care of text/graphics separation and handling of curvi-linear character strings. Due to the problem complexity, different techniques are generally used in a sequential order for this purpose. These techniques concern segmentation of characters, recognition of these isolated characters, touching character separation, and grouping characters into words or lines. In this section, we overview the literature according to this pipeline. Later, each chapter goes further details on each topic of this pipeline.

There exists many pieces of published work on **text/graphics separation**. The algorithm due to Fletcher and Kasturi [FK88] uses simple heuristics based on the characteristics of text characters. Directional mathematical morphology approach has been used by Luo et al. [LAD95] for separation of character strings from maps. The idea is to separate large linear segments by directional morphology and histogram analysis of these segments. Tan et al. [TN98] illustrates a system using *Pyramid* structure. Multi-resolution representations of such a pyramid structure help to select different regions for segmentation. Cao and Tan [CT01] proposed a method of detecting and extracting text characters that are touched with graphics. It is based on the interpretation of intersection of lines in the overlapped region on the vectorized image of text and graphics. A consolidated method, proposed by Tombre et al. [TTP+02] used connected components analysis to make it stable for graphics-rich documents.

For the purpose of **multi-oriented text character recognition**, Adam et al. [AOC+00] used Fourier Mellin Transform. Some of the multi-oriented character recognition systems consider character alignment [HYSS01]. The main drawback of these methods is the distortion due to realignment of curved text. Parametric eigen-space method is used by Hase et al. [HSYS03]. Xie and Kobayashi [XK91] proposed a system for multi-oriented English numeral recognition based on angular patterns. Pal et al. [PKRP06] proposed a modified quadratic classifier based recognition method for handling multi-oriented characters.

There are many published papers towards segmentation of the touching characters of normal horizontal direction [CC99, SLLC05] but to the best of our knowledge, there exist no work towards multi-oriented and multi-sized **touching strings segmentation**. For touching character recognition methods in normal direction, Kahan et al. [KPB87] used projection profiles as the objective function. Fujisawa et al. [FNK92] used profile features for touching numeral segmentation. Later Yu and Yan [YY98]

presented a segmental technique using structural features for single-touching handwritten numeral strings. Dimauro et al. [DIP92] applied contour based features along with some descending algorithms for the touching character segmentation purpose. Oliveira et al. [OLBS00] used contour, profile and skeleton features to find a set of points for touching characters segmentation.

There exist some published works in multi-oriented and curved **text line extraction** in artistic document. Goto and Aso [GA99] proposed a local linearity based method to detect text lines in English and Chinese documents. In another method, proposed by Hones and Litcher [HL94], line anchors are first found in the document image and then text lines are generated by expanding the line anchors. Loo and Tan [LC02] proposed a method using irregular pyramid for text line segmentation in such documents.

There are many software products to retrieve words in document where text lines are in horizontal direction. In this type of documents, the high quality OCR has good performance on the accuracy of retrieval [CHTB94]. On the other hand, there is not much work which can deal with **word searching** in graphical documents. The existing work in the literature [DMS95b, AOC$^+$00] considered the traditional pipeline of character segmentation, grouping characters into words, etc, to apply the OCR later. Deseilligny et al. [DMS95b] proposed an algorithm based on dynamic programming to construct optimal strings by using constraints on orientation homogeneity. A systematic connected-character search is employed at each extremity of detected strings. Most of these approaches consider text lines to be straight in such documents and rarely search curvi-linear words due to its complexity.

Besides ASCII words, there are many graphical entities such as dropcaps [CDOM09], logos [ZD09], seals [Che96], etc. which are also used for indexing of graphical documents. Symbol spotting based methods locate graphical symbols in a document image [NKI05]. **Document retrieval using seal** has already been explored by many researchers. A prior knowledge of the boundary shape of seal (e.g. circular, rectangular, elliptical, etc.) is useful to locate seal in documents [ZJD06]. Often seal has been treated as a symbol and methodology like segmentation by Delaunay tessellation [CW98] are applied for seal recognition. Correlation based algorithm for seal imprint verification is also presented in [HHYY96, Hor01]. Matsuura et al. [MM02] verified seal image by using the discrete K-L expansion of the Discrete Cosine Transform (DCT). Lee and Kim [LK89] used the attributed stroke graph obtained from skeleton, for registration and classification of seal.

The fact of working with retrieval of graphical documents raise several problems to tackle. One of the main reasons could be the complexity, which grows incrementally from character segmentation to text word retrieval. Another important issue is the choice of the methodology allowing us to efficiently retrieve documents based on query word or symbol. Simple OCR methods or rendering of textual query to image for spotting purpose will not be applicable in graphical documents. Because characters may present in different fonts in a single document and their inter-character spacing is not known a priori. Also, due to curvi-linearity of the characters in a word, generation of such synthetic words may not be possible.

1.5 Objectives and Contributions of the Thesis

To solve the problems mentioned above, we have proposed several methods to tackle text/graphics analysis and retrieval of documents based on isolated character labeling. The objective of the thesis are divided in the following blocks.

- On one hand, we study and analyze the **segmentation** problems of text characters in graphical documents which are usually in multi-scale and multi-orient environment. The interpretation of graphical document requires the discrimination of graphical objects and text before applying OCR in text layer.
- Next, for handling text characters in font, scale, and rotation independent environment, the **recognition** process involving feature descriptor and the classifier must be efficient enough.
- We formulate methodologies for graphical **document interpretation** based on text component information. Connected components in graphical documents are recognized and based on the positional information of characters, the text lines are extracted.
- We provide two different application scenarios: one with query word and another with query seal for such retrieval systems. When a query word is searched in an archive of graphical document, the objective is to index the documents that contain the word along with its possible location. Also, a graphical object containing text information such as seal can be used to index the documents.

In the following, we mention in details about our contributions in this thesis. The contributions of research towards **text information extraction** in graphical documents are the following:

- We have developed a robust rotation invariant feature descriptor vector to recognize isolated character/symbol in graphical documents. Size and rotation invariant features are considered here for the recognition of multi-sized/multi-oriented characters and the features are computed from angular information obtained from the external and internal contour pixels of the characters.
- In the context of touching strings recognition, we have also proposed an approach for multi-oriented n-character touching string segmentation scheme. When two or more characters touch, they generate a big cavity region at the background portion. Based on convex hull information, at first, we use this background information to find some initial points to segment a touching string into possible primitives (a primitive consists of a single character or part of a character). Next, some of these primitives are merged to get optimum segmentation. Dynamic programming algorithm is applied for this purpose using the total likelihood of characters as the objective function.
- We have discussed the separation of text and symbols from graphical documents. We developed a system to retrieve text and symbols from graphical documents. Connected component features and skeleton information are used to identify text

characters from graphics lines on the basis of their geometrical features. Next, we proposed the adaptation of the Scale Invariant Feature Transform (SIFT) approach in the context of text character localization (spotting) in graphical documents.

- We have proposed a line extraction technique for artistic/technical documents. To handle documents of wide variations in terms of size, font, orientation, layout, etc., the proposed technique is based on the foreground and background information of the characters in a text line. Most researchers focus their attention only on foreground components and discard the background part (background part refers to those white areas of a binary image surrounding the actual black foreground).

Contributions towards **graphical document retrieval** are the following:

- We present a novel word indexing architecture in graphical documents to facilitate searching of query word which can take care of overlapping/touching cases. We use the spatial information of labeled connected components as the high level descriptors. A generic architecture is proposed to index such spatial information. Given a query text, the system extracts positional knowledge from the query and use the same to search the possible zones to generate hypothesis. These hypotheses are further reduced to find the probable locations of a word. Spatial arrangement of character components drives our system to locate text word in such non-structured environment. The system works on multi-scale and multi-rotation environment.

- Sometimes, text and graphics are mixed in an object and the understanding of the object depends on the mutual information between them. "Seal" is one of such graphical entities that contains text characters along with a logo and a graphical boundary surrounding it. In the earlier research works, this kind of object has been treated as a whole entity and symbol recognition based approaches are used to interpret it. In this thesis, we propose a scheme on document indexing based on seal object detection. The main contribution is the use of recognized local text components as high level descriptors and the generation of the hypotheses of the seal location based on spatial arrangement of these descriptors. Our approach is based on local text characters instead of only pixel-level descriptors to overcome distortion and affine-transformation problems which are the main drawbacks in existing approaches. Also, we have formulated our approach to be scalable and adaptable to process large collection of documents. It does not need a previous segmentation, thus provides direct focus on seal detection.

1.6 Organization of the Thesis

The organization of the chapters in this dissertation is as follows:

- In Chapter 1, a preview of the whole thesis has been provided, including the scope of the thesis, the problems and contributions.

- In Chapter 2, we propose a novel scheme towards the recognition of multi-oriented and multi-scale text characters using angular information obtained from the contour pixels of the characters. Circular and convex hull rings have been used to divide a character into smaller zones to get zone-wise features for higher recognition results. We combine circular and convex hull features and these features are fed to a Support Vector Machine (SVM) for recognition.

- In Chapter 3, we present an approach towards the segmentation of multi-oriented touching strings into individual characters. For this purpose, convex hull based background information are used to segment a touching string into possible primitive segments and later these primitive segments are merged to get optimum segmentation using dynamic programming.

- In Chapter 4, we discuss in detail about the automatic separation of text and graphics in document image. We will present an approach to separate text and symbols from graphical documents. Here, connected component features and skeleton information are used to identify text and symbol from graphics on the basis of their geometrical features. Next, we propose the adaptation of the Scale Invariant Feature Transform (SIFT) approach, in the context of text character localization (spotting) in graphical documents.

- In Chapter 5, we propose a novel method to extract individual text lines from such document pages, and the method is based on the foreground and background information of the characters of the text. To effectively utilize the background information of the string, the water reservoir concept is used.

- In Chapter 6, we present an approach towards the retrieval of words from graphical document images. The recognition results of text characters are used to form character pairs with the neighboring components. An indexing scheme is designed to store the spatial description of components and to access them efficiently. Given a query text word, the character pairs present in the document are searched first. Next the retrieved character pairs are joined to form character string and a string matching is used to find the query word.

- Chapter 7 deals with document retrieval based on a graphical symbol "seal". A seal is characterized by scale and rotation invariant spatial feature descriptors computed from recognition result of individual connected components (characters). The concept of Generalized Hough Transform (GHT) is used to detect the seal. A voting scheme is designed for finding possible location of the seal based on the spatial feature descriptor of component pairs. The peak of votes in GHT accumulator validates the hypothesis to locate the seal in a document.

- Conclusions are given in Chapter 8. First, a summary of the main contributions has been given. Afterwards, we discuss about the proposed approaches and their corresponding experimental results. Finally, future work is reported.

- Finally, the datasets used in the experiment are described in the Appendix.

Chapter 2

Multi-Oriented Text Character Recognition

This chapter deals with text character recognition in multi-scale and multi-oriented environment. We propose a novel scheme towards the recognition of such text characters using background and foreground information. The features are computed from different angular information obtained from external and internal contour pixels of the characters. Angular information are computed in such a way that they do not depend on the size and rotation of the characters. Circular and convex hull rings have been used to divide a character into smaller zones to get zone-wise local features for higher recognition results. We also include angular slope of contour pixel to add more discriminating power in the feature. We combine these angular features and feed them to a SVM (Support Vector Machines) classifier for recognition. We have tested our approach on character datasets of English and two popular Indian scripts namely: Devnagari and Bengali.

2.1 Introduction

As stated in the introductory chapter, Optical Character Recognition (OCR) involves electronic translation from scanned images of handwritten, typewritten or printed text into digital images and translating these images into machine-encoded form (say ASCII codes) that computer can manipulate. As the amount of new written material increased, the need to process all in a fast and reliable manner grown. Today there are many OCRs that are designed using different algorithms. All of the popular algorithms goes for high accuracy and high speed. Many OCR softwares e.g. Abbyy[1], Iris[2], Cvision[3] etc. which are available commercially have been developed to cope up with this growth. These softwares deal with the text in scanned documents where text lines are usually horizontal and they are parallel to each other.

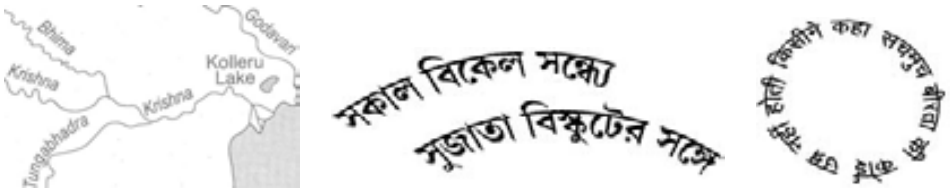
[1]http://www.abbyy.com/
[2]http://www.irislink.com/
[3]http://www.cvisiontech.com/

Many documents are printed in stylistic (artistic) way to catch people's attention. With the advancement of information technology, recognition of characters and symbols in such documents becomes an extremely important issue. Handling text character in such documents is a challenging research topic, aiming at recognizing characters in multi-scale and multi-rotation environment and overcoming all difficulties encountered.

Our contribution of this chapter is an important task of this thesis, as it will be noticed along the thesis, "recognition of individual characters in multi-oriented and multi-sized environment" drives different modules of the rest of the thesis.

2.1.1 Motivation

There are printed artistic or graphical documents where text lines of a single page may have different orientations. The text lines in these documents may be written in curved, rotated, or in other stylistic ways. As a result, it is difficult to detect the skew of such documents and hence character recognition of such documents is a complex task. Some examples of such complex documents are shown in Fig.2.1 for illustration. In Fig.2.1(a), we show a map, where English characters are in different rotation for annotation purpose. Fig.2.1(b) presents a Bengali magazine document where characters are printed in curvi-linear way for attention. Fig.2.1(c) shows a synthetic document of Devnagari where text characters are in different rotations.



(a) (b) (c)

Figure 2.1: Examples of graphical/artistic document images (a) Map (b) Bengali magazine (c) Devnagari synthetic image.

The OCR systems that are available commercially, do not perform well in handling these rotated characters. In this chapter, we will tackle this problem by proposing a recognition method, which is applicable to characters and symbols in script independent way. The essence of our method is a shape feature descriptor based on angular information of contour pixels.

2.1.2 Related Work

Shape description is an important issue of symbol recognition. Feature extraction can have strong influence in the performance of symbol recognition tasks. A suitable representation should be compact, complete i.e. general enough to represent symbols from different domains, discriminant i.e. able to maximize the intra-class similarity and the inter-class dissimilarity, computationally manageable, extensible and able to support distortion models [LVSM02].

Many shape description methods have been developed and reported in the past. A detailed review of shape description techniques can be found in [ZL04]. A rotation invariant feature is defined as a special kind of feature that is invariant when the input pattern is rotated. Some previously proposed rotation invariant features are ring projection, the number of strokes, the number of stroke intersections, the number of multi-fork points, Fourier descriptor moments, classical rotation invariant moments, and Zernike invariant moments, Hu's moment, etc.[DMS95b, KH90b, PL92, WCL94, Hu62].

Review of the State-of-the-Art Approaches

In the literature there exists some pieces of work on English and Chinese multi-oriented text recognition [LWT04, SMA00, TCS91, UIOK06, USI+07]. Some of the multi-oriented character handling approaches consider character re-alignment for recognition. The proposed approach by Hase et al. [HYSS01] is based on the types of the text (horizontal, vertical, curved, inclined etc.), where characters in a text line are re-aligned horizontally and then OCR techniques are used. The main drawback of these methods is the distortion due to the re-alignment of curved text.

Xie and Kobayashi [XK91] proposed a rotation invariant recognition system using plural standard patterns with different inclinations (angular variation) of the component and they obtained 97% recognition accuracy from the 10 English digits.

Adam et al. [AOC+00] used a new set of features using Fourier Mellin Transform for classification of multi-oriented and multi-scaled patterns. This method was applied to the documents of the French Telephonic Operator.

Parametric eigen-space based method is used by Hase et al. [HSYS03] for rotated and/or inclined character recognition. First they construct an eigen sub-space for each category using the covariance matrix which is calculated from a sufficient number of rotated characters. Next, a locus is obtained by projecting the rotated characters onto the eigen sub-space and interpolating between their projected points. To recognize a query character, the character is also projected onto the eigen sub-space of each category and verification is done from the distance between the projected point of the query character and the locus.

Yang and Wang [YW01] proposed a three-stage system for multi-oriented Chinese character recognition where features are computed from each pair of pixels. The approach is mainly based on geometric measures of the foreground pixels of the characters.

Monwar et al. [MHP07] proposed a rotation invariant approach where they treat each character as a two-dimensional recognition problem, taking advantage of the fact that characters can be described by a small set of 2D characteristic views of different angles (0°-360°). Character images of different angles are projected onto a feature space that best encodes the variation among known character images.

Hayashi and Takagi [HT06] proposed a rotation invariant recognition system for English numerals. Thinning algorithm is used initially to divide a numeral into elementary sub-patterns like straight line, C-shaped line, and 0-shaped line. Next, based on different features like curvature, angle information, length and arc-length etc. of the sub-pattern, the numeral is recognized.

Tsai and Chen [TC02] proposed a rotation invariant pattern matching technique using wavelet decomposition and ring projection representation. The multi-resolution wavelet technique reduces an original image to small sub-images at a low multi-resolution level. The approach transforms the image into a representation in which both spatial and frequency information present.

Circular primitives, which are well adapted to rotation invariant recognition, have been used by Kita [Kit92]. This is based on the analysis of the shape through a set of concentric circles. A set of circular strobes has been used to describe the shape around its centroid.

Recently, Iwamura et al. [ITHK09] proposed a simple algorithm of camera-based recognition of characters and pictograms. With the help of new geometric hashing and voting techniques, the proposed method runs in real-time for rotated character recognition.

In Table 2.1, we summarize the state-of-the-art approaches for multi-oriented text character recognition methods. Most of the existing works [HSYS03, ITHK09] proposed in the literature considered characters of noise free. A small set of character dataset is used for the performance evaluation. Some particular fonts of characters are used in most of the existing works [AOC$^+$00, HSYS03]. Above all, the shape descriptors proposed in the literature for character recognition have been tested only in Latin character datasets. Their performances have not been evaluated in the datasets of other scripts.

Table 2.1: Related work proposed on multi-oriented character recognition.

Method	Key Feature in the Proposed Algorithm
Xie and Kobayashi [XK91]	Templates at different angular variation
Hase et al. [HYSS01]	Re-alignment in horizontal direction
Adam et al. [AOC$^+$00]	Fourier Mellin transformation
Hase et al. [HSYS03]	Parametric eigen-space
Yang and Wang [YW01]	Geometric feature
Monwar et al. [MHP07]	Projection at different angles
Hayashi and Takagi [HT06]	Structural features in thinned image
Tsai and Chen [TC02]	Wavelet decomposition and ring projection
Kita [Kit92]	Circular primitives
Iwamura et al. [ITHK09]	Geometric hashing and voting

2.1.3 Outline of the Approach

Given these large numbers of existing methods, one could argue that it is not necessary to develop new invariant features. Nevertheless, comparative evaluation studies [BSA91] have shown that the features are not perfect in terms of recognition accuracy, especially when the images are noisy. Also, since symbols and characters are isolated (i.e., not integrated in a string), the orientation information is missing. Furthermore their size can be variable, as can be their orientation.

In this chapter, we propose a novel recognition scheme for isolated text character

in multi-sale and multi-oriented environment. Size and rotation invariant features are computed from different angular information obtained from the external and internal contour pixels of the characters. Angular information is computed in such a way that it does not depend on the size and the rotation of the characters and they make the feature rotation invariant. Circular and convex hull rings have been used to divide a character into smaller zones to get more local features for higher recognition results. Finally, we use the combination of these local features to improve the results. Support Vector Machine (SVM) classifier is used for recognition.

We give an exhaustive performance evaluation of our proposed approach using different datasets. We have used character dataset of 3 different scripts namely: English, Bengali and Devnagari for the experiment. Next we show the performance using Mpeg-7 and handwritten symbols.

The overall organization of the rest of the chapter is as follows. In Section 2.2, the feature descriptor used to describe the isolated multi-oriented and multi-sized characters is detailed. We explain SVM classifier in Section 2.3. The experimental results are discussed in Section 2.4. Finally, conclusion is given in Section 2.5.

2.2 Feature Extraction

To handle multi-sized/multi-oriented character recognition, we have considered size and rotation invariant features and the features are mainly based on relative angle information of the contour pixels of the characters. This feature is considered as global feature. To get local features, circular and convex hull ring based zoning and angular slope are used. Details of the feature extraction are given as follows.

2.2.1 Global Feature

For a character, the relative position of a pixel with respect to its neighbouring pixels will not change even if the character is rotated in arbitrary directions. Hence relative angle of the pixel with respect to its neighbouring pixels will also not change. Since relative angle does not change because of rotation, we use this information for multi-oriented character recognition.

For an input image, internal and external contour pixels are noted and they are used to determine the relative angle feature of the image. Given a sequence of consecutive contour pixels $V_1 \dots V_i \dots V_n$ of length n, the relative angle of the pixel V_i is the angle formed at V_i by three pixels V_{i-k}, V_i and V_{i+k} (Here, $2 \times k$ is the total number of the neighbor pixels of V_i considered for angle computation, k pixels before V_i and k pixels after V_i). From each contour pixel we will get two angles (one from the background side and other from the foreground side) and the angle corresponding to the background side is considered here. See Fig.2.2, where the relative angle computation of contour pixels is illustrated. In this figure, β_i is the angle obtained at the contour pixel V_i, when $k = 3$. For a better relative angle information of a pixel V_i, we consider two angles, obtained by considering $k = 3$ and 4, and the average of the two angles is the estimated value of the relative angle of the pixel V_i. To choose the value of k we considered different pair of values [(k=2,k=3), (k=3,k=4), (k=4,k=5), (k=5,k=6) and (k=6,k=7)] and noted that average angular information obtained from the pair

(k=3,k=4) showed best recognition results in our method (see Table 2.9) and hence we considered k = 3 and 4.

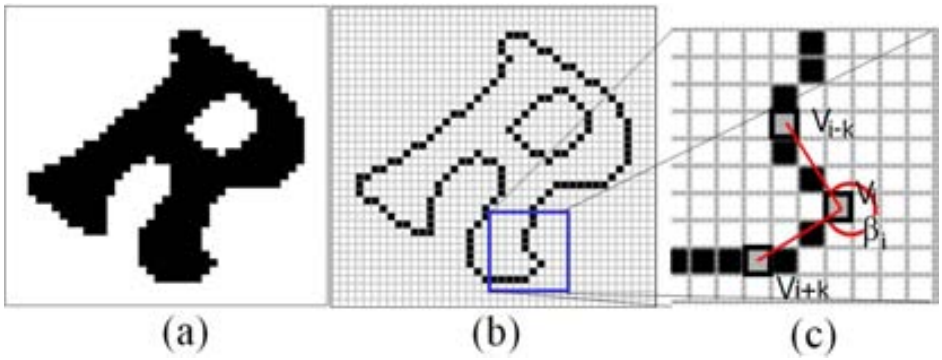


Figure 2.2: Illustration of relative angle computation from contour pixels. Here (a) is the original image, (b) contour pixels of the image shown in (a), and (c) an enlarged version of a portion of (b) where β_i is the angle at the contour pixel V_i when k=3.

The relative angles obtained from all the contour pixels of a character are grouped into 8 bins corresponding to eight angular intervals of 45° (0°-45°, 45°-90°, 90°-135°, etc.). For a character, the frequency of these angles of 8 bins will be similar even if the character is rotated at any angle in any direction. For illustration, see Fig.2.3. Here, we compute the histogram of the relative angles corresponding to 8 bins of two rotated shapes of three characters 'W', 'G' and 'T'. From the figure it can be noted that the angle histogram of each pair of characters is similar although the characters have different rotations.

2.2.2 Local Feature

We may feed this global angular frequency of the characters into the classifier for recognition. To get higher accuracy we divide a character into zones to have zonal information and zone-wise relative angle information is computed. Circular and convex hull rings have been used to get zone-wise features. Angular slope information of pixels are also used to get more local features. These are discussed as follows.

Circular Ring based Division

A set of seven circular rings is considered here and they are defined as the concentric circles considering their center as the center of Minimum Enclosing Circle (MEC) of the character and the minimum enclosing circle is considered as the outer ring. The radii of the rings are chosen such that, the area of each ring is equal. Let R_1 be the radius of MEC of the character, then the radii (outer to inner) of these seven rings are R_1, R_2, R_3, R_4, R_5, R_6, R_7 respectively. Where,

$$\Pi R_1^2 - \Pi R_2^2 = \Pi R_2^2 - \Pi R_3^2 = \Pi R_3^2 - \Pi R_4^2 = \Pi R_4^2 - \Pi R_5^2 = \Pi R_5^2 - \Pi R_6^2 = \Pi R_6^2 - \Pi R_7^2 = \Pi R_7^2$$

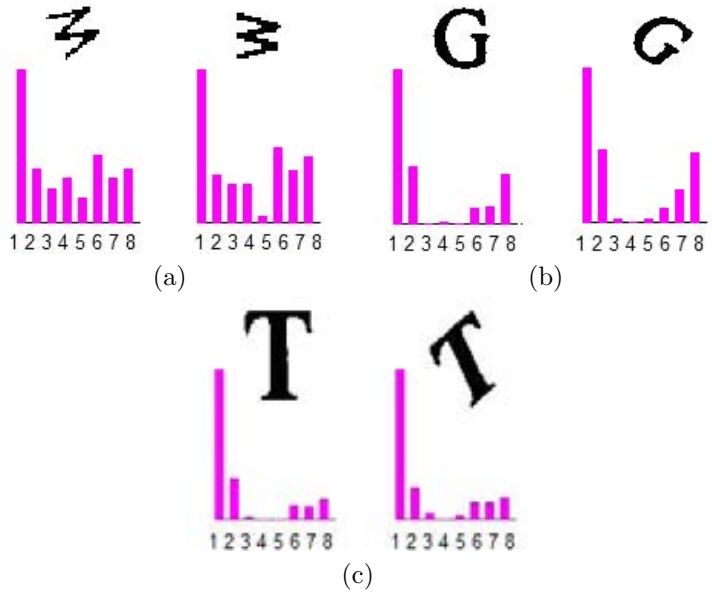


Figure 2.3: Input images of the characters 'W', 'G', 'T' in 2 different rotations and their angle histogram of contour pixels are shown. The numbers 1-8 represent 8 angular bins.

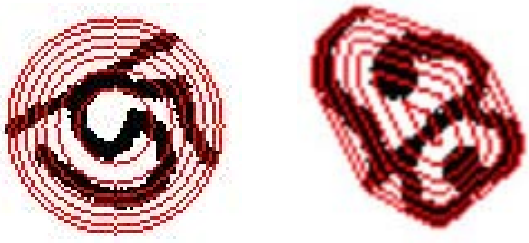
See Fig.2.4(a), where these rings are shown on a Bengali character. These circular rings divide the MEC of a character into seven zones. Number of zones was decided as 7 based on the experiment (see Table 2.8).

Convex Hull based Division

The Convex Hull [Gra72] or convex envelope for a set of points X in a real vector space of d-dimensional data R^d is the minimal convex set containing X. The convex hull $conv(X)$ is defined as the smallest convex set in R^d containing X. In another way, a convex hull is a minimal convex shape entirely bounding an object. It may be easily visualized by imagining an elastic band snapped around an object shape.

Convex hull rings are computed based on convex hull boundary of a character. Let C_H be the outer boundary of the convex hull of a character. We compute 7 convex hull rings and let the outermost convex hull ring be H_1. The outer boundary of the ring (C_{H1}) is the C_H itself. If the outer boundary C_H is reduced inside by a width of $R_1 - R_2$ pixels then inner boundary of C_{H1} is obtained, and the outermost ring C_{H1} is the portion between these two boundaries. Other 6 convex hull rings are similar in shape but smaller in size and they are computed as follows. The 2nd convex hull ring (say C_{H2}) is obtained by reducing the inner boundary of C_{H1} by a width of $R_2 - R_3$ pixels inside. Similarly, the 3rd convex hull ring can be obtained if the inner boundary of C_{H2} is reduced by a width of $R_3 - R_4$ pixels inside, and so on. Convex hull rings are shown in Fig.2.4(b) in a Bengali character. Values of $R_1 \ldots R_2 \ldots R_7$

are the radii of the circular rings discussed in the previous paragraph.



(a) (b)

Figure 2.4: (a) Seven circular rings and (b) Seven convex hull rings are shown on Bengali character.

Note that MEC is circular in nature for all objects and hence rings obtained from MEC are also circular. In contrast, convex hull of an object is a closed polygon and its shape differs depending on the shape of objects. Convex hull ring may not be circular and it is better adjusted to the shape of the object. Also both are invariant to rotation due to their intrinsic properties. Convex hull ring uses the zoning based on the border pixels of the hull whereas the circular ring works based on circle. Thus, the feature computed from convex hull and circular ring work in different manner for each object shape. To get such different features we used both circular and convex hull ring for our experiment. To get the feature, we compute the frequency of the relative angle (as discussed before) of the contour pixels present in each of the rings. For a ring, we get 8 relative angle information and hence we have 56 features (7 rings $\times$ 8 relative angle) in each of the circular and convex hull based divisions.

Angular slope with respect to center of MEC

In the earlier section, we have used relative angle computed from neighborhood contour pixels for global features. Here, we compute angular feature based on slope of the contour pixels with respect to the center of the MEC and combine them. First, we group the contour pixels of a character into 8 bins based on the relative angle with respect to the neighbor pixels. Now, the slope with respect to the center of MEC of the bin-wise contour pixels is computed. Here, by the slope of a pixel V_i with respect to center of MEC, we mean the angle formed at V_i by $\overrightarrow{OV_i}$ and $\overrightarrow{V_iV_{i+k}}$. O is the center of the MEC of a character and k is distance of neighbor pixels as discussed earlier. The slope obtained from all the contour pixels of a bin are further grouped into 8 sets corresponding to 8 angular interval of 45° (=360/8). Thus, for 8 bins of contour pixels (distributed by relative angles), we have $8\times8 = 64$ dimensional slope features. See Fig.2.5 where the angular slope of two contour pixels are illustrated. In this figure, γ_i and γ_j are angles obtained at two different contour pixels V_i and V_j respectively. It is to be noted that, these angles γ_i and γ_j are different.

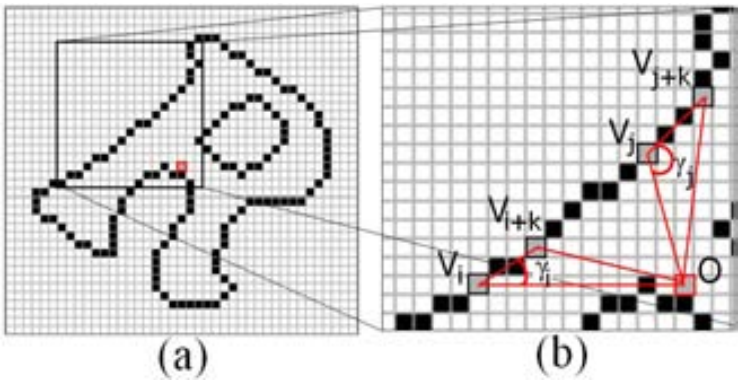


Figure 2.5: Illustration of angular slope computation of contour pixels. Here (a) is the contour image of Fig.2.2 with its MEC marked in Red. (b) an enlarged version of a portion of (a) where γ_i and γ_j are angles at two different contour pixels V_i and V_j with MEC ('O') respectively.

2.2.3 Combination of Angular Features

Finally, considering 7 circular rings and 7 convex hull rings, we have 56 (8 relative angular bins $\times$ 7 circular rings) features from the circular rings, 56 (8 relative angular bins $\times$ 7 convex hull rings) features from the convex hull rings. We also have 64 (8 relative angular bins $\times$ 8 sets of angular slopes) features from angular slope with respect to the center of MEC. As a result, we have a 176 (56+56+64) dimensional feature vector for the classification. It is to be noted that, the feature vector is rotation invariant. To obtain scale invariance, the feature vector is normalized. The feature vector is divided by the total number of contour pixels for this purpose. Hence, we get the feature value between 0 and 1.

We have shown the 2D-plot of 176 dimensional features of characters between intra-class and inter-class characters in Fig.2.6. From the figure it can be seen that the corresponding features of same character classes are similar although the characters are multi-oriented. This combined feature vector is fed to classifier for obtaining classification label.

2.3 Classification

From a recent literature survey due to Liu et al. [LS09], we noted that SVM classifier with RBF kernel shows better performance than other classifiers like MLP Neural Network, Polynomial Network Classifier (PNC), Discriminative Learning Quadratic Discriminant Function (DLQDF), etc. This leads us to work with SVM classifier in our classification stage. Fig.2.7 shows the schematic diagram for our classification approach. It uses a particular training set of examples with labels. During training, the learning algorithm constructs a decision rule which can then be used to predict the labels of new unlabeled examples. For recognition, we feed the features in a SVM classifier. In the following, we will detail about SVM classification process.

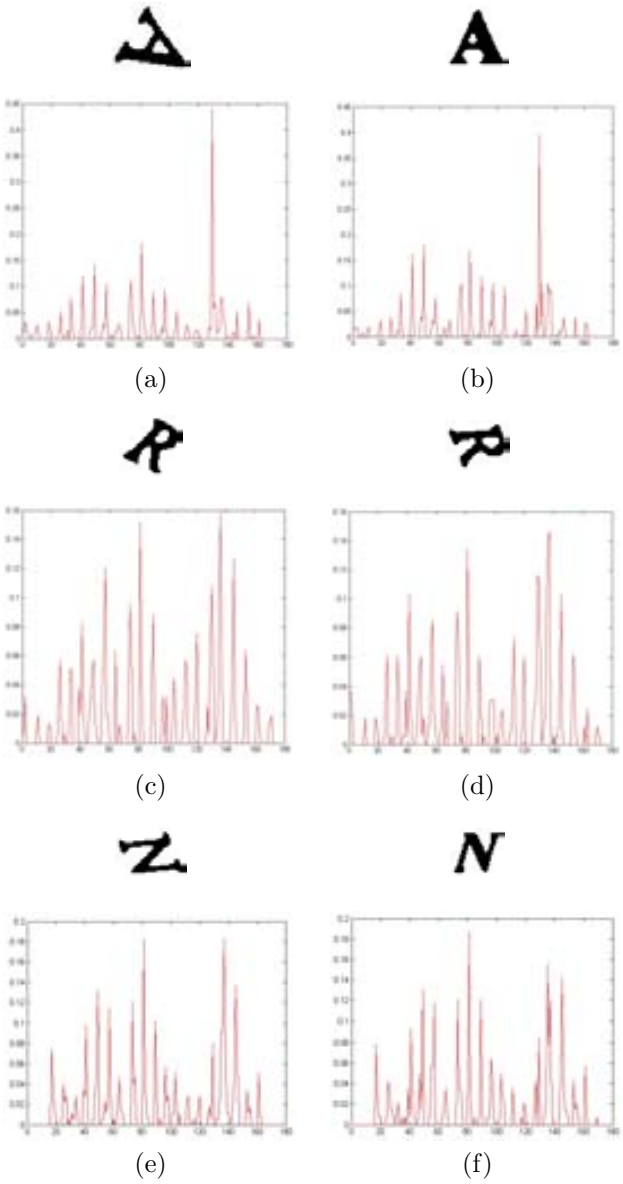


Figure 2.6: Feature vectors are plotted for different orientations of characters 'A', 'R' and 'N'.

2.3.1 SVM Classifier

The Support Vector Machine (SVM) classifier is a machine learning approach based on the structural risk theory introduced by Vapnik in [Vap95]. It is successfully applied in a wide range of applications [BLGT06]. In particular, a SVM classifier is

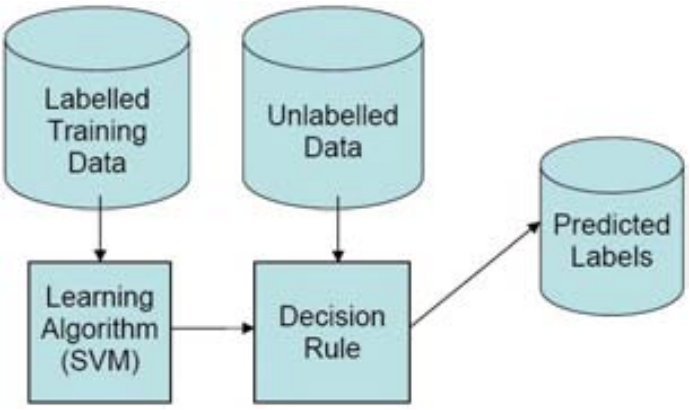


Figure 2.7: The inductive inference process of SVM in schematic form.

capable of finding the optimal hyperplane that separates two classes. This optimal hyperplane is a linear decision boundary separating the two classes and leaving the largest possible margin between the samples of the two classes. Unlike most learning algorithms based on empirical risk, the SVM does not depend on probability estimation. This characteristic makes it more robust against the well known curse of dimensionality, mainly for small datasets, since classification success does not depend on the dimensions of the input space [Mar10].

In particular, the training of an SVM classifier can be summarized as follows. Consider a set of labeled training samples represented by $D = (x_1, y_1), \ldots, (x_n, y_n)$, where $x_i \in R^d$ denotes a d-dimensional vector in a space, and $y_i \in \{-1,+1\}$ is the label associated with it. The SVM training process, which produces a linear decision boundary (optimal hyperplane) that separates the two classes (-1 and +1) can be formulated by minimizing the training error while maximizing the margin separating the samples of the two classes.

$$\texttt{min} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i,$$

$$\texttt{subject to} \quad y_i((w^T x_i) + b) \geq 1 - \xi_i, \xi_i \geq 0, i = 1 \ldots n \tag{2.1}$$

where, w is a weight vector orthogonal to the optimal hyperplane, b is the bias term, C is a trade-off parameter between error and margin, and ξ_i is a non negative slack variable for x_i. The optimization problem in Equation 2.1 is usually solved by obtaining the Lagrange dual, which can be reformulated as:

$$\texttt{max} \quad \frac{1}{2}\sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j x_i x_j$$

$$\texttt{subject to} \quad 0 \leq \alpha_i \leq C, \sum_{i=1}^{n}\alpha_i y_i = 0 \tag{2.2}$$

where $(\alpha_i)_{i\in n}$ are Lagrangian multipliers computed during the optimization for each training sample. This process selects a fraction l of training samples with $\alpha_i \geq 0$. These samples are called support vectors and are used to define the decision boundary. In extreme cases, the number of support vectors will be the same as the number of samples contained in the training set. As a result, the w vector can be denoted as $\sum_{i=1}^{n} \alpha_i y_i x_i$. Fig.2.8 illustrates the general idea of the decision boundary computed by the SVM where there are two classes (circles and squares) separated by an optimal hyperplane. The training samples that were selected as support vectors are located under and between the dashed lines (margin).

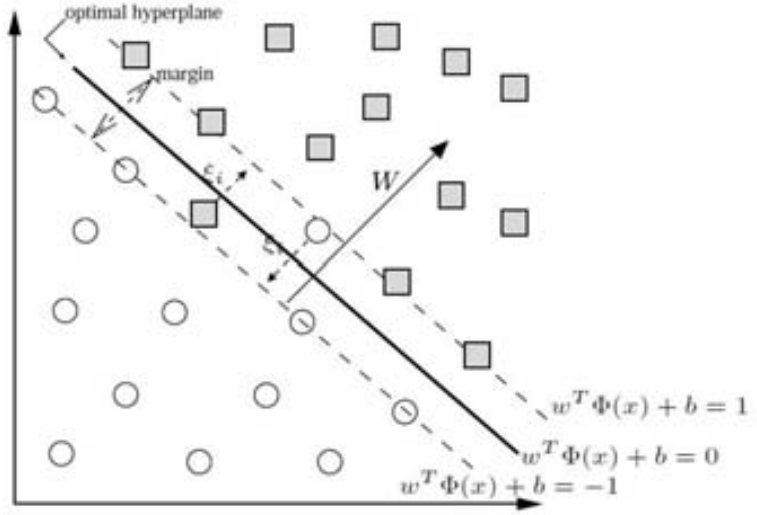


Figure 2.8: Illustration of SVM optimal hyperplane separating two classes.

However, this SVM formulation only works for linearly separable classes, and, since real-world classification problems are almost never solved with a linear classifier, an extension is needed for nonlinear decision surfaces. To solve this problem, the dot products $(x_i.x_j)$ in the linear algorithm are replaced by a nonlinear kernel function K(.), where $K(x_i, x_j) = \phi(x_i).\phi(x_j)$, and ϕ is a mapping function $\phi : R^d \longmapsto H$. Such a replacement constitutes the so-called "kernel trick" [Bur98]. In order to work, the kernel function $K(x_i, x_j)$ must satisfy the Mercer condition [Vap95]. The kernel trick enables the linear algorithm to map the data from the original input space R^d to some different space H (possibly infinitely dimensional), called the feature space. In this space, nonlinear SVMs can be generated, since linear operations in that space are equivalent to nonlinear operations in the input space. The most common kernels used for this task and their parameters $(\gamma, r, u$ and $\tau)$ are listed in Table 2.2. The decision function derived by the SVM classifier for a test sample x and training samples x_i can be computed as follows, for a two-class problem:

$$sign(f(x)) \texttt{ with } f(x) = \sum_{i}^{l} \alpha_i y_i K(x_i, x) + b \tag{2.3}$$

Table 2.2: Most common used kernels in SVM.

Kernel	Inner Product Kernel
Linear	$\mathrm{K}(x_i, x_j) = x_i^T x_j$
Polynomial	$\mathrm{K}(x_i, x_j) = (\gamma x_i^T x_j + r)^u$, u > 0
Radial Basis Fucntion (RBF)	$\mathrm{K}(x_i, x_j) = \exp(\text{-}\gamma \left\|x_i - x_j\right\|^2)$, $\gamma > 0$
Sigmoid	$\mathrm{K}(x_i, x_j) = \tanh(\gamma x_i^T x_j + \tau)$

In the same way this extension deals with nonlinear problems, the primary SVM formulation requires additional modification to solve multi-class problems ($c > 2$). There are two approaches for handling this:

- One-Against-One (OAO): This strategy arranges pairs of classifiers into separate classes, and is also called a pairwise scheme, where the total number of classifiers is $c(c-1)/2$. Given a test sample, the classification result is obtained by comparing the pairs and assigning the class with the maximum number of votes to it.

- One-Against-All (OAA): In contrast, the one-against-all strategy yields one classifier for each class c that separates that class from all the other classes. The final decision is made by the winner-takes-all method, in which the classifier with the highest output function designates the class.

In this work, we use the *OAO* strategy, since it has been demonstrated to be faster to train and uses fewer support vectors than the *OAA* approach [Bur98]. Overall, the SVM is a powerful classifier with strong theoretical foundations and good generalization performance. We used the freely available SVM software package libSVM[1] [CL] implemented by Chang and Lin. In the following section, the character recognition results are provided using our multi-oriented shape feature descriptor and SVM classification tool.

2.4 Result and Discussions

In order to examine the proposed method, we designed and conducted experiments with different datasets. We constructed datasets of text characters from 3 different scripts namely: English, Bengali and Devnagari. We also evaluated the proposed method in the dataset of pictorial symbols (MPeg-7) and hand-written symbols (music symbols). In the following sections, we will discuss about these datasets and accuracy in different experimental evaluation.

2.4.1 Experiment with Text Characters

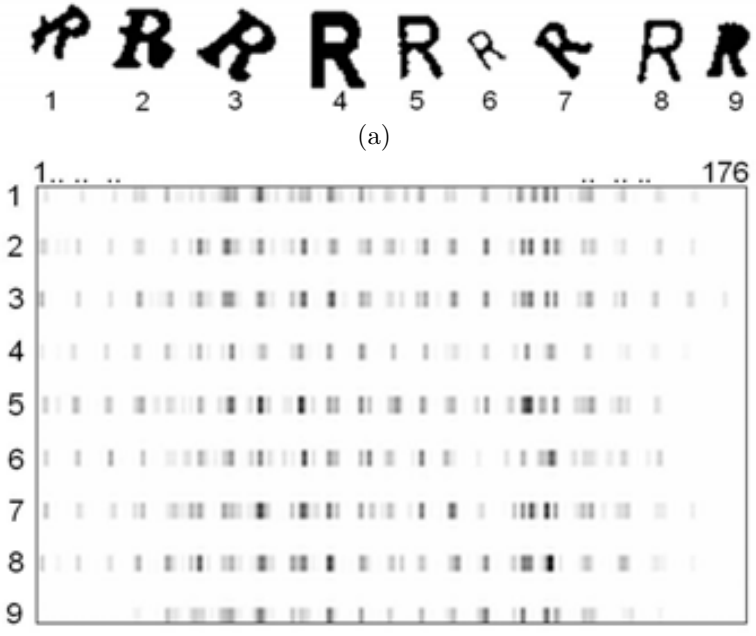
In this section we will discuss about the types and properties of isolated text characters, collected from different scripts.

[1] http://www.csie.ntu.edu.tw/ cjlin/libsvm/

English Isolated Character

For the experiment of English character recognition, we considered data from two different sources. One source of data is from graphical documents (e.g. map, seal documents) only and its size is 8,250 characters. Second part of data was the same data used in Pal et al. [PKRP06] and the size of this data is 18,232 (See Appendix A.1.1 for more details). Finally, we have 26,482 data from English character dataset. Data are scanned at 300 dpi and we have used a histogram based global binarization algorithm [Ots79].

Both English uppercase and lowercase alpha-numeric characters were considered for our experiment, so we should have 62 classes (26 for uppercase, 26 for lowercase and 10 for digit). We are considering arbitrarily rotation (any angle up to 360 degrees) of the shapes and because of shape similarity due to orientation, some of the characters/digits like 'd' and 'p'; 'b' and 'q'; '6' and '9'; etc. are grouped together. We will get the character 'p' if we rotate the character 'd' 180 degrees. Hence, we considered total 40 classes. We show the similar character classes in Table 2.3. Different fonts have been trained for recognition. To get an idea of quality of extracted text characters, we show some samples of character 'R' in Fig.2.9(a). Gray value representation of the feature vector of the multi-oriented samples are shown in Fig.2.9(a) are displayed in Fig.2.9(b).



(a)

(b)

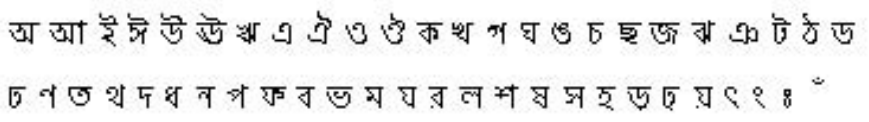
Figure 2.9: (a) Some samples of character 'R' from the dataset are shown. (b) Each element of 176-feature vector are displayed horizontally with its gray scale value for each sample of 'R'.

Table 2.3: List of similar shaped characters. Characters in a cell are treated as same class.

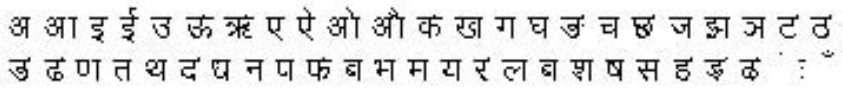
b q	P p d	7 L	0 O o	6 9	1 I l i	J j
S s	u n U	V v	N Z z	X x	W w M m	C c

Bengali and Devnagari Characters

The alphabet of the modern Bengali script consists of 11 vowels and 39 consonants. These characters may be called basic characters. Out of the 49 basic characters of Devnagari script, 11 are vowels and 38 are consonants. Characters of different font sizes 12, 16, 20, 26, 30, 36 and 40 point-size are considered for the experiment (See Appendix A.1.2 for more details). The basic characters of Bengali and Devnagari scripts are shown in Fig.2.10(a) and Fig.2.10(b) respectively.



(a)



(b)

Figure 2.10: Basic characters of (a) Bengali and (b) Devnagari alphabet are shown. First eleven are vowels and rest are consonants in both the alphabets.

To compute the accuracy of the proposed recognition scheme we consider only basic characters of Bengali and Devnagari scripts. We tested our system on 15,389 characters (7874 Bengali and 7515 Devnagari characters). To get the idea of quality of data used in our experiment, some samples of the data are shown in Fig.2.11.

Metric used for Experiment

The feature vectors, obtained from character shape descriptor are passed to a SVM classifier to get the recognition label. Gaussian kernel with Radial Basis Function (RBF) has been chosen in our experiments to recognize multi-oriented text characters. We noticed that Gaussian kernel gave highest accuracy when the value of its gamma parameter is 1.5 and the penalty multiplier parameter is set to 100. Hence, the parameters are set as above for the whole experiment. The classification is done in 5-fold cross-validation way. Here, the data set is divided into 5 subsets, of which 4 subsets were used for training and rest for testing. This process is repeated 5 times.

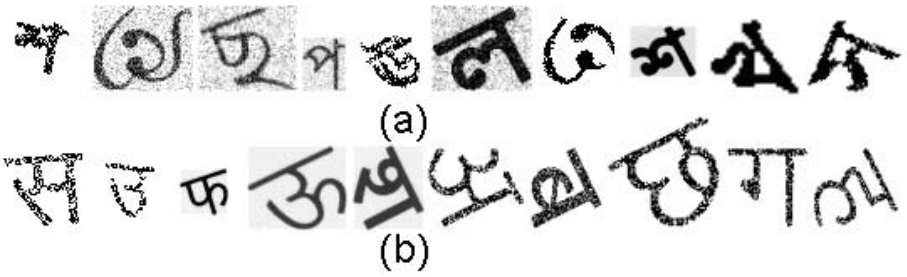


Figure 2.11: Examples of some sample data used in our experiment (a) Bengali and (b) Devnagari.

The recognition rates for all the test subsets were averaged to calculate recognition accuracy. This is done for each of character database, separately. Isolated character classification has been tested with different types of kernel functions: Linear, Polynomial (degree 3), RBF and Sigmoid functions. English character dataset has been used for this purpose. Since we obtained best recognition accuracy using RBF kernel (see Table 2.4), we use the same kernel for other experiments.

Table 2.4: Classification with SVM using different kernel functions.

Linear	Polynomial	Radial Basis Function	Sigmoid Function
99.41%	99.56%	99.61%	87.31%

For recognition result computation we used different measures and the measures are defined as follows:
Recognition rate = $N_C^R * 100/N_T^R$
Error rate = $N_E^R * 100/N_T^R$
Reject rate = $N_R^R * 100/N_T^R$
Reliability = $N_C^R * 100/(N_R^R + N_C^R)$
Where N_C^R is the number of correctly classified characters, N_E^R is the number of misclassified characters, N_R^R is the number of rejected characters and N_T^R is the total number of characters tested by the classifier. Here $N_T^R = N_C^R + N_E^R + N_R^R$.

Recognition results

From the experiment, we noted that the rotated character recognition accuracy of the proposed scheme was 99.61% for English, 98.86% for Bengali and 99.18% for Devnagari when 176 dimensional feature vector was used and no rejection was considered. Also, from the experiment we noted that 99.76% (99.82%) accuracy was obtained when first two (three) top choices of the recognition results were considered in English. Detail results of English, Bengali and Devnagari characters with different choices are shown in Table 2.5. From the Table 2.5 it can be noted that recognition accuracy increases by 0.73% when we considered two top choices instead of one top choice in Bengali. We analyzed the results and noted that the increment obtained

by two top choices instead of one top choice was mainly due to the presence of some similar shape characters.

Table 2.5: Recognition result based on different choices when no rejection was considered.

Number of choices from top	Accuracy with 0% rejection		
	English	Bengali	Devnagari
Only 1 choice	99.61	98.86	99.18
Only 2 choice	99.76	99.59	99.73
Only 3 choice	99.82	99.70	99.89
Only 4 choice	99.89	99.78	99.93
Only 5 choice	99.97	99.82	99.95

Rejection versus error and reliability rate computation

We computed character recognition results with different rejection rates. We noted that 99.45% (99.53%) reliability with 0.53% (0.43%) error was obtained when 1.58% (1.60%) rejection was considered in Bengali (Devnagari) characters of the proposed system. 99.84% (99.96%) reliability with 0.13% (0.04%) error was obtained from the system when 9.58% (9.30%) data were rejected in Bengali (Devnagari) characters. Recognition reliability with different rejection rates is given in Table 2.6 for Bengali dataset and in Table 2.7 for Devnagari dataset. Rejection was done based on the difference of the optimal likelihood values of the best and the second-best recognized character.

Table 2.6: Error and reliability results with respect to different rejection rates on Bengali dataset.

Rejection rate (%)	Error rate (%)	Reliability (%)
1.58	0.53	99.45%
2.50	0.35	99.63%
3.49	0.27	99.72%
5.10	0.21	99.77%
9.58	0.13	99.84%

Table 2.7: Error and reliability results with respect to different rejection rates on Devnagari dataset.

Rejection rate (%)	Error rate (%)	Reliability (%)
1.60	0.43	99.53%
2.52	0.28	99.71%
3.48	0.16	99.83%
5.06	0.09	99.90%
9.30	0.04	99.96%

Evaluation of different parameters

We computed character recognition results with different numbers of rings used for zoning the isolated characters. This experimental evaluation is done on English character dataset. We noted best recognition results from 7 (see Table 2.8) and hence we decided number of ring zones as 7.

Table 2.8: Recognition accuracy of English characters using different choice of rings.

4 rings	5 Rings	6 rings	7 rings	8 rings
96.37 %	98.84 %	99.28 %	99.61 %	99.61 %

The choice of the value of k is evaluated with different pair of values [(k=2,k=3), (k=3,k=4), (k=4,k=5), (k=5,k=6) and (k=6,k=7)]. The recognition accuracy is with different choices are shown in Table 2.9. We noted that average relative angle information obtained from the pair (k=3,k=4) showed best recognition results in our method and hence we considered k = 3 and 4.

Table 2.9: Recognition accuracy of English isolated characters using different value of 'k' (length of neighbor pixel).

k=2 & k=3	k=3 & k=4	k=4 & k=5	k=5 & k=6	k=6 & k=7
99.24 %	99.61 %	99.61 %	99.58 %	99.46 %

Recognition result with different combination of features

In Table 2.10, we show the results of recognition accuracy on the combination of circular rings, convex hull rings and angular slope. We have performed this comparative study on the dataset of English character from graphical documents. The size of the dataset is 8250. We have obtained highest 98.89% accuracy with 176 dimensional feature vector. Different combination of feature are shown in Table 2.10.

Table 2.10: Recognition result of English characters with different combination of angular information.

Feature Descriptor	Circular Ring (CR)	Convex Hull Ring (CHR)	Angular slope (AS)	CR + CHR	AS + CR	AS + CHR	AS + CR + CHR
With 5 Rings (Feature Dim.)	95.50 (8×5)	93.88 (8×5)	97.42 (8×8)	96.52 (40+40)	98.74 (64+40)	98.44 (64+40)	98.82 (64+40+40)
With 7 Rings (Feature Dim.)	95.87 (8×7)	94.24 (8×7)	97.42 (8×8)	96.93 (56+56)	98.81 (64+56)	98.56 (64+56)	98.89 (64+56+56)

Error Analysis

From the experiment we noticed most of the errors occur because of the similar structural shape of some of the characters in Bengali/Devnagari script. We also noticed the main confusing character pair from our experiment and their error rates are shown in Table 2.12(a) and Table 2.12(b) corresponding to Bengali and Devanagari characters. The Bengali characters in the first row of Table 2.12(a) confused maximum times between them and their confusion rate was 0.127% (1.830%) when it is computed from overall (within two-class) samples. The next most confusing pair was the samples from 2nd row of Table 2.12(a), and they confused 0.076% (1.609%) cases. In Devnagari, character samples from 1st row of Table 2.12(b) confused maximum and their confusion rate was 0.146% (2.926%) when it is computed from overall (within two-class) samples.

Results on poor/noisy image

Since our feature detection technique is not very sensitive to noise, our scheme shows good results even if the samples are poor in quality. To get an idea of such samples, some noisy images used in our system are shown in Fig.2.13. Our system recognized all the samples of Fig.2.13, correctly.

Comparison of results

To get an idea about the comparative results of recognition we compare our results with some of the work done for English and Chinese printed characters (see Fig.2.14). The method due to Xie and Kobayashi [XK91] was tested on ten English numerals and obtained 97% accuracy from the numerals. Adam et al. [AOC$^+$00] obtained 97.5% accuracy on English characters. In another work [PKRP06] using modified quadratic discriminant function (MQDF) for English rotated character recognition it is obtained 98.34% accuracy from the system. We also tested the same data used in [PKRP06] and we obtained 99.61% accuracy from this current approach. Yang and Wang [YW01] reported 97.40% accuracy from Chinese characters. Our proposed method showed 98.86% (99.18%) accuracy for Bengali (Devnagari) script when tested on 7874 Bengali and 7515 Devnagari data. To the best of our knowledge, there is no published work related to this proposed work on any Indian languages.

To get the idea of comparative recognition results of different rotation invariant descriptors, we present a comparative study of different descriptors like (a) Angular Radial Transform (ART) [RCB05], (b) Hu's Moments [Hu62], (c) Zernike moments [KH90a] and (d) Fourier-Mellin [AOC$^+$00], and we tested them on same datasets. These descriptors are used due to their invariance to rotation, translation and scale. We used three datasets of Bengali, Devnagari and English isolated text characters for comparison. A common platform is used before using the feature descriptors to the dataset of Bengali, Devnagari and English characters. This is done by applying same pre-processing algorithm to the whole dataset to reduce the noise effects. The feature vectors, obtained from different descriptors are passed to a SVM classifier to get the recognition accuracy. The classification is done in 5-fold cross-validation way. In Fig.2.14 we show the percentage of accuracy (lower part of the figure) obtained from

Confusing character pairs		Confusion rate	
		(overall)	Within two-class
ব	ব	0.127	1.830
ঘ	ঘ	0.076	1.609
ঞ	ঞ	0.038	1.027
ড়	ড়	0.038	1.266
ধ	ধ	0.038	0.955

(a)

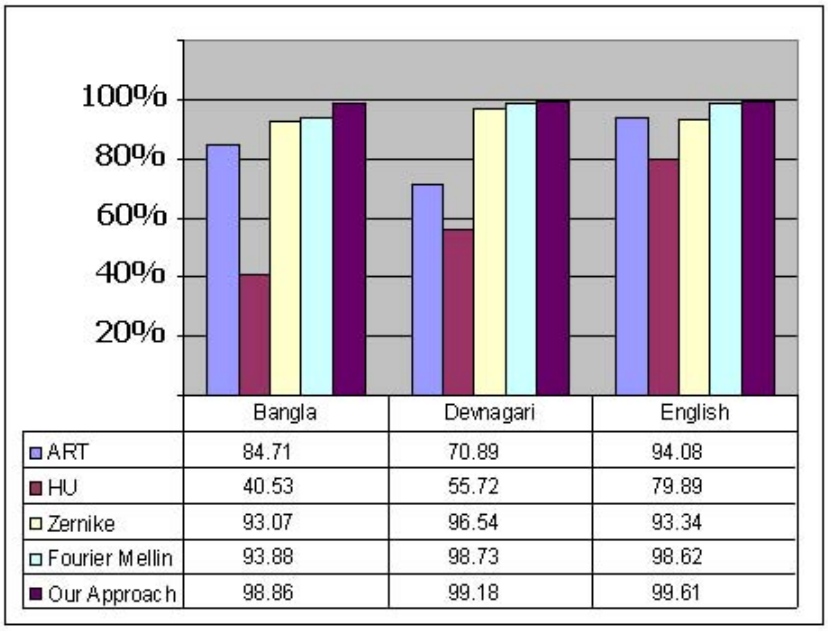
Confusing character pairs		Confusion rate	
		(overall)	Within two-class
ओ	ओ	0.146	2.926
फ	फ	0.041	0.811
ध्र	ध्र	0.040	0.993
क्ल	क्ल	0.027	0.544
म	म	0.027	0.725

(b)

Figure 2.12: Confusion character pairs in (a) Bengali and (b) Devnagari.

different descriptors. For easy observation of the results, a bar chart of the results is also shown in the upper part of Fig.2.14. From the Fig.2.14 it can be seen that Hu's moments shows the lowest recognition accuracy. For Bengali characters it gave only 40.53% accuracy. We noted that our approach shows the best performance in all the three scripts. Fourier-Mellin performs much closed to our approach in English and Devnagari but is less accurate in Bengali text characters (about 4.98%) due to its high intra shape similarity.

Figure 2.13: Examples of some noisy images from isolated characters of Bengali (1st row) and Devnagari (2nd row).



	Bangla	Devnagari	English
ART	84.71	70.89	94.08
HU	40.53	55.72	79.89
Zernike	93.07	96.54	93.34
Fourier Mellin	93.88	98.73	98.62
Our Approach	98.86	99.18	99.61

Figure 2.14: Comparative recognition results obtained from different descriptors (when tested on the same data set).

2.4.2 Experiment with MPEG7 dataset

To better assess the descriptive power of our "Angle-based" feature descriptor, we have evaluated with MPEG-7 shape silhouette database, where images contain high inter-class variability in terms of scale, rotation, rigid and elastic deformation [Bis09]. The database consists of 1400 images: 70 shape categories, 20 images per category. We show different images of this database in Fig.2.15. The comparison of classification accuracy with different feature descriptor is given in Table 2.11. The best performance

is obtained by our Angle-based approach (90.78%).

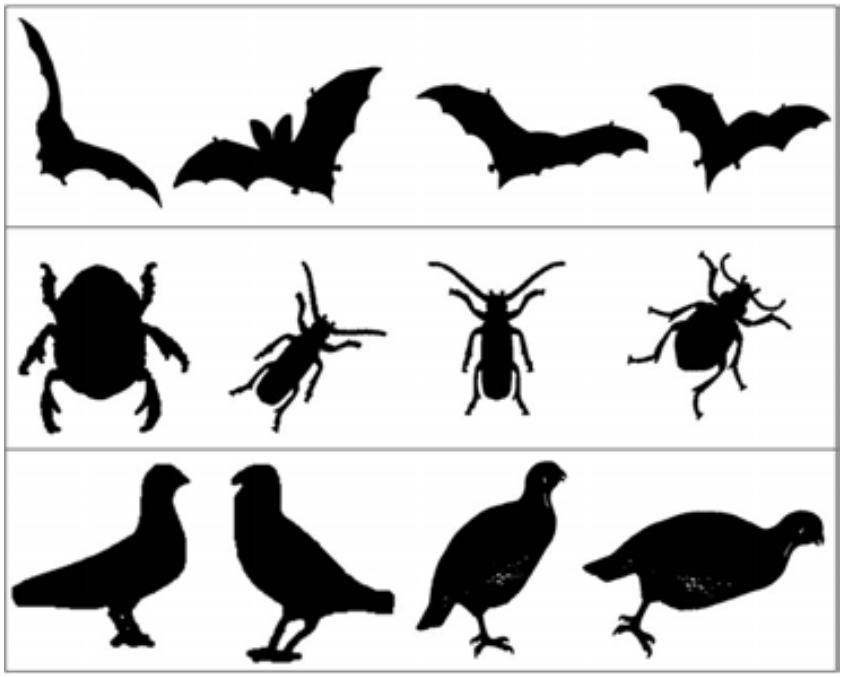


Figure 2.15: Example of shapes in the MPEG-7 database for 3 different categories.

Table 2.11: Comparison of classification accuracy on the 70 MPEG-7 symbol categories.

ART	HU	Zernike	Fourier Mellin	Angle based Approach
66.07	39.64	65.64	71.5	90.78

2.4.3 Experiment with Handwritten Music Symbols

Finally, we have tested our feature with handwritten music symbols [Bis09]. The dataset of music clefs was obtained from a collection of modern and old musical scores (18th and 19th centuries) of the Archive of the Seminar of Barcelona. The database contains a total of 4094 samples in 7 different class categories. We show some images of this database in Fig.2.16. The comparison of classification accuracy with different feature descriptor is given in Table 2.12. The best performance is obtained by our Angle-based approach (99.31%).

Figure 2.16: Example of different hand-drawn musical symbols.

Table 2.12: Comparison of classification accuracy on the handwritten symbol categories.

ART	HU	Zernike	Fourier Mellin	Angle based Approach
95.94	72.12	88.98	92.35	99.31

2.5 Conclusion

In this chapter, we have presented a novel shape descriptor feature based on angle information for isolated character recognition in graphical documents. The proposed system does not depend on the skew of the document and handles multi-oriented text as well as normal text. Size and rotation invariant features are considered for the recognition of multi-sized/multi-oriented characters and the features are computed from different angular information obtained from the contour pixels of the characters. Angular information is computed in such a way that it does not depend on the size and the rotation of the characters and they make the feature rotation invariant. Circular and convex hull ring have been used to divide a character into smaller zones to get more local features of the shape.

The approach has been evaluated on printed character datasets of different scripts (English, Bengali and Devnagari). Different state-of-the-art descriptors are compared, showing the robustness and performance of the proposed scheme when classifying symbols with high variability of appearance (fonts), rigid or elastic deformations, and noise. We have also compared our descriptor with MPEG-7 dataset and handwritten music symbols for its competence. The experimental results show that the angle based approach better represents the symbols.

As mentioned in Introduction chapter, performance of OCR is affected mainly due to touching characters present in document. We need a segmentation approach which can take care of such touching components in graphical documents. In the next chapter we will propose a touching character segmentation approach in multi-scale and multi-orientation environment. Multi-oriented isolated character recognition, presented in this chapter, will be used for generating the recognition confidence to obtain such segmentation.

Chapter 3

Multi-Oriented Touching Text Character Segmentation using Dynamic Programming

In the last chapter, we have discussed isolated character recognition in graphical documents. However, due to noise, text characters do not appear isolated always in such graphical documents. Characters in string touch each other and such touching character segmentation is a major problem of achieving higher recognition rates by OCR. In this chapter, we present a scheme towards the segmentation of English multi-oriented touching strings into individual characters. When two or more characters touch, they generate a big cavity region at the background portion. Based on convex hull information, at first, we use this background information to find some initial points to segment a touching string into possible primitives (a primitive consists of a single character or part of a character). Next, some of these primitives are merged to get optimum segmentation. Dynamic programming algorithm is applied for this purpose using the total likelihood of characters as the objective function. Experiments are performed in databases of real and synthetic touching characters. The results show that the method is efficient in segmenting touching characters of arbitrary orientation and size.

3.1 Introduction

As discussed in Chapter 1, electronic media becomes more and more accessible, the need for transferring off-line documents to the electronic domain grows. Optical Character Recognition (OCR) works by scanning source documents and performing character analysis on the resulting images giving a translation to ASCII text which can then be stored and manipulated electronically like any standard electronic document. As part of the OCR process, character segmentation techniques are applied to word images before individual characters images are recognized. The simplest way to perform character segmentation is to use the small space between characters as

segmentation regions. This strategy fails when there are touching or broken characters, which often occur in degraded text images. Some examples of such documents are photocopies, faxes, historical documents, etc. which are often degraded due to compression, bilevel conversion, ageing or poor typing [CW00, SLLC05]. In these situations, two or more characters may be segmented as one character image or one character image may split into multiple pieces. Due to degradation of small inter-character space, adjacent characters in a string touch together and they share common pixels in touching regions [SSR10].

3.1.1 Motivation

In graphical documents such as maps, engineering drawings, etc. or artistic documents, text lines appear frequently in different orientations rather than usual horizontal direction. The purpose of such orientation and curvi-linearity is to catch people's attention at some particular words/lines or to annotate the location of graphical objects. Thus, a single document may contain strings with different inter-character spacing in the strings due to the annotation, style, etc. Also, the curvi-linear nature of the text line makes the orientations of characters in a string different. As a result, it is difficult to detect the skew of such strings and hence character recognition of such documents is a complex task [RPLK08].

Segmentation of touching components is one of the difficulties to get higher recognition rates by OCR systems. The OCR systems available commercially do not perform well when documents are rotated or the words are multi-oriented. When touching occurs in multi-oriented documents (e.g. artistic or graphical documents), it is much more difficult to segment such multi-oriented touching than touching segmentation of normal horizontal touching. Touching in curvi-linear string leads to false character segmentation and hence wrong recognition result occurs.

Text-lines could appear at different directions in a same document as illustrated in Fig.3.1. It can be seen from Fig.3.1(a), the word "PLANET" contains a touching string "LANE" of four characters. In Fig.3.1(b), we show a map where many characters in the word "Mayurakshi" are touching and they are oriented in different directions. Orientation of two touching strings "ON" and "RE" of Fig.3.1(c) are perpendicular to each other. In Fig.3.1(d), it may be noted that orientations of "es" and "no" in the word "Couesnon" are not the same and such strings create difficulty for segmentation.

3.1.2 Related Work

There are many published papers towards the recognition and segmentation of the touching characters of horizontal direction [CC99, LCSS99, KKS00, VOB$^+$08] and they are briefly reviewed here.

Among the earlier pieces of work on touching character segmentation, one class of approaches uses contour features of the connected components for segmentation [KKS00, SS95, Fen91]. When analyzing the contour of a touching pattern, valley and crest points are derived. Next, a cutting path is decided to segment the touching pattern by joining valley and crest points. Kahan et al. [KPB87] used projection

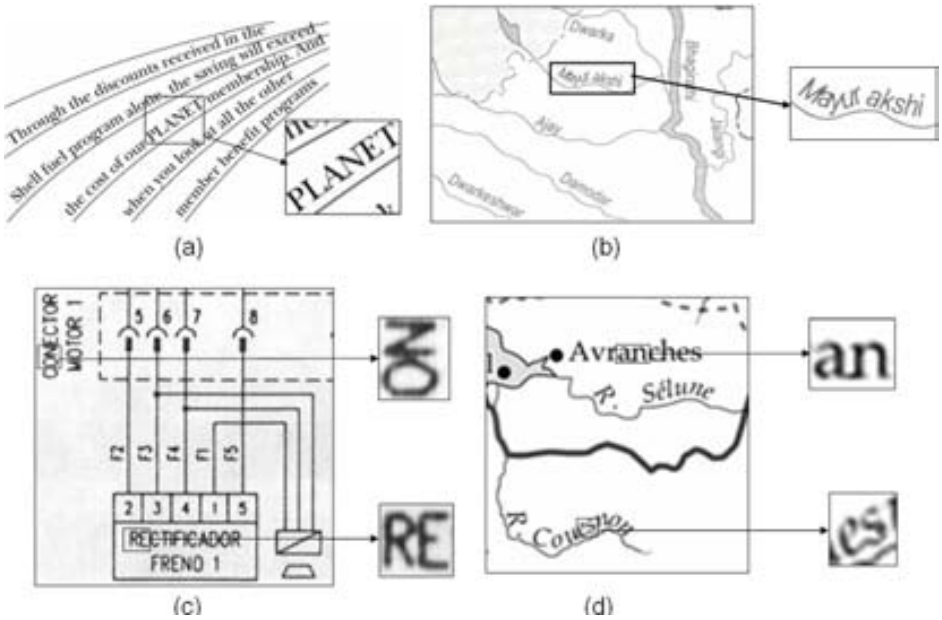


Figure 3.1: Example of documents showing multi-oriented touching characters in (a) an advertisement, (b) and (d) maps, and (c) electrical diagram.

profiles as the objective function for touching character segmentation. They used the idea of joining adjacent characters that have minimum vertical projection. The segmentation function is calculated from the ratio of the second derivative of the projection-profile curve to its height. Later, Lu et al. [Lu95] introduced a peak-to-valley function to improve the segmentation approach. Fujisawa et al. [FNK92] used profile features for touching numeral segmentation. Upper and lower profiles of the connected component are computed and the distance between upper and lower profiles is analyzed to detect the segmentation points.

Afterwards, Liang et al. [LSA94] proposed discriminating functions for machine printed touching character segmentation. Pixel projection and profile projection techniques are employed as discriminative functions to solve heavily touching printed characters. Next, they applied forward segmentation along with a backward merge procedure based on the output of a character classifier. It works on the components generated by discriminating functions. Yu and Yan [YY98] presented a segmental technique using structural features for single-touching hand-written numeral strings. At first, the touching region of the character components is determined based on the structural points. Next, a candidate touching point is pre-selected using the geometrical information of special structural points. Finally morphological analysis and partial recognition results are used for the purpose of segmentation. Dimauro et al. [DIP92] applied contour based features along with some descending algorithms for the touching character segmentation purpose.

Another class of approaches is based on thinning [CW00, LCSS99]. In this ap-

proach, thinning of foreground and/or background pixels of the connected pattern are processed. End and fork points obtained by thinning are used for cutting points extraction. These methods are time consuming and in addition they generate protrusions. These protrusions sometimes give wrong results because they bring some confusions among the actual fork and end points.

A water reservoir based technique [PBC03] is employed to locate inter-character spaces in touching numeral strings. Water reservoir is a metaphor to illustrate the cavity region of a component. In this sense, if water is poured from a side of a component, the cavity regions of the background portion of the component where water will be stored are considered as reservoirs of the component. Based on the size of water reservoirs, the segmentation zones of the touching string are selected. Next, segmentation is done using structural information of these reservoirs.

Recently, Yong et al. [YYZ09] proposed an approach using supervised learning on the labeled examples and a Markov Random Field (MRF) approach has been applied for this purpose. Further, a propagation minimization method is employed to select the candidate patches based on the compatibility of the neighbor patches. The output of the MRF after the iterative belief propagation forms a segmentation probability map. Finally, the cut position is extracted from the map. An accuracy rate of 94.8% is reported.

Methods based on combination of features have also been used for the touching segmentation. Oliveira et al. [OLBS00] used contour, profile and skeleton features to find a set of points for touching characters segmentation. First, local minima of contours and profile features are defined as basic point (BP). Second, a point with more than two pixels in its neighborhood is defined as intersection point (IP). Afterwards, a Euclidean distance scheme is applied to determine proximity between IP and BP for segmentation.

As discussed above, a number of methods for treating the problem of segmentation in character recognition have developed remarkably in the last decade. In Table 3.1, we summarize some state-of-the-art approaches for text character segmentation algorithms.

Table 3.1: Related work proposed on touching character segmentation.

Method	**Key Feature in the Proposed Algorithm**
Fujisawa et al. [FNK92]	Upper and lower profiles of characters
Chen et al. [CW00]	End and fork points obtained from thinned image
Yu and Yan [YY98]	Structural features
Pal et al. [PBC03]	Feature based on water reservoir concept
Yong et al. [YYZ09]	Supervised learning using MRF
Oliveira et al. [OLBS00]	Contour, profile and skeleton features

The state-of-the-art approaches of touching character segmentation generally consider touching of characters in horizontal text strings. These methods assume the characters of strings are aligned horizontally and thus segmentation features are devised for such characters in horizontal strings. The features used in most of the approaches for text character recognition are generally not rotation invariant. In

graphical documents when characters touch, it is difficult to know the angle of alignment of characters in the touching regions. The characters along a touching portion may be in different orientations with respect to the baseline of the word. In Fig.3.1(b), we showed some examples of different orientations of such touching. Due to their complex structure, methods using orientation based matching may not be useful to segment them properly. For segmentation purpose, we need text character feature that can take care size and rotation invariant string. Hence, we propose here a segmentation approach to deal with characters in multiple orientations.

3.1.3 Outline of the Approach

Although many techniques are available for segmentation of horizontal touching characters, to the best of our knowledge there is no work towards multi-oriented touching character segmentation. In this chapter, we propose an approach for multi-oriented character touching string segmentation scheme. Though our objective is to segment n-touching component into its corresponding characters, we will also provide a brief details of touching character segmentation when number of components present in a touching component is known. This idea is proposed with the number of characters being two in the touching component.

When two or more characters touch, they generate a big cavity region at the background portion. We use this background information to find the segmentation point in our method. To handle the background information the convex hull is used. In the proposed scheme of 2-character segmentation, at first, a set of initial segmentation points is predicted based on the concave residua of the convex hull of a touching string using Douglas Peucker approximation. Using these initial points, we determine some candidate segmentation lines to segment a touching component into 2 parts. The recognition confidence of the two sub-images of a touching string, obtained from each candidate segmentation line, is computed and the candidate segmentation line from which we get optimum confidence is the actual segmentation line.

When the number of characters is not known for a touching component, i.e. for n-character touching component, the touching string is segmented first into primitive segments based on the initial segmentation points. A primitive segment consists of a single character or a part of a single character. Next, the primitive segments are merged to get optimum segmentation and dynamic programming is applied using total likelihood of characters as the objective function.

In Chapter 2, we have detailed about shape descriptor and classification approach. A feature descriptor of size 176 dimension was computed using angle based approach. Next, SVM classifier had been used to build the character shape model from multi-size and multi-oriented feature of our training data. SVM generates separate class-model for each character class with the training dataset. Given a set of primitive segments, we compute the recognition confidence to obtain the corresponding class and use this value as the cost function in our dynamic programming approach.

The rest of the chapter is organized as follows. Proposed segmentation approach for two touching characters is discussed in Section 3.2 and for n-touching characters in Section 3.3. The experimental results are discussed in Section 3.4. Finally conclusion is given in Section 3.5.

3.2 Segmentation of 2-Touching Characters

Recognition of individual characters in multi-oriented and multi-sized environment drives the segmentation hypothesis of touching characters in our system. We assume, the number of text characters in the touching component is known and the number is 2. Segmentation of the 2-characters touching string is done based on the recognition confidence of different segmented pairs obtained from the string. The block-diagram of our approach is shown in Fig.3.2. Details of the segmentation method are discussed below.

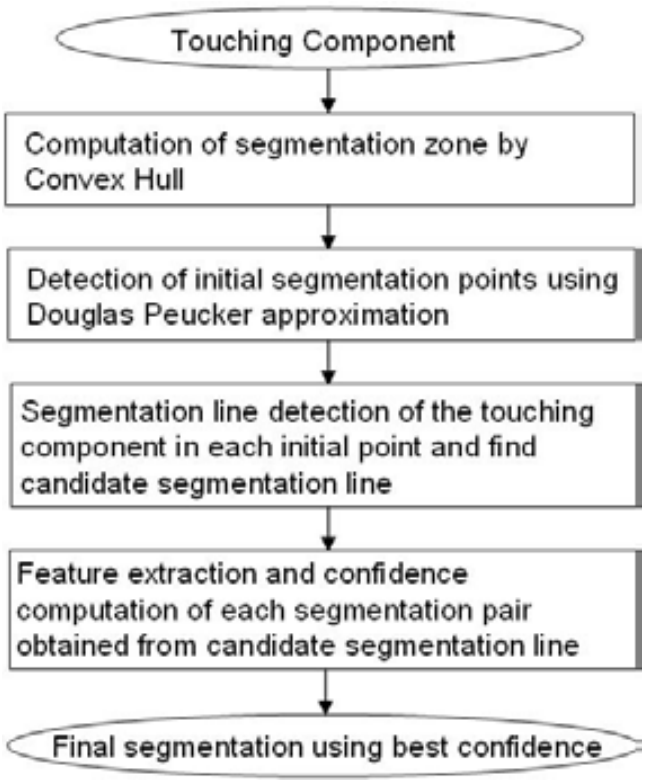


Figure 3.2: Block diagram of our character segmentation approach.

3.2.1 Computation of Segmentation Zones

When two or more characters touch, generally they generate big cavity regions in the background portion between touching components. If components in a string are in horizontal direction, the water reservoir concept can be used to find these cavity regions [PBC03]. Water reservoir based algorithms may not be applied due to the multi-oriented nature of the strings. We have considered some properties of convex hull to take care of this problem.

Convex Hull and Properties of Concave Residua

As mentioned in Chapter 2 (Section 2.2.2), a convex hull is a minimal convex shape entirely bounding an object. For our character segmentation approach we define some properties of residuum of convex hull which are as follows.

1. Residuum area (R_A): The area of a residuum is defined by the number of pixels inside the residuum.
2. Residuum surface level (R_{SL}): The R_{SL} of a residuum is the line obtained by joining two endpoints of the open face of the residuum.
3. Residuum border pixels (R_{BP}): The border pixels of each residuum are defined as the contour pixels of the residuum excluding the R_{SL} pixels.
4. Residuum height (R_H): It is the depth of the farthest residuum border pixel from R_{SL}.

In Fig.3.3, convex hull residua and their different parameters for a text character 'S' are shown.

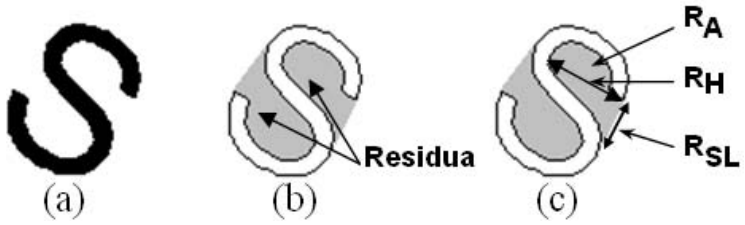


Figure 3.3: (a) Image of the character 'S'. (b) Two residua from the convex hull of 'S'. (c) Different parameters of convex hull are shown in a residuum.

Segmentation Zones

For a touching component image, the cavity regions are determined by finding residua of that component through convex hull. These residua cover the cavity regions of the touching component and thus, they are used to determine the segmentation zone of touching characters. The residua found from the convex hull of a touching character are shown in Fig.3.4. For this touching component '72', we find a total of four cavity regions.

Given a touching component, we may find many small cavity regions along with the segmentation zones due to the degradation of contour of the characters. Even, the presence of "Serif" in some fonts of text characters (e.g. Times New Roman) also produces small cavity regions. These small cavity regions are considered as noise and thus, these regions are not considered for segmentation. To do that, the residua having height more than stroke-width are taken care for segmentation purpose. The stroke-width (St_w) of the word is the statistical mode of object pixels' run lengths [CC99]. For a component, St_w is calculated as follows. The component is scanned

row-wise (horizontally), column-wise (vertically) and then in two diagonal directions (45° and 135°). If rl different runs of lengths $r_1, r_2 \ldots r_{rl}$ with frequencies $f_1, f_2 \ldots f_{rl}$, respectively are obtained by the scanning the component, then value of St_w will be r_i if $f_i = \max(f_j)$, j = 1, 2 $\ldots rl$.

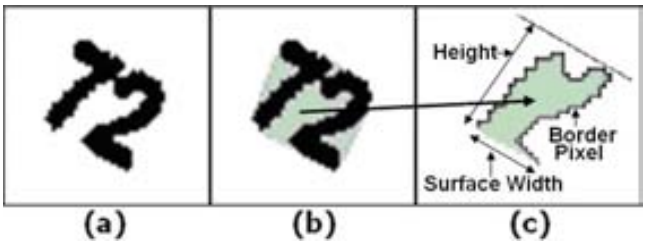


Figure 3.4: (a) Touching character. (b) The residua of the convex hull are shown and marked by grey color. (c) Different parameters of the residuum.

3.2.2 Initial Segmentation Point Detection

To segment a touching string into possible primitives, segmentation points are next computed from the segmentation zones extracted previously. To do it, we employ a polygonal approximation method to the contour pixels of residuum borders. Polygonization provides key-points which are at the corner of edges in the corresponding segmentation zones. There are different algorithms for reducing the points in a polyline to produce a simplified polyline that approximates the original within a specified tolerance. For example Douglas-Peucker algorithm, Vertex Reduction algorithm, etc. Most of these algorithms work in any dimension since they only depend on computing the distance between points and lines. Among existing algorithms of the literature [Kol03], we have selected the Douglas-Peucker [DP73] polygonal approximation algorithm. This algorithm is well adapted to localize hard curvature points along a border. A short presentation of this algorithm is given as follows.

Polyline Approximation : Douglas-Peucker Algorithm

The classical Douglas-Peucker [DP73, HS92] line-simplification algorithm is recognized as the one that delivers the best perceptual representations of the original lines. It works from top down by starting with a crude initial guess at a simplified polyline, namely the single edge joining the first and last vertices of the polyline. Then the remaining vertices are tested for closeness to that edge. If there are vertices further than a specified tolerance, $\epsilon \geq 0$, away from the edge, then the vertex furthest from it is added to the simplification. This creates a new guess for the simplified polyline. Using recursion, this process continues for each edge of the current guess until all vertices of the original polyline are within tolerance of the simplification.

More specifically, in the Douglas-Peucker algorithm, the two extreme endpoints of a polyline are connected with a straight line as the initial rough approximation of the polyline. Then, how well it approximates the whole polyline is determined by

computing the distances from all intermediate polyline vertices to that (finite) line segment. If all these distances are less than the specified tolerance ϵ, then the approximation is good, the endpoints are retained, and the other vertices are eliminated. However, if any of these distances exceeds the ϵ tolerance, then the approximation is not good enough. In this case, we choose the point that is furthest away as a new vertex subdividing the original polyline into two (shorter) polylines. The Fig.3.5 shows a few steps for obtaining approximated polyline.

For a polyline shown in Fig.3.5(a), the contour pixel (V_t) has the maximum distance from the line joining the farthest points (V_1 and V_n) of the polyline. Next, the polyline is simplified with lines V_1V_t and V_tV_n as shown in Fig.3.5(b). In Fig.3.5(c), this process is iterated in polyline segment V_tV_n and V_u is selected having the maximum distance between points V_t and V_n. For simple 2D planar polylines the time complexity is *O(nlogn)*.

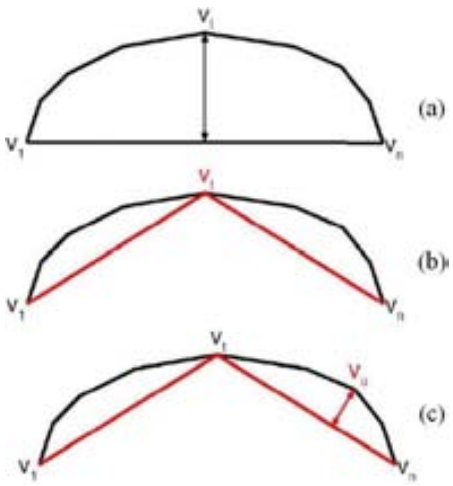


Figure 3.5: Stages of Douglas-Peucker for Polyline Approximation.

Initial Segmentation Points

The two end points of residuum surface level (R_{SL}) of each selected residua are regarded as the initial rough estimation of the polyline. Using this initial guess, the other vertices are approximated using a tolerance threshold ϵ. The value of this tolerance threshold is selected with stroke width (St_w) precision. After approximation, the list of vertices are treated as key-points. The advantage of using polygonal approximation is that it provides the key-points which are at the corner of edges in the corresponding segmentation zones. These key points are necessary for touching character segmentation because, usually when the characters touch they form a corner in the touching region.

We have noted that some of the key-points might appear very near to R_{SL} due to the appearance of hard curvature or degradation in the contours of these zones. These key-points do not provide the segmentation lines and thus are selected for removal.

This is because, we assume the touching position between characters is closed to deepest point from that residuum surface level. In Fig.3.6, the initial segmentation points are shown for the image Fig.3.4(a).

Figure 3.6: Initial segmentation points (black dot) found from concave residua after using polygonal approximation are shown on a touching component.

For the segmentation purpose, a set of lines were determined from the residuum contour area. Each of these segmentation lines segment the touching component into 2 parts.

3.2.3 Computation of Candidate Segmentation Lines

Once we get initial segmentation points, for each initial segmentation point (V_i^S) we find another point (V_j^S) through which the touching component can be cut into 2 parts. Computation of V_j^S is done as follows.

We know the angle of the direction of the R_{SL} with the horizontal axis. For each initial segmentation point, we find a segmentation line passing through this point and perpendicular to the corresponding R_{SL}. This line segments a touching component at the point V_i^S. The perpendicular angle to the R_{SL} generally gives us a clue to the direction of the segmentation line, hence we used it here. We draw a line from the point V_i^S in the opposite side of R_{SL} and perpendicular to the R_{SL} until it passes through the object pixels. Let, the last object pixel on this line be V_c^S. The line from V_i^S to V_c^S may be considered as segmentation line. This segmentation line may not give the best segmentation always and hence to get better segmentation the point V_c^S is tuned to have a better segmentation point. This tuning is done by considering some neighbor contour points of the point V_c^S. To get neighbor pixels, the contour is traced upto a length of stroke-width (St_w) in clockwise and anti-clockwise direction starting from V_c^S. These traced pixels are neighbor pixels. The neighbor pixel having the minimum distance from V_i^S is chosen as V_j^S. The line obtained by joining V_j^S and V_i^S is the segmentation line.

Thus, for every initial segmentation point, we have the respective segmentation line. But, to reduce the computation complexity, we remove some segmentation lines which divide the image in two very un-equal sizes. The segmentation lines that segment the touching component into two parts of similar sizes are considered for future consideration. Size similarity is done by finding the minimum enclosing rectangle (MER) of the segmented sub-images. Let l_1 be the length of larger side of MER of one sub-image and l_2 be the corresponding length of other sub-image. If

$\max(l_1, l_2) \geq 1.5 \times \min(l_1, l_2)$, then we ignore such segmentation lines. The remaining segmentation lines are considered as candidate lines. Candidate segmentation lines of Fig.3.4(a) have been shown in Fig.3.7. From this set of candidate segmentation lines, we will choose the best line using recognition confidence obtained by SVM and that line will be the segmented line of the touching component.

Figure 3.7: Four Candidate segmentation lines obtained from segmentation points are shown.

To get an idea about the number of candidate segmentation line for each touching component, we computed total number of segmentation line in our dataset. We noted that on an average, we obtained 2.81 segmentation lines for a touching string.

3.2.4 Selection of Best Segmentation Line

The best segmentation line will be selected by recognition confidence, computed by SVM. This is done as follows. For each segmentation line we will have two sub-images which are formed by dividing the two-character touching string by the segmentation line. Two sub-images are tested for their recognition confidence by SVM (discussed in Chapter 2). Based on the recognition result of a sub-image, our SVM generates a value (between 0 and 1) for each sub-image and this value is the confidence for that sub-image. We add the confidence of these two sub-images to get the total confidence. When a line separates a touching character at the proper location, we generally get highest confidence value from the sub-images of the string.

For each segmentation line, we compute the confidence value of two segmented parts and use the cumulative confidence value as the cost function to the corresponding segmentation line. A rejection threshold (TH_{rej}) is included to achieve good segmentation performance. If the cumulative confidence value is less than TH_{rej}, we do not consider that touching component for that segmentation. Character segmentation accuracy with different rejection threshold is discussed in Table 3.3. The segmentation line for which we get the highest confidence result determines the final segmentation line. In Fig.3.8, we have shown the final segmentation result of the touching character of Fig.3.4(a). For better understanding of the proposed touching character segmentation technique, algorithm steps of the scheme are provided in Algorithm 1.

Figure 3.8: Image shows final segmentation line.

Algorithm 1 Segmentation of 2-touching character

Require: Touching component (C_T)
Ensure: Segmentation line detection to separate C_T into characters
Compute Convex Hull of C_T and find the residua.
//create a list (SL) of segmentation line with their confidence
$SL \Leftarrow \oslash$
for all residua R_i of C_T **do**
Generate the initial segmentation points ($V_1^S \dots V_n^S$) using Douglas Peucker polygonal approximation in the contour of R_i
for all initial segmentation points V_j^S **do**
Compute Segmentation Line (L_{ij}^S) at V_j^S perpendicular to the R_{SL} of R_i and detect whether L_{ij}^S is candidate segmentation line or not
if L_{ij}^S is candidate segmentation line **then**
Segment C_T into two sub-components (C_{Lij}^1 and C_{Lij}^2) along L_{ij}^S
Compute multi-oriented character recognition confidences G_{Lij}^1 and G_{Lij}^2 corresponding to C_{Lij}^1 and C_{Lij}^2
$G_L ij = G_{Lij}^1 + G_{Lij}^2$
Store the 2-tuple vector (G_{Lij}, L_{ij}^S) into SL
end if
end for
end for
Select the 2-tuple (G_L^{max}, L_{max}^S) having maximum confidence from SL
Consider L_{max}^S as the Segmentation Line of C_T

3.3 Segmentation of n-Touching Characters

Continuation with the idea of 2-character touching segmentation, a touching component of n-characters could be segmented if the number of character in the touching component be known before. This concept is restricted because the number of characters in the touching component has to be known a priori. It is a hard constraint, since it is not always possible to know the number of characters in a real touching component in multi-scale and multi-rotation environment. To solve this problem, we propose an optimization algorithm to split n-character touching components. The proposed algorithm not only determines the optimal cutting points to segment indi-

vidual characters but also uses the number of characters in the component (n) that maximizes a score function based on character recognition. The block-diagram of our approach is shown in Fig.3.9.

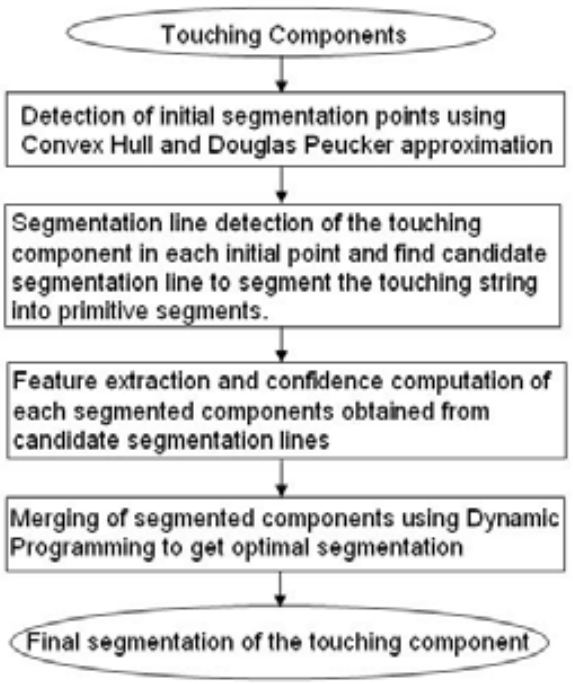


Figure 3.9: Block diagram of proposed character segmentation approach.

3.3.1 Touching Component Detection

There may exist touching or non-touching characters in a word. A component is detected as touching or isolated before applying the segmentation approach on touching string. For this purpose, at first, a Connected Component (CC) labeling is applied to extract individual components of the word (See Chapter 4). For each component, we compute the recognition confidence for all character class models using SVM and rank their confidence scores in descending order. If a component is recognized by the SVM with a high accuracy, we assign it as a non-touching or isolated character. If the difference between the top two recognition scores of a component is high, it is also considered as non-touching character. The rest of the components are considered as touching. These touching components are processed for segmentation using our approach. We may get some false positive labelling due to such separation based on confidence score. For example, the difference between the top two recognition scores may be less for characters like 'D', 'O'. Such mislabelled components do not affect the final segmentation result because the proposed dynamic programming approach takes care such error with the final optimal score.

3.3.2 Estimation of text-line orientation

If a n-character string or word is rotated to a certain angle, we estimate a rough angle from the minimum rectangular bounding box of the string. This rough inclination angle is used to arrange the primitive segments of the touching component in a sequential order, such that the next steps of our algorithm will be applied to merge some of them. We compute the bounding box of the word and find the angle (α) of the major axis with the horizon. The more are the numbers of characters in a word, the better is the approximation of the angle. An approximate height of the word (H_w) is found from its bounding box (See Fig.3.10). It is to be noted that, when there are few characters in the word and the characters have ascenders and descenders, α indicates an approximated angle of the inclination of the shape. In Fig.3.10, we show a multi-oriented word along with its bounding box. α and H_w are marked in this figure. It can be noted that, if this word is rotated by α then all the components of the word will not be in horizontal mode. Hence, existing approaches of horizontal touching character segmentation can not be applied in such string after rotating it by α.

Figure 3.10: A multi-oriented word and its bounding box are shown. α indicates the angle of inclination and H_w is the height of the bounding box.

3.3.3 Primitive Segmentation

For each touching component, we compute the segmentation lines from segmentation zones as discussed in Section 3.2. In short, we find the residua in the background of the touching string. Segmentation points are calculated in these residua by polygonal approximation of the residua contour. For each initial segmentation point in segmentation zones, we find a segmentation line passing through this point. From a segmentation point, we compute a line which is perpendicular to the corresponding residuum surface label and that perpendicular line segments the image into parts. The line is tuned by checking the neighborhood contour pixels in which the length of the segmentation line is small. This is discussed in details in Section 3.2.3.

It may happen that the touching portion of the components creates segmentation zones on both sides (top and bottom) of the characters. Such touching creates multiple segmentation points hypothesis in both sides. These points generate segmentation lines for the components. As a result, some of the segmentation hypotheses may lie very close to each other. To reduce the choices of hypothesis we remove some of the lines which are very closed. If the distance between two segmentation lines is less than St_w, the bigger segmentation line is not considered for segmentation. Moreover, if the length of a segmentation line is greater than $0.75 \times H_w$, that segmentation line

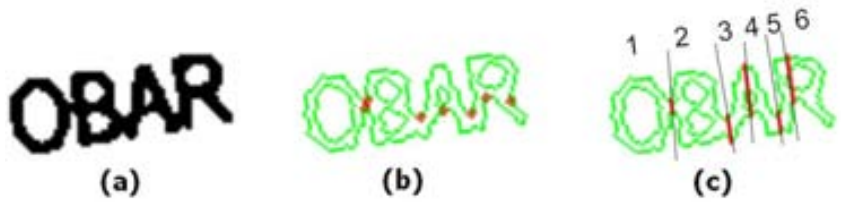


Figure 3.11: (a) Touching string. (b) Initial segmentation points found from concave residua. (c) Candidate segmentation lines obtained from selected segmentation points.

is also not considered. This value is set up from the experimental result.

For each initial segmentation point we get the corresponding segmentation line. If we have n segmentation lines, the image is segmented in $(n+1)$ sub-images. The (candidate) segmentation lines of Fig.3.11(a) have been shown in Fig.3.11(c). Note that, there were 7 initial segmentation points (See Fig.3.11(b)) and we have got 5 segmentation lines. These segmentation lines split the touching component into six primitive segments (see Fig.3.11(c)). These primitive segments are arranged in a sequence in a direction of angle α. Now, we will merge some primitive segments for correct segmentation. Dynamic programming technique is used for the purpose.

3.3.4 Merging of Primitive Segments by Dynamic Programming

The core of a dynamic programming (DP) algorithm [KTYS94] is the module that takes a set of symbols (list of primitive segments in our case) and a set of labels (possible characters) and returns optimum assignment of labels to symbols assuming that an optimum assignment is the sum of the sub-assignments. The following section briefly reviews the general concepts of dynamic programming.

Dynamic Programming

The essence of many segmentation-based recognition methods is a dynamic programming technique for calculating the best matching score between a word $C_1C_2 \ldots C_{Nc}$ and an image represented by a sequence of primitive segments $S_1S_2 \ldots S_{Ns}$:

$$\texttt{value}(N_c, N_s) = \texttt{max}\{\texttt{value}(N_c - 1, k) + \texttt{match}(S_{k+1,Ns}, C_{Nc}) \| N_c - 1 \leq k < N_s\}$$

where match (segment(s),character) is an evaluation function that estimates the correlation between a union of primitive segments and the respective character, and $S_{l,m}$ is the sequence of primitive segments from the lth to the mth [SRI99]. The assumption that each primitive segment contains at most one character can be relaxed. If for example, we allow for a primitive segment to correspond to two characters, we need to update the maximization above by checking additionally the following expression:

$$\texttt{value}(N_c - 2, N_s - 1) + \texttt{match}(S_{Ns,Ns}, C_{Nc-1}C_{Nc})$$

This requires additional training of the evaluation function so it can handle a pair of characters in a single primitive segment. A graphical illustration of the dynamic-programming approach is shown in Fig. 3.12.

In order to find the best matching score of a single word, an array $A(N_c, N_s)$ is formed, where N_c and N_s are the number of characters and primitive segments respectively. The node $A(i, j)$ holds the value of the best match between the first i characters in the string and the first j primitive segments. The induction step is computed according to the above-mentioned formula. After the dynamic-programming phase is finished, the optimal segmentation that is associated with the highest score can be restored by backtracking.

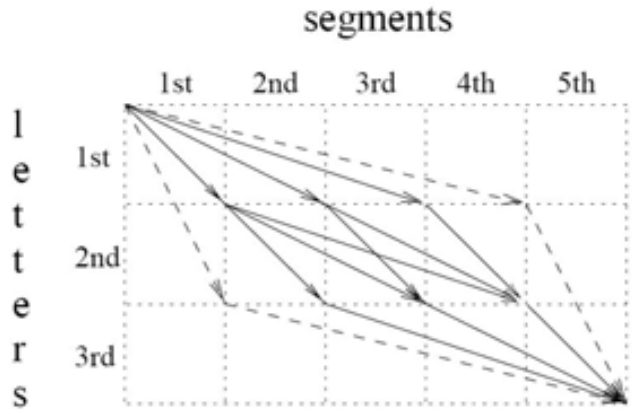


Figure 3.12: A graphical illustration of the dynamic-programming algorithm used for finding the best match between a three-letter word and a sequence of five primitive segments. The dashed lines represent paths in which a single segment contains two characters, and therefore will be considered only if the segmentation criteria tolerates such alternatives. Reprinted from [SRI99]

Dynamic Programming applied to multi-oriented touching character segmentation

Given a touching image, the primitive segments are merged so that the average character likelihood is maximized using DP. The likelihood of each character is calculated using a recognition accuracy obtained by SVM.

To apply the DP algorithm, the primitive segments are sorted from left to right following the direction of α. Let $S_1, S_2, ..S_n$ are list of n primitives. In Fig.3.11(c) the primitive segments are indexed according to their sorting order. We use two tables to store the character likelihood of primitive segments after merging (see Fig.3.13). In the Score Table ST, we enter the classification score $\{c_{uv}\}$ and in the Label Table

LT, we enter the character classification label $\{l_{uv}\}$ where $1 \leq u \leq v \leq n$.

$$c_{uv} = \bigcup_{i=u}^{v} S_i \quad \texttt{, score from ST}$$

$$l_{uv} = \bigcup_{i=u}^{v} S_i \quad \texttt{, label from LT}$$

In these tables, the cells correspond to the recognition result of a cumulative grouping of primitive segments. The possible merging results of the primitive segments of Fig.3.11(c) are shown in Fig.3.13(a) and Fig.3.13(b). For example, in Fig.3.13(b), the cell l_{35} represents the character likelihood of merging the primitive segments $s3$, $s4$ and $s5$. The label obtained by our SVM is 'm'. In table ST, the cell c_{35} indicates the corresponding classification score (0.183) to obtain label 'm'. If the classification score of merged segments is very low (a threshold value of 0.1 is decided empirically), we do not consider it. Cumulation of primitive segments is continued till the width of the resultant image is less than $1.2 \times H_w$. This value is chosen based on the size of Latin alphabets.

v (merged primitives) →

u (starting primitive) ↓

	S1	S2	S3	S4	S5	S6
S1	0.969					
S2		0.856	0.705			
S3			0.566	0.380	0.183	
S4				0.266	0.645	0.245
S5					0.158	0.839
S6						0.103

(a)

v (merged primitives) →

u (starting primitive) ↓

	S1	S2	S3	S4	S5	S6
S1	O					
S2		B	a			
S3			t	A	m	
S4				t	n	m
S5					l	R
S6						2

(b)

Figure 3.13: (a) Score Table ST and (b) Label Table LT of character string "OBAR".

Next, we check the total likelihood of the character groups. The group having maximum likelihood is chosen and the corresponding primitives merging are their segmentation result. In Fig.3.11(c), 1st segment corresponds to letter 'O', 2nd segment corresponds to letter 'B', 3rd and 4th segments correspond to letter 'A' and 5th and 6th segments correspond to letter 'R'. The assignment of primitive segments for the characters O B A R is also represented by:

$i \rightarrow 1\ \ 2\ \ 3\ \ 4$ and $j(i) \rightarrow 1\ \ 2\ \ 4\ \ 6$

where i denotes the letter number, j(i) denotes the number of the last primitive corresponding to the i-th letter. Note that the number of the first primitive segment corresponding to the i-th letter is $j(i-1)+1$. Given $j(i)$, (i = 1...n), the total likelihood of characters is represented by

$$L = \sum_{i=1}^{n} l(i, j(i-1)+1, j(i)) \tag{3.1}$$

where $l(i, j(i-1)+1, j(i))$ is the likelihood for i-th letter. The optimal assignment (the optimal segmentation) that maximizes the total likelihood is found in terms of the dynamic programming as follows. The optimal assignment $j(n)^*$ for n-th letter is the one such that

$$L^*_{j(n)} = L(n, j(n)^*) = \texttt{Max}L(n, j(n)) \tag{3.2}$$

where $L(k, j(k))$ is the maximum likelihood of partial solutions given $j(k)$ for the k-th letter. This is defined and calculated recursively by
$L(k, j(k))= \texttt{Max}_{j(1),j(2)..j(k-1)} \sum_{i=1}^{k} l(i, j(i-1)+1, j(i))$

$$= \texttt{Max}_{j(k-1)}[l(k, j(k-1)+1, j(k)) + L(k-1, j(k-1))] \tag{3.3}$$

$$\texttt{and} \quad L(0, j(0)) = 0 \ \texttt{for} \ j(0) = 1, 2, ...m \tag{3.4}$$

Starting from (3.4), all $L(k, j(k))$'s are calculated for $k = 1, 2, ..., n$ using (3.3) to find $j(n)^*$ using (3.2). The rest of $j(k)^*$'s ($k = n-1, n-2, ..., 1$) are found by back tracking a pointer array representing the optimal $j(k-1)^*$'s which maximizes $L(k, j(k))$ in (3.3).

Given a segment group, the feature vector is calculated for a character class. Based on the character likelihood, the total likelihood of a word is found in terms of the dynamic programming technique discussed above. In Fig.3.14 we have shown the final segmentation result of the touching character of Fig.3.11(a).

Figure 3.14: Final segmentation lines are drawn on Fig.3.11(a) after applying our proposed approach.

3.4 Experimental Results

To the best of our knowledge, there exists no standard database to evaluate character segmentation methods in a multi-orientated and multi-size context. For our experiments, we have constructed our own database using real as well as synthetic data.

The real data is collected from graphical documents e.g. maps, newspapers and magazines. It contains touching character components of different font, size and orientation. Synthetic data is generated from Arial and Times New Roman fonts. These datasets have been produced using the system described in [DVPK10]. The touching strings are composed of single-word images with different scales, orientations and fonts, with corresponding groundtruth at character level. The words are selected from a dictionary (of 52 country names), with random scaling and rotation parameters. Average number of characters in the words are 7-8. See Appendix A.2 for more details.

3.4.1 Performance Evaluation of 2-Touching Characters

For the experiment of 2-touching characters, we considered 1250 components. These are from real data. In Fig.3.15, we have shown some touching images with their segmentation results. We have obtained 93.16% accuracy in two-touching character segmentation with 176 dimensional features when no rejection is considered.

Figure 3.15: Few images showing segmentation lines in 2-character touching.

To get the idea of our touching character segmentation result, we provide the result using different top choices in Table.3.2. It is to be noted that we achieved high accuracy (97.52%) from our scheme with only top two choices of segmentation. We also provide the accuracy of touching character segmentation based on rejection threshold TH_{rej} (discussed in Section 3.2.4) in Table.3.3. We have obtained 99.34% accuracy in 2-touching characters when 6.5% components are rejected.

Table 3.2: Segmentation result based on different top choice (when no rejection is considered).

Number of top choices	Segmentation Accuracy
Only 1 choice	93.16%
Only 2 choice	97.52%
Only 3 choice	98.64%

Table 3.3: Segmentation result based on different rejection.

Rejection Threshold (TH_{rej})	Segmentation Accuracy
0%	93.16%
2.7%	96.83%
6.5%	99.34%

3.4.2 Performance Evaluation of n-Touching Characters

In the experiment on n-touching character segmentation, we tested our scheme on 450 words (200 real and 250 synthetic). The synthetic data contains touching as well as non-touching characters. There were 880 touching components in these 450 words. Also we noted that 2050 characters touched in these 880 touching strings. The touching strings are of different size and orientation. Some of the data are in upside down way to check the rotation invariance nature of our method.

In the experiment of n-touching character segmentation, at first, we provide some qualitative results to show how segmentation are done with our approach. To get an idea about the segmentation results, we have shown some touching images with their segmentation results in Fig.3.16. The character segmentation is done using character recognition by angle based features. In Fig.3.16(c), some of the words are touching in curvi-linear fashion. Our method also segmented them correctly.

We have compared the touching character segmentation results using 2 different multi-oriented text descriptors namely: angle based features and Fourier-Mellin moment. Fourier-Mellin moment shape descriptor has been chosen due to its good performance in recognizing isolated multi-oriented characters [AOC$^+$00]. We obtained 91.36% and 88.38% segmentation accuracy in overall experiment using angle based features and Fourier-Mellin respectively. In Table 3.4, we provide the accuracy of touching characters segmentation based on number of characters present in a touching component. We noted that, our system provides better results on 2-character touching strings than 3 or more character touching strings. As the number of characters in the touching component increases, the complexity enhances and hence the separation task. Fig.3.17 provides the comparative results of different datasets using two different features. It can be noted that, angle based features provides better segmentation results than that of Fourier Mellin in all these datasets.

Table 3.4: Segmentation results on touching string.

No. of Characters in a Component	Total Components	Angle-based Feature	Fourier Mellin
2	635	92.60%	91.18%
3	206	89.97%	86.25%
≥ 4	38	86.18%	73.68%
Total	879	91.36%	88.38%

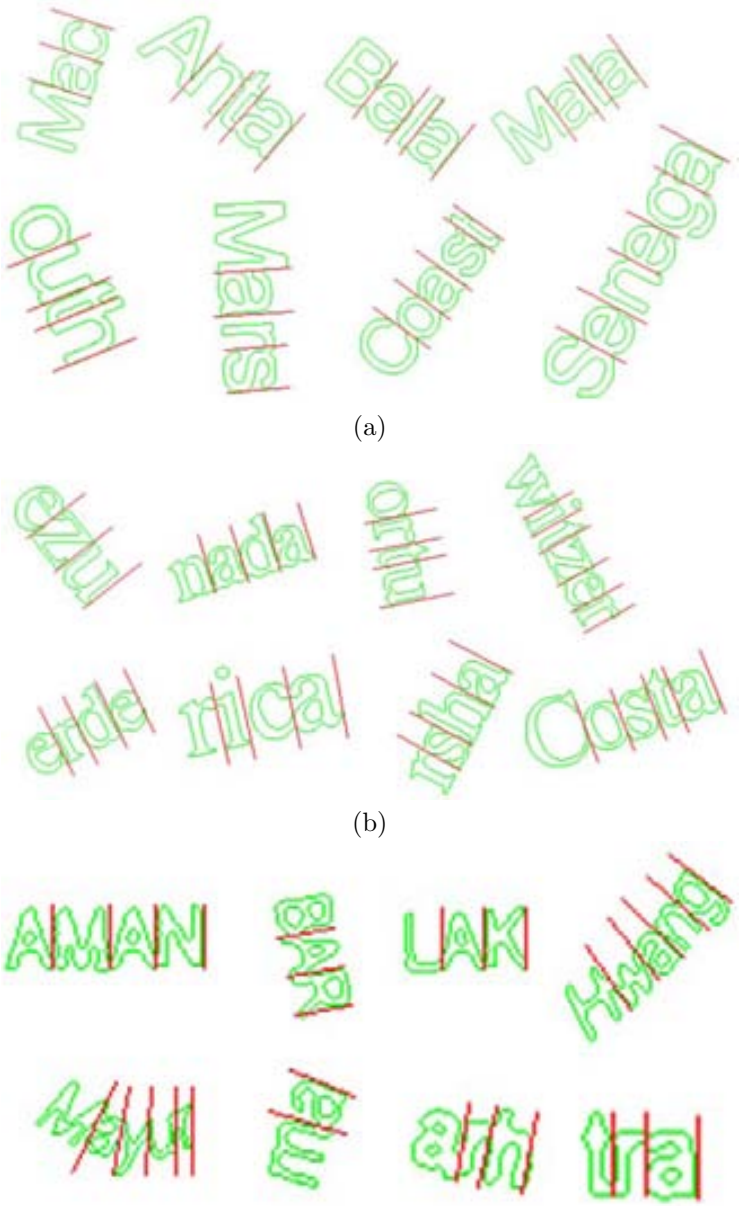


Figure 3.16: Correct segmentation results of different datasets : (a) Arial font (b) Times New Roman font (c) Real data.

Error Analysis

Fig.3.18 shows some wrong segmentation of touching characters from our method. It is noted that most of the segmentation errors are due to following cases. (a) When a

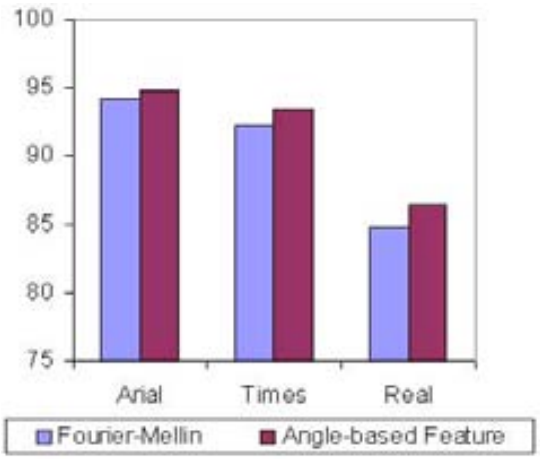


Figure 3.17: Percentage of touching character segmentation accuracy in datasets of "Arial", "Times New Roman" fonts and Real data.

touching string can be segmented in more than two ways to get the valid segmented characters. For example, in Fig.3.18(a) the touching string formed from the characters 'r' and 'm'. But this touching string can be visualized as 'r-r-n' also, and our system segmented this string into 'r', 'r' and 'n' instead of 'r' and 'm' which we consider as erroneous. (b) The character shapes like 'h' (Fig.3.18(b)) may be splitted in two parts and our system segments this character into 't','I'. We also considered it as wrong segmentation. (c) Since our method is based on convex hull, when touching is made in two or more positions, then we may not find any segmentation point in the touching cavity region. Hence we get erroneous results.

This problem can be avoided, by using a word dictionary in the dynamic programming algorithm. In this dictionary based approach, a lexicon containing a large number of words is pre-defined and heuristic methods, such as maximum matching, are utilized to match against the lexicon to segment such touching components.

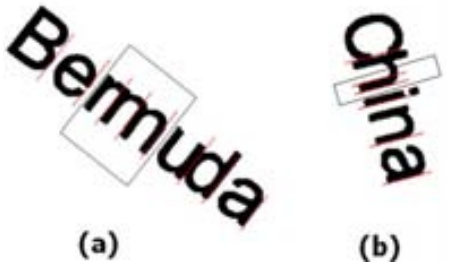


Figure 3.18: Some character strings showing wrong segmentation results.

Comparison of result with/without knowing number of components

We have performed an experiment for the evaluation of having the knowledge a priori the number of components present in a touching component. This test is performed to check whether the information of number of characters improves the performance of touching component segmentation or not. For the experiment, we have created a restricted dataset of touching components which contain 2 characters. The dataset contains 635 components (as mentioned in Table 3.4) and the characters are in multi-scale and multi-orientation. The segmentation on this dataset is done based on the concept discussed in Section 3.2. We have obtained 95.74% of accuracy in this experiment. From the Table 3.4, it can be noted that we obtained 92.60% accuracy on two character touching strings. Thus, it is to be noted that we achieved 3.14% (95.74% - 92.60%) higher accuracy than dynamic programming based approach when additional information of number of characters in the touching component is used.

3.5 Conclusions

In this chapter we have proposed a scheme towards segmentation of multi-oriented and multi-sized n-character touching strings. First, the touching component is segmented into primitives and then the best sequence of model characters shapes are obtained based on a dynamic programming approach using the primitive segments. This algorithm is efficient to take care of character string in noisy environments.

We also performed an adhoc segmentation approach of touching characters based on the knowledge of the number of characters in the touching string. In such restrictive dataset, we have obtained better performance. But, in a real environment, it is not possible to know a priori the number of characters in the touching component. Hence, the proposed approach based on dynamic programming explores the complete idea for such segmentation.

To the best of our knowledge, this work is pioneer towards multi-oriented and multi-sized n-character touching strings segmentation. We have tested our method on multi-oriented touching character data with different fonts and scale. As the feature of text characters are devised for multi-scale and multi-orientation, some touching characters are not segmented properly due to different possibility of segmentation. The efficiency can be improved by using a word dictionary in the dynamic programming algorithm and by using contextual information.

It is to be noted that the previous chapter and this chapter discuss in detail about isolated and touching character recognition in multi-size and multi-orientation environment. In the next chapter, we will present text character extraction from graphical documents.

Chapter 4

Text Character Separation in Graphical Documents

In chapter 1, we have mentioned that OCR in graphical documents is a difficult task. It is due to frequent overlapping of text characters with graphical lines in such documents and hence separation and recognition of text characters are challenging. This chapter deals with character separation in graphical documents. Graphical document images usually contain more than one kind of graphical symbols besides text information. It requires proper understanding of text and graphical symbols to interpret these document images. Here, first we present a methodology to retrieve text and symbols from graphical documents. Connected component features and skeleton information are used to identify text character symbols from graphical lines on the basis of their geometrical features. Next, we present the adaptation of the Scale Invariant Feature Transform (SIFT) approach in the context of text character localization (spotting) in graphical documents. Experiment is performed in a dataset of graphical documents of English script.

4.1 Introduction

In Chapter 2 and 3, we have discussed about character recognition and touching character segmentation in multi-scale and multi-rotation environments. In graphical documents, however, text characters do not appear isolated. These documents usually contain more than one kind of graphical symbol besides text characters information. Because of the difference in characteristics, texts and graphics symbols are processed in different ways. Thus, separation of text/graphics is important task before processing them separately.

The digitization programs from paper-based documentation towards computerized storage and retrieval systems have been prompted by many advantages to be gained from the "electronic document" environment. However, several factors are still in research progress in handling these documents. Generally, paper documents are often

a mixture of text and graphics, and a system must exist to separate the text from the graphical objects before useful electronic versions of the documents could be created and used. Once the text image has been separated, it is then usually processed by a character recognition system while the graphics could be edited in a graphics editor either as raster or vector graphics. For example, due to the emergence of Geographical Information Systems (GIS), map acquisition and recognition have become a pursued topic, both by the industry and the academy.

The interpretation of graphical documents requires the discrimination of graphical parts and text before recognition of such multi-oriented text. The distinction between text and graphics might be in a sense subjective and statistical. The problem for detection of such text characters and graphical lines is many-folded. Text/symbols often touch/overlap with long graphical lines. Sometimes, the text lines are curvilinear to annotate graphical objects. Also, due to the usage of multi-oriented and multi-scale characteristics of text characters, the interpretation of such documents is difficult.

4.1.1 Motivation

Separation of text/graphics in document images is one of the fundamental aims in graphics recognition. In forms processing, the problem of overlapping text has been dealt with reasonably well [WS94]. Forms contain only straight lines and the problem is much simplified. Graphical documents such as maps, electrical diagrams, etc. contain more complex and heterogeneous information which requires proper discrimination of text/graphics [CT01], [FK88], [TTP+02]. Here, the aim is to segment the document into two layers: a layer assumed to contain text and symbols and the other one containing the rest of graphical objects such as diagrams, long lines, border of the regions etc. As an example, see Fig.4.1, where some characters are touched/overlapped with graphical lines, and segmentation of such documents is very difficult. The problem has received a great deal of attention in the literature because of the different processing approach of text and graphics. At the component level the problem is not too intense. The spatial distribution of the components and their sizes, can be measured in a number of ways, and fairly reliable classification can be obtained. Difficulties arise however, when either there is text and symbol embedded in the graphics components, or text and symbol touched with graphics. It includes frequent intersection of text and symbols with graphical lines and curves and segmentation of such documents is very difficult. The separation problem of text and graphics intersections has not yet been dealt successfully, although there exist many pieces of published papers.

4.1.2 Related Work

Text/symbol identification in complex documents is done in two ways. Majority of the methods use a segmentation approach of these text/symbols and then recognize them. On the other hand, a few methods work on recognizing the symbols before approaching segmentation.

The algorithm due to Fletcher and Kasturi [FK88] uses simple heuristics based

Figure 4.1: Locations of isolated (bounded by blue box) and touching (bounded by red box) characters of 'R' are shown in a part of graphical image.

on the characteristics of text characters. The method is insensitive in text font style, size and orientation. One of the assumptions was that the text characters do not touch with graphics or other characters and each text character forms an isolated component.

Directional mathematical morphology approach has been used by Luo et al.[LAD95] for separation of character strings from maps. The idea is to separate large linear segments by directional morphology and histogram analysis of these segments. Large segments are considered as part of graphics; effectively leaving small text character segments.

Lu [Lu98] used an algorithm to erase non-text regions from mixed text and graphics engineering drawings, rather than extracting text regions directly. The algorithm is used to extract text characters, dimensions, and symbols on documents of engineering drawings.

Tan et al. [TN98] illustrates a system to extract text strings from a mixed text/graphics image using a *Pyramid* structure. Multi-resolution representations of such a pyramid structure help to select different regions for segmentation. The pyramid helps to identify and locate words or phrases in the image efficiently and quickly. Road maps have been chosen for the experiments. Extraction of the road names that are labeled along the contours of roads are described.

Cao and Tan[CT01] proposed a specific method of detecting and extracting text characters that are touched to graphics. It is based on the observation that the constituent strokes of characters are usually short segments in comparison with those of graphics. They looked for the lines in the overlapped region on the vectorization of the document image. It combines line continuation with the feature line width to interpret intersection of text and graphics.

A more consolidated method is proposed by Tombre et al. in [TTP+02]. This method is based on the analysis of the connected components, originally proposed by Fletcher and Kasturi [FK88]. The modified algorithm has covered a number of improvements to make it more stable for graphics-rich documents. For a general text graphics separation method they discussed about the selection of thresholds. A post-

processing step is proposed to retrieve the text components which are touched with graphics. This is done by local segmentation analysis in the distance skeleton of the image.

Dhar and Chanda [DC06] present a methodology for extraction and recognition of symbol features from topographic maps. The method proceeds by separating the map into its different color layers and then recognizes the features in each layer on the basis of symbol-specific geometrical and morphological attributes. The output is stored in the form of an "e-map" or computer recognizable map. And later this information is used to answer user queries.

Su et al. [SLY+09] proposed a holistic and contextual constraints of the text and the graphical objects at different hierarchies in the engineering drawing. The algorithm segments texts by recognizing and extracting graphics. Then it employs a novel two-step coarse-to-fine process to first group potential characters into string groups and then extract the individual strings from them. A prior contextual knowledge is used to determine the potential orientation and location of the character string.

Recently, Hoang and Tabbone [HT10] considered text and graphics components as two separate two-dimensional signal which are mixed in graphics rich documents. They employed Morphological Component Analysis (MCA) method to obtain separate text and graphics components by choosing discriminative over-complete dictionaries. Curvelet transform is used as the dictionary for graphics and the undecimated wavelet transform is used as the dictionary for text.

The table 4.1 shows a comparative study in different text graphics separation approaches. Connected component based techniques are more general in the context of text graphics information. On the other hand skeleton based methods offer much more reliability but has not been explored in that extent.

Table 4.1: Comparative study of the different approaches

Approach	CC Analysis	Skeleton	Morphology
Fletcher et al. [FK88]	Yes	No	No
Luo et al. [LAD95]	No	No	Yes
Cao et al. [TN98]	Yes	Yes	No
Tombre et al. [TTP+02]	Yes	Yes	No
Dhar and Chanda [DC06]	Yes	No	No
Hoang and Tabbone [HT10]	Yes	No	Yes

Because of the complexity of the problem, the separation of text and graphics has not yet been dealt successfully when text and graphics portion intersects. The shape of non-analytic curves found from text character is difficult to analyze where background noise or overlapping/touching of graphical lines exist. Existing text/graphics separation methods considered so far generally suffer from lack of robustness when text graphics touch/overlap. In summary, most of the methods mentioned above are based on one or two of the assumptions listed below:

- The document images should not contain many noise components, and they should be prepared noise-free, using some standard.

- The text should be printed or hand-printed, and it should seldom touch graphics. It is preferably written vertically or horizontally, rather than in any other direction.

In this chapter, we consider taking care of these problems using a combination of top-down and bottom-up approaches.

4.1.3 Outline of the Approach

The segmentation of overlapping text and graphics is in fact a chicken-and-egg problem. In one hand, the purpose of segmentation is character recognition. On the other hand, correct segmentation may require the recognition of the characters [CT01]. In our proposed method, text/graphics components are separated in a bottom-up approach. An approach using connected components and skeleton information is presented to segment text characters from pixel level. Next, detection of characters in touching/overlapped graphical lines is explored using a top-down object detection approach such as SIFT [Low04].

Using features from connected components and skeletons, we propose an approach to segment isolated characters, touching character, dash and long line components. The component in which both character and long line present due to overlapping are considered as mixed component. Using Hough transform and skeleton analysis, these mixed components are analyzed and text characters are segmented from these components.

Next, we present the evaluation of the SIFT approach in the context of text character spotting in graphical documents. SIFT approach deals with the localization and detection of multiple instances of text characters in overlapping regions. The character shapes are extracted from graphical document using an isolated character extraction method. Next, these text characters are used as query images to search other instances of characters of similar shapes in touching/overlapped graphical regions. For example, we showed some isolated characters (bounded by blue box) in Fig.4.1. The isolated characters are learnt on-the-fly to have the knowledge of shape of the query character. Next, other similar characters which are touching (bounded by red box in Fig.4.1) are searched using this knowledge. Thus, if a character is touched with graphical lines, a top-down approach is used to locate them.

The rest of the chapter is organized as follows. We discuss the text/graphics segmentation approach from pixel level in Section 4.2. In Section 4.3, we present the adaptation of SIFT approach to detect text characters in graphical documents. The experimental results are demonstrated in Section 4.4. Finally, conclusion is given in Section 4.5.

4.2 Text/Graphics Segmentation

We consider a bottom-up approach for text/graphics separation. In a general bottom-up approach the individual base elements (pixel) of the system are first specified in great detail. These elements are then linked together to form larger subsystems (connected component), which then in turn are linked, sometimes in many levels, until

a complete top-level system (text, graphical elements) is formed. In our system, we start from pixels in raster images. Next, connected components (CC) are formed from pixels. Next, skeleton information is computed from these connected components. With the CC and skeleton information, text/symbols are finally separated. The different steps of this approach will be discussed in the following sections.

4.2.1 Connected Components and Thinning

In this section, the connected component labelling and thinning process are discussed.

Connected Component (CC) Labeling

Labeling of connected components ("objects") is the basis for the generation of object features as well as of some kind of filtering. Many text graphics separation methods in binarized documents are based on connected components analysis i.e. grouping black pixels together which are connected. It works by scanning an image, pixel-by-pixel [BW97] (from top to bottom and left to right) in order to identify connected pixel regions, i.e. regions of adjacent pixels which share the same set of intensity values V. The connected components labeling operator scans the image by moving along a row until it comes to a point p (where p denotes the pixel to be labeled at any stage in the scanning process) for which V_{pix}=1. When this is true, it examines the four neighbors of p which have already been encountered in the scan (i.e. the neighbors (i) to the left of p, (ii) above it, and (iii and iv) the two upper diagonal terms). Based on this information, the labeling of p occurs as follows:

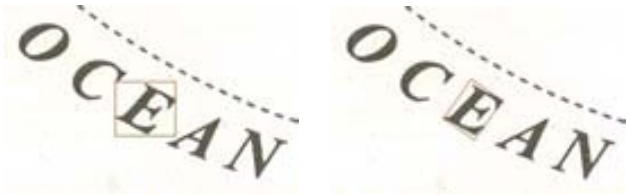
- If all four neighbors are white, assign a new label to p, else
- if only one neighbor has V_{pix}=1, assign its label to p, else
- if more than one of the neighbors have V_{pix}=1, assign one of the labels to p and make a note of the equivalences.

After completing the scan, the equivalent label pairs are sorted into equivalence classes and a unique label is assigned to each class. As a final step, a second scan is made through the image, during which each label is replaced by the label assigned to its equivalence classes.

The characters and symbols found by connected component analysis are classified through feature extraction. Simple geometrical features can be very effective in many applications [BW97]. Some effective geometrical features of a connected component are namely, aspect ratio, perimeter, area, maximum and minimum distances from the boundary to the center of mass, number of holes, Euler number (number of connected components minus number of holes), compactness, horizontal and vertical projections, etc.

Minimum Enclosed Rectangle (MER): Some of these geometrical features mentioned above involve the computation of the bounding box around the text/symbol. Many existing approaches [FK88] use simple rectangle to compute geometrical features. Since we work in graphical documents and the text characters in such documents are rotated, we use *Minimum Enclosed Rectangle* (MER) for such feature computation.

This rectangle surrounds all the pixels in the connected component in such a way, that it finds minimum area. It is calculated by building convex hull on the set of all the pixels and applying rotating calipers technique to the hull. Fig.4.2(a) shows the normal enclosed rectangle and Fig.4.2(b) shows the best enclosing rectangle of a connected component (here, it's a text character 'E').



(a) (b)

Figure 4.2: Text Character Image in (a)Horizontal-Vertical Rectangle and (b) Minimum Enclosed Rectangle.

Thinning

Thinning is the process of reducing the thickness of each line of patterns to just a single pixel. Lam et al. [LSC92] presented a comprehensive survey on thinning methodologies. It discussed about a wide range of thinning algorithms, including iterative deletion of pixels and nonpixel-based methods. There are algorithms to find thinned image by medial axis and distance transform. Some algorithms work on the contour line.

One of the popular thinning algorithm for character recognition, proposed by Ahmed and Ward [AW02] works well for text document images due to its rotation invariant nature. This algorithm is iterative. At each iteration, it deletes every point that lies on the outer boundaries of the symbol, as long as the width of the symbol is more than one pixel wide. This is based on a set of rules which are applied simultaneously at each iteration to every pixel. The iterations are repeated until no further changes occur. It has the advantage of producing the same thinned symbols regardless of the rotation of the original symbols. It produces less extraneous pixels (distortions) (see Fig.4.3). Thinning characters to their central lines results in thinned symbols that better represent the characters (See [AW02] for details).

4.2.2 Selection of Size Criteria for Text Component Extraction

While separating the connected components, it is necessary to set a size criteria that can be used as an initial filtering approach to roughly classify the components into texts and graphics. Tombre et al. [TTP+02] proposed a size-histogram analysis from the bounding box size of all the connected components. By a correct threshold selection obtained dynamically from the histogram, the large graphical components are discarded, leaving the smaller graphics and text components. To do so, the most populated area and the average area are computed. Let A_{mp} be the number of

(a) (b)

Figure 4.3: (a)Different Symbols with different fonts in English, Arabic and Chinese. (b) Corresponding thinned images. (Reprinted from [AW02])

components in the most populated area and A_{avg} be number of components having average size in the histogram. Then, size threshold T_{txt} is considered as

$$T_{txt} = h \times max(A_{mp}, A_{avg}) \tag{4.1}$$

where h is the proportionality factor. This approach is followed for initial connected components classification. In graphical documents, the characters are in multi-oriented fashion and inclined to certain angle. So we have used MER of components for size histogram analysis. As the characters are of different sizes, the value of h was set to 3 from the experiment [TL03].

4.2.3 Components Classification

Above criteria based on the area feature of the connected components is good enough to group a component into one between text or graphics layer [TTP+02]. But, if characters overlap with long lines, they cannot be separated by simple CC based rules. Skeleton information can be useful here to detect the long segments. It is assumed that the lengths of character segments are smaller compared to those of graphics. Hence, to detect the long graphical lines in such components, we integrate skeleton information. The skeleton segments are decomposed at the intersection point and the length of segments (l_s) are calculated. l_s is computed from MER of the corresponding segment. If the length of a segment is bigger than T_{txt}, the segment is considered as long segment.

We distribute the components into 5 classes, isolated characters, touching characters, dashed components, long components and mixed components. The distribution is done based on the information of connected components and skeleton segments. In the skeleton image of each component, if there exists no long segment, then the component is included into one of the 3 groups : isolated character, touching character or dash component group. Otherwise, it is considered as mixed or long component. The description of each of them is given below. Fig.4.4 shows different components of a graphical document.

- Isolated Character: This group includes text characters and small symbols which have size less than T_{txt}.
- Dashed Component: These are mainly small elongated components. These components are separated from isolated character groups using aspect ratio. The value of aspect ratio is considered as 4 using experimental results. This group includes dash segments from the dash line along with some isolated characters, such as '1', 'l' etc. These alphabets/numbers are combined into these layers, because, at the pixel level analysis, these characters hold the same property as dash segments and cannot be separated without context information.
- Touching Character: It consists of the components where more than one isolated characters touch each other. These connected components have larger aspect ratio than isolated characters and do not contain long skeleton segments. The length of skeleton segments in this component must be less than T_{txt}.
- Long Line: These are the graphical components. The segments of these components are larger compared to the text characters size T_{txt}. These lines can be of straight or curve type.
- Mixed Component: This group consists of the components where both graphical components and text components are present. It happens due to their overlapping in that component. The long graphical lines are of two types: straight and curve. The separation of these graphical lines from these components is explained in Section 4.2.4.

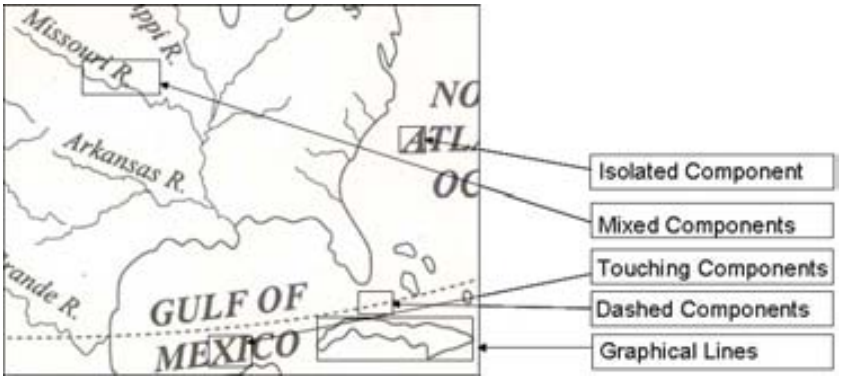


Figure 4.4: Different types of components are shown in a map image.

4.2.4 Long Graphical Lines Removal

Mixed components are next considered for text separation. Segmentation of the mixed components is done in two stages. In the first stage, long straight lines are removed from the mixed components using HT analysis. Next, long curve segments

are detected and removed from the remaining part of the image using skeleton analysis. The flow chart of the proposed scheme is shown in Fig.4.5. It presents the overall process for mixed components segmentation. In following sections, we will explain how long lines are segmented from mixed components.

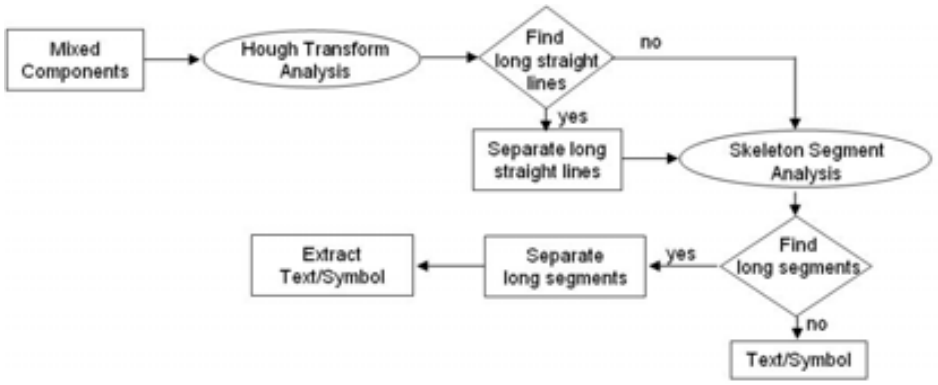


Figure 4.5: Flow-chart for mixed component segmentation.

Removal of Straight Lines

We perform Hough Transform (HT) to detect the straight lines present in the mixed components. In the following sections, we will briefly discuss about HT and later we will show this approach in our application.

Hough Transform: The Hough Transform [DH72] is a technique which can be used to isolate features of a particular shape within an image. Because it requires that the desired features be specified in some parametric form, the classical HT is most commonly used for the detection of regular curves such as lines, circles, ellipses etc. In document image HT can be used to detect straight lines at any orientation. This method involves a transformation from the image coordinate plane to parameter space or in other word from Cartesian space(x, y) to sinusoidal curves in (ρ, θ) space via the transformation

$$\rho = xcos\theta + ysin\theta \tag{4.2}$$

where ρ is the distance from the origin of the (x, y) space to the line, and θ is the angle of the normal to the line. This produces a sinusoidal curve in the (ρ, θ) space for each point (x, y). Each (x, y) location of a foreground pixel in the image plane is mapped to the locations in the transform plane for all possible straight lines through the point (for all possible values or ρ and θ). When multiple points are collinear, their transformations will intersect at the same point on the transform plane. Depending on accumulation of pixels, the straight lines are sorted out. The (ρ and θ) locations having the greatest accumulation (maximum) of mapped points indicate lines with those parameters (see Fig.4.6).

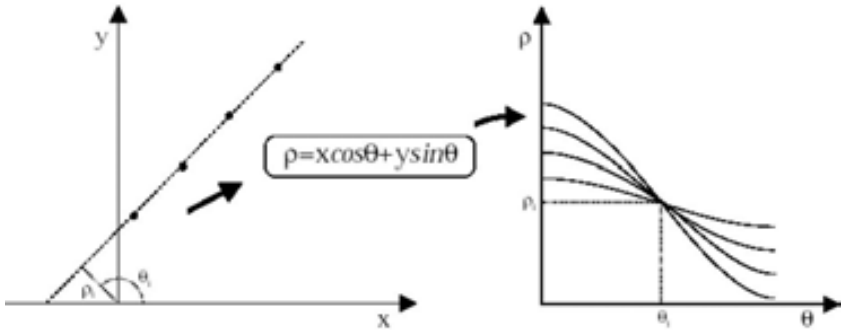
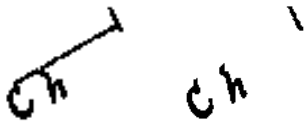


Figure 4.6: Transformation of a line from cartesian to parametric space.

Segmentation of Straight Lines: We have used HT technique to detect straight lines in mixed components. See Fig. 4.7(a), where some characters are touched with a straight line portion. Our objective is to keep the character portions while removing the long graphical line. To do so, we have used stroke-width information in our approach to calculate the line width (l_w). Stroke width of a line segment is computed using the statistical mode of the black run-lengths, obtained by scanning the segment in horizontal, vertical and two horizontal directions (See Section 3.2). The portions of the line where the width is more than l_w, are separated from straight line. Fig. 4.7(b) shows the remaining part of the image after straight line removal in Fig. 4.7(a) by HT analysis.



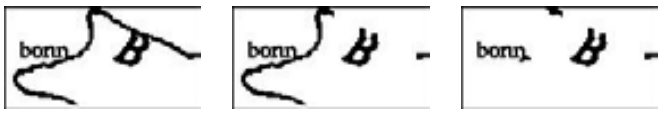
(a) (b)

Figure 4.7: (a)Characters are touched with a long line in a mixed component .(b) Isolated characters after removal of straight long line.

Removal of Curve Lines

Other long lines that are touched to text/symbols are considered in this step. According to the text and graphics feature, it is assumed that the lengths of character segments are smaller compared to those of graphics. At first, the mixed component is thinned to obtain the skeleton image. Next, all the segments of skeleton are decomposed at the intersection point of the skeleton image. The segments having length (l_s) larger than character size threshold (T_{txt}) are chosen for elimination. After removal of long segments, the remaining portions are considered as character components. Fig. 4.8(a) shows an initial mixed component with touching characters. The component after removing long straight line is shown in Fig. 4.8(b). The long curve lines are removed by checking the segment length. After removing curve long lines, the

remaining portions are shown in Fig. 4.8(c).



(a) (b) (c)

Figure 4.8: (a)A mixed component. (b) Long straight part is removed. (c) Extracted part after removing the long curve lines.

4.3 Text Character Spotting using SIFT

Recently, the Scale Invariant Feature Transform (SIFT) [Low04] has emerged as a cutting-edge methodology in general object recognition as well as for other machine vision applications [BLGT06]. The spatial organization of patch-based descriptors computed from key-points is a powerful tool to recognize objects scenes [MS01]. One of the interesting features of the SIFT approach is the capability to capture the main features of an object by means of local patterns extracted from a scale-space decomposition of the image. The main advantage of SIFT is that, this approach works in invariant to image scale, rotation, addition of noise/occlusion. With the wide applicability and potential of this technique, for the classification of 2D objects, recently, this approach is also investigated in graphical symbol recognition [RL08]. Though, SIFT has been researched for object detection in other computer vision field, the use of SIFT in text/graphics separation is not explored before.

We extract the knowledge from a bottom-up (pixel-based) approach and use it for SIFT for text detection. First, the isolated characters are extracted from a graphical document. Next, these characters are labeled using a rotation invariant character recognition system. Given a query text character, the system learns the shape of characters from the recognized (labeled) character sets. Thus, the different shapes of each character are learnt dynamically using this bottom-up approach. Next these characters are used as query images to search other instances of characters of similar shapes in touching/overlapped graphical regions.

The flow chart of the proposed scheme is shown in Fig.4.9. Isolated characters are extracted from graphical components as discussed in the earlier section. These characters are recognized using a SVM classifier. The SVM is trained before to build the character shape models from corresponding feature of different text characters of the database. Finally, the recognized models are queried using SIFT to locate text character in touching/overlapping zones.

4.3.1 SIFT Approach for Object Recognition

The SIFT approach of object recognition is a combination of selecting “local-features” and their “matching” method. SIFT features are invariant to image scale and rotation, and are demonstrated to provide robust matching in both the spatial and frequency domains, reducing the probability of disruption by occlusion, clutter, or

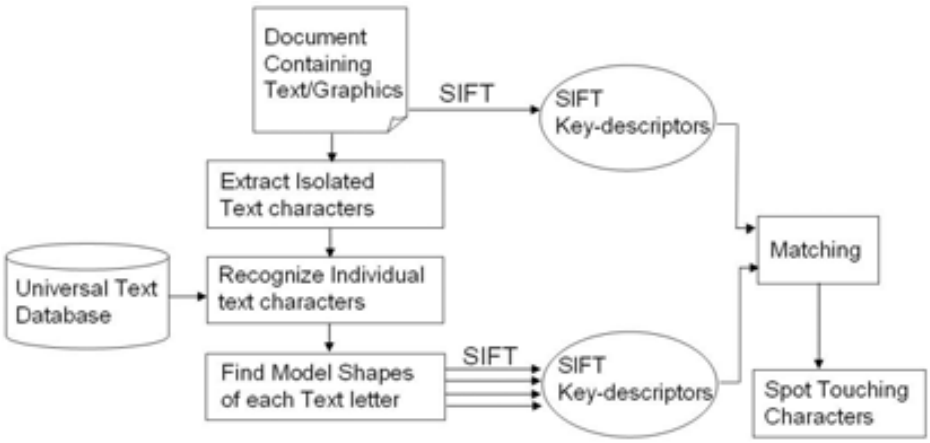


Figure 4.9: Touching text character spotting system.

noise. In addition, the features are highly distinctive, which allows a single feature to be correctly matched against a large database of features. SIFT feature usage for object recognition involves two steps - (1) SIFT feature extraction and (2) Object recognition.

Feature Extraction

Following are the major stages of computation used to generate the set of image features.

Scale space extrema detection: Distinctive points are selected by identifying maxima/minima of the document image after convolving the image with a Difference-Of-Gaussian (DOG) filter. This is done by convolving the image with Gaussian filters at different scales and taking differences of the resultant images.

Key-point Localization: At each candidate location, a detailed model is fit to determine location and scale. Keypoints are selected based on measures of their stability. Once a candidate keypoint has been found by comparing a pixel to its neighbors, a detailed step is performed using the neighbor data for location, scale, and ratio of principal curvatures.

Orientation assignment: One or more orientations are assigned to each key-point based on local image gradient directions. This method is used to incorporate rotation invariance to the key-point. To determine a key-point orientation, a gradient orientation histogram is computed in the neighborhood of the key-point.

Generation of key-point descriptor: The descriptor is meant to encode the key-point and information about the neighboring points. The information is encoded based on the local image gradients at these points. Once a keypoint orientation has been selected, the feature descriptor is computed as a set of orientation histograms on 4×4 pixel neighborhoods. Each histogram contains 8 bins. Thus the descriptor obtained is a 128 ($4 \times 4 \times 8$) element descriptor.

Object Recognition

Recognizing an object using SIFT can be performed using the following steps.

Key-point matching: Key-point matching is done by matching the key-point descriptors from the test image with those of a template query image, using a nearest neighbor approach. The nearest neighbor match is compared with the next (second nearest) closest one to ensure that a match is only accepted if it is distinctive enough.

Hough Transform: The Generalized Hough Transform (GHT) [Bal81] is used to cluster key-point matches that are consistent with a single object hypothesis. Each key-point is characterized by a 2D location, a scale and an orientation. Thus a 4 dimensional hough space is used for this purpose. Finally, a set of potential hypothesis of the object in the test image is obtained using HT. An example of object recognition is shown in Fig.4.10.

Figure 4.10: Example of object recognition with SIFT.

4.3.2 Generation of Text Character Prototypes

We generate universal character models for each character shapes using training data which are collected from different graphical documents. Models are generated using the angular information of contour pixels configuration and SVM classifier (see Chapter 2). Next, the isolated components which were included in text components group using CC analysis and skeleton information are fed to SVM classifier for recognition. A prototype image of each character class will be used to detect other instances of the same character.

For each component, we compute the recognition confidence for all character model classes using SVM classification process and rank the confidence scores in descending order. If we recognize a component with a very high accuracy, we accept it as a good-shaped character. If the difference between the top two recognition scores of a component is high, it is also considered as good-shaped character. The score difference is selected based on experimental results. The components of low recognition accuracy

are not considered as isolated components. In Fig.4.11, we show a portion of the text layer of Fig.4.1 containing isolated characters and their recognition label. It is to be noticed that, the character 'n' is identified as 'u' because of its shape similarity nature in rotation invariant environment. Thus, given an ASCII character to search in the document, we find the shapes used in that document corresponding to the ASCII value using this approach. The isolated characters of each character label are ranked according to the confidence value by classifier. Next, the character shapes which have the highest ranking are selected as the representative characters.

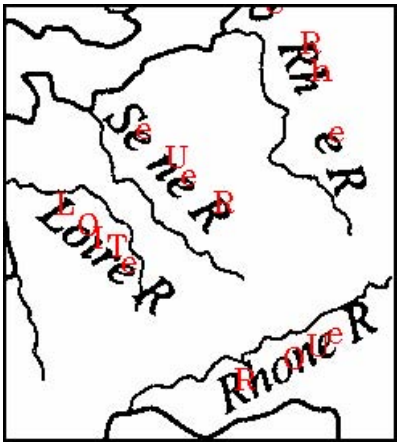


Figure 4.11: A part of a map with their isolated character label shown in Red color.

Graphical documents may contain text characters of different fonts (for e.g. "Arial", "Times New Roman", etc.) or style (normal, italics) to annotate and give importance to find the location names present in the document. To learn on-the-fly the font used in the document, different fonts are trained in SVM. It is to recognize variation of style or font of characters. Thus, for each alpha-numeric text character value, the system finds component shapes according to their font style. This representative text character shape is used as templates to find other text character images that were not extracted or recognize well due to touching or noise. Fig.4.12 shows two characters 'R' and 'S'. These characters are selected using the highest recognition confidence value.

4.3.3 Locating Similar Text Characters using SIFT

The character prototypes found above are used as query model to locate overlapped/ touching characters from graphical documents. SIFT is used to detect and describe the local features in this context. For each text font model we find the local SIFT features. As explained earlier, each SIFT feature is composed of four parts: the locus (location in which the feature has been found), the scale, the orientation and the descriptor. The descriptor is a vector of 128 dimension. Some examples of the keypoints using SIFT approach to two different text characters are shown in Fig.4.12. In the figure, the line denotes the direction of dominant direction. The figure explains

the different keypoint-feature descriptors corresponding to different text characters. Next, SIFT is applied in the document comprising of rest of components and graphical lines.

We reduce the search space in the document using skeleton analysis. According to text and graphics feature, it is assumed that the length of character segment is smaller compared to that of graphics. The skeleton segments having length l_s larger than T_{txt} (T_{txt} is computed from Eq.4.1) are chosen for elimination. The remaining portions after removal of long segments are considered for potential regions of text characters.

(a) (b)

Figure 4.12: (a) and (b) SIFT features of two text characters are shown. A circle denotes the zone of corresponding descriptor.

Next, we search for text characters in the document using SIFT. We may find many false alarms due to natural curvature of graphical lines. Since we estimated the size of character using connected components analysis, different criteria based on character size are used to reduce false positives in our work.

Usually, text character images are very small compared to graphical line segments. So, the character images produce a set of SIFT keypoint descriptors, many of them are for corner regions. These corner regions are not always distinctive to produce the location of the character in the graphical document. Many corner regions exist other than text character regions. To reject the keypoint descriptors which contain only local corner regions, a size threshold is set to reject them. This filtering is done by comparing the region of key-descriptor to the size of query character. A size threshold (T_{K1}) is chosen for this purpose. The value of threshold is selected as $T_{K1} \geq l_{q1}/2$, where l_{q1} is the size of the query character. By this size threshold, the keypoints which capture more than half regions of the text character are only accepted.

The document images may also contain many corner regions which produces many false positive. For this purpose, the SIFT keypoints whose size are very small or very large are not considered. The character size threshold (T_{txt}) is used to remove the keypoints which are larger in document. Very small keypoint-regions are also discarded using a size threshold set up from experimental results.

Finally, the local SIFT features of each text character are matched with each feature to the character SIFT features. The corresponding matching locations are detected as probable zones of text characters. See Fig.4.13, where probable location of two specific characters 'R' and 'S' are shown in the document. It is to be noticed that we have located the characters 'R' and 'S' by our approach which were intersected

by curve lines.

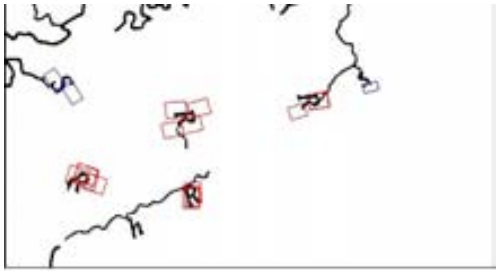


Figure 4.13: Matching result of SIFT features of text characters 'R' and 'S' in the document image. Corresponding SIFT features are marked by different colors. Here, red color indicates matching of character 'R' and blue color indicates matching of character 'S'.

4.4 Experimental Results

In this section we illustrate the performance of the proposed methods. As there exists no standard database to evaluate text/graphics segmentation, we have used datasets of graphical documents like maps, electrical diagrams, etc. These documents are considered from real as well as synthetic data. Real data are collected from maps, electrical diagrams, etc. These document images were digitized by a flatbed scanner at 300 dpi. We have considered 20 different real geographical maps and 10 electrical diagram images to test our method. They contain text characters of different scale and orientation. These datasets are described in Appendix A.5.

We have also used one dataset of synthetic maps which are generated by Delalandre et al. [DPV+08]. Arial font was used for annotation purpose in these maps. The backgrounds (graphical lines, boundaries etc.) of these maps are taken from a few real maps and the text words of country/river names are placed in the foreground using a set of rules (See Appendix A.5.1 for more details). We show two different test images of the same background in Fig.4.14. The graphical long lines i.e. geographical borders and rivers shown in these maps are kept fixed. The text portions are randomly placed and oriented to generate different synthetic documents. We have considered 10 synthetic maps from this dataset.

In these documents, graphical lines exist in both straight and curved ways. The long lines are overlapped with text in many places. The character arrangements in the text strings are of both linear and curvi-linear. In electrical drawings, the characters are touched often due to printing or noise issue.

4.4.1 Performance of Text/Graphics Separation

To demonstrate the efficiency of the proposed method, experiments have been carried out on real as well as synthetic datasets. To get an idea of our text/graphics separation results in graphical documents, some of the images are shown in Fig.4.15.

(a) (b)

Figure 4.14: (a) and (b) Two synthetic maps of the same graphical background but different foreground.

It is to be noted that almost all text character components are extracted using our approach. However, our algorithm produces false positives due to small dash-shaped characters. For example, the dashed components are present in Fig.4.15(d). Most of the errors occurred due to touching of character components with the graphical lines. While removing the graphical lines, sometimes part of the graphical lines retains. In Fig.4.15(d), the character 'N' in the word "MEDITERRANEAN SEA" is recovered but some portion of graphical lines are also present. It was due to the splitting of the segments at the intersection point of skeleton. In Fig.4.15(f), the character 'g' was recovered although it was touched with the graphical circular boundary of a seal.

To evaluate the performance of the system with the retrieved text characters, we use common ratio of precision (P) and recall (R). The precision measures the quality of the retrieval system in terms of the ability of the system to include only relevant items in the result. For a given retrieval result, the precision measure P is defined as the ratio between the number of relevant retrieved items and the number of retrieved items. Whereas recall measures the effectiveness of the system in retrieving the relevant items. The recall R is defined as the ratio between number of relevant retrieved items to the total number of relevant items in the collection. Precision and Recall are computed as follows.

$$P = \frac{\|retrieved \cap relevant\|}{\|retrieved\|}$$

$$R = \frac{\|retrieved \cap relevant\|}{\|relevant\|}$$

A quantitative evaluation of the method is also given in Table 4.2. The evaluation is performed on studying 10 different documents from different documents including electrical diagrams, map, etc. We have used recall measure of extracted text components for the evaluation. We manually count the number of characters in the original images. Next, we check the characters obtained using our approach. Most of the errors in our approach are due to characters printed in reverse color of foreground text layers. See Fig.4.16, where text information are printed in white color which we

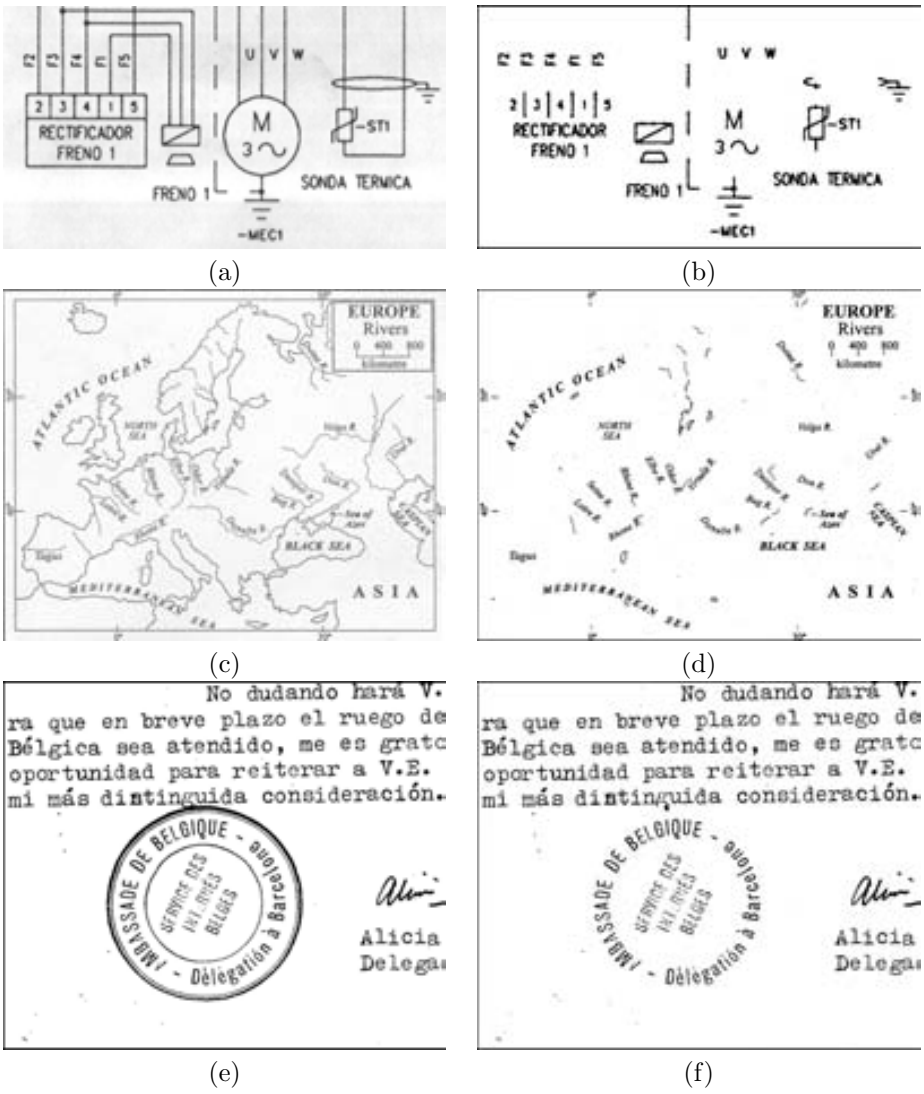


Figure 4.15: Text components are separated from different graphical documents. (b) Electrical diagrams from Fig.(a), (d) Map from Fig.(c) and (f) Seal document from Fig.(e).

considered as background image. With our binarization algorithm, the text characters from these background were not extracted properly.

From the experiment, we noted that, if a long line is not fully straight and it contains sufficient straight portion, this straight part will be detected by Hough Transform and will be removed leaving the other small parts. These small segments will fall into isolated and touching character groups. Similar kind of errors occur for long curve lines. In skeleton analysis, a curve line may not be fully removed if it encounters

Table 4.2: Performance evaluation

Image	Total Number of Characters	Accuracy (Recall)
1	251	248 (98.80%)
2	236	236 (100%)
3	318	303 (95.28%)
4	458	424 (92.58)
5	173	173 (100%)
6	185	184 (99.46%)
7	201	201 (100%)
8	362	326 (90.06%)
9	277	241 (87.00%)
10	180	180 (100%)

Figure 4.16: A document shows the printing of text characters in two different modes of color: black and white.

many junctions with other lines in its path. This will result fragmentation of lines into small segments and hence false alarms occur. We have checked the precision measure of extracted text characters in these 10 images. The average precision of text character extraction is 71.56%. The lower performance is due to many false positives. In our present work, grouping of dash-like components into text/symbol layer was not considered. The use of context information may solve this problem. For example, the isolated dashed components are likely to be of dashed lines if they are arranged in a linear fashion.

4.4.2 Performance of Text Character Spotting

We have considered 10 different real geographical maps to test our method. In these documents, graphical long lines are overlapped with text in many places. Sometimes text characters are broken into different components due to printing or noise issue. We show some qualitative examples in Fig.4.17. In these image potions, different characters 'E', 'N', 'B', and 'w' are searched respectively. Characters 'E', 'N', 'B' are touched with graphical lines or other text characters.

From the experimental result, it is found that this approach can take care of locating characters of arbitrary shape and orientation in graphical documents. Also, it is observed that, our approach can find the locations of corresponding characters with some false positives. These false positives can be removed by using text line information described in Chapter 5.

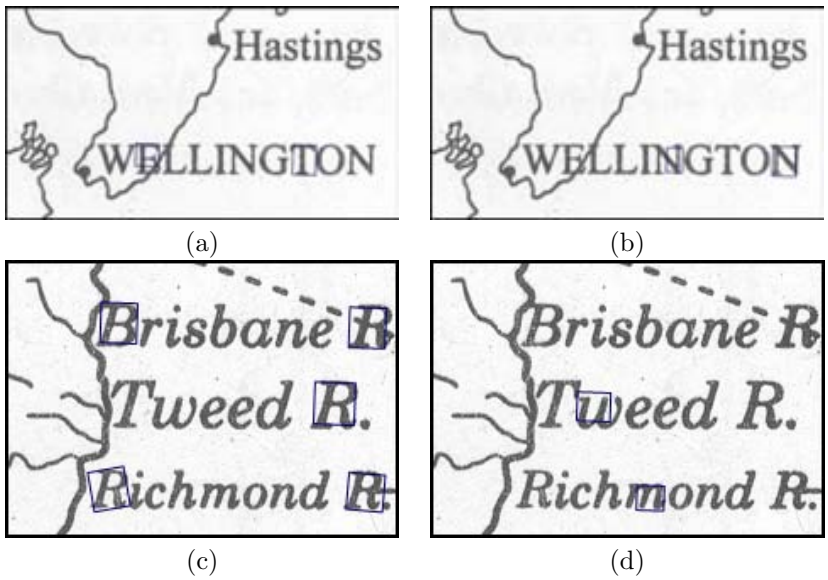


Figure 4.17: Different characters are searched in images (a) Character 'E' (b) Character 'N' (c) Character 'B' and (d) Character 'w'.

In map documents from real datasets we noted that, there were total 68 characters touching/overlapped with graphical lines. Using SIFT, we recovered 55 characters with accuracy 80.88%. Characters like 'L', 't', 'f', etc. were not detected properly, due to their not-enough distinctive nature in real images.

4.5 Conclusion

In this chapter a combination of bottom-up and top-down approaches has been investigated in the context of text character extraction and localization in graphical documents. Here, at first, we grouped the connected components present in the document. CC analysis and skeleton information are used to identify text from graphics.

Next, we have adapted SIFT in this application to locate text characters which are touched/overlapped.

SIFT has been useful to find the probable locations of characters which are not extracted properly due to white noise or overlapping with graphical lines. The local text character model shapes are identified dynamically for each text character. Next, the model shapes are used to locate the similar images in other areas of the graphical image. This enables us to detect the text characters which are touched with graphical lines on-the-fly. SIFT descriptor finds the possible locations of text characters. It is appreciable that the other instances of text characters are detected using SIFT in different poses such as scale and rotation.

Due to smaller size of character shapes and presence of linear segments in Latin text characters, SIFT produces many false positives in documents. Adding size criteria constraint in post-processing, number of false positives are reduced in our approach. The rest of the errors can be solved using context information i.e. the positions of other characters from the string.

For OCR of graphical documents, it is important to group the text characters in line as mentioned in Chapter 1. Also, context information is helpful for better text information extraction in graphical documents. In the next chapter, we will discuss about text line separation from documents containing multi-oriented and curvi-linear lines.

Chapter 5

Text Line Extraction using Background and Foreground Information

In previous chapters, we have explained in detail about text character recognition and extraction in graphical documents. For the OCR of such documents, we need to extract individual text lines from the documents. In such documents, the text lines are annotated in different orientations to illustrate locations or symbols. Extraction of individual text lines from multi-oriented and/or curved text documents is a difficult problem, as the text lines may not follow a single direction only. In this chapter, we propose a novel method to extract individual text lines from such document pages, and the method is based on the foreground and background information of the characters of the text. To effectively utilize the background information, a water reservoir concept is used here. The proposed method is validated by extensive experiment using datasets of graphical documents (e.g. maps, electrical diagrams), camera-based warped documents and noisy images containing seals etc.

5.1 Introduction

In graphical documents (e.g. maps, engineering drawings), artistic documents etc., there exist many printed text lines which are not parallel to each other. Here, the text lines may be printed in several orientations (multi-oriented documents) or the text lines may be curved in shape [BM95, DMS95a, HAT96, POYC01] as explained in Chapter 1. For the OCR of such documents, the text lines in a document must be segmented. This chapter deals with a novel technique to extract (segment) individual text lines from such multi-oriented or curved nature of text in documents.

5.1.1 Motivation

If a document image only contains horizontal and vertical text lines with broad inter-line spacings, and if the shape of each text block in the document is represented by a square or a simple combination of squares, the text lines can be extracted by a very simple algorithm which is based on the iterative splitting of elements using global projection profiles [NSV92]. However, the method using global projection profiles cannot separate text blocks correctly if the text blocks have a complex shape or if they make inroads into other elements.

Extraction of individual text lines from artistic or technical documents having multi-oriented or curved text is a difficult problem. In graphical documents, the components of text lines can be of different sizes and font styles. Again, the inter-character distance varies time to time for annotation purpose. Examples of some such documents are shown in Fig.5.1. In Fig.5.1(c), the inter character distance in the word "OCEAN" of a map is different for annotation and it has different orientation in different location to annotate the graphical lines.

5.1.2 Related Work

There are many techniques to extract text lines from single oriented documents [LZT07], but the published work on extraction of multi-oriented and curved text lines are few.

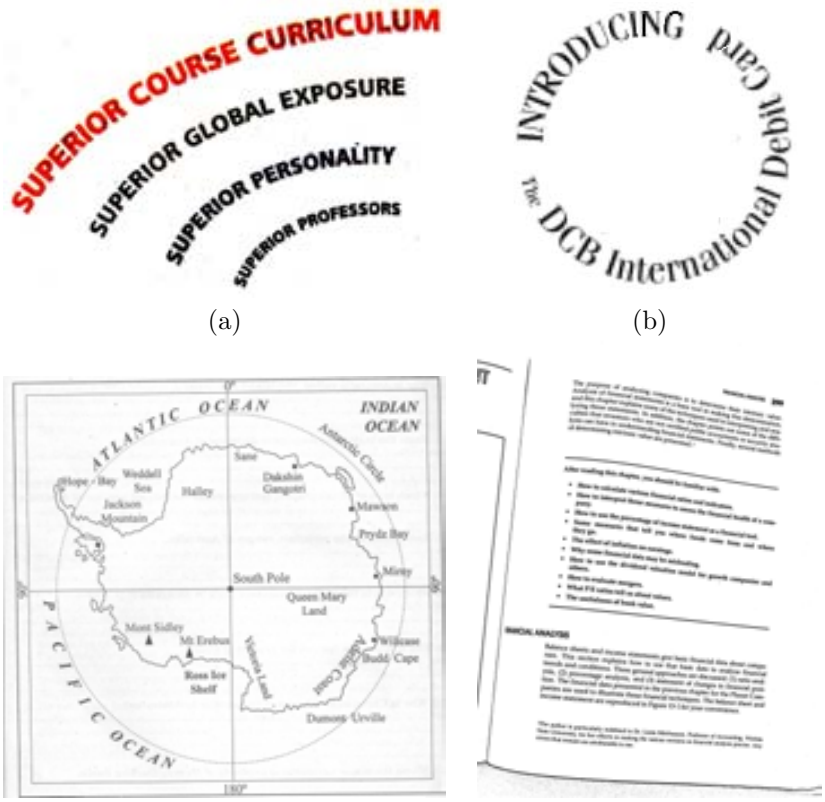
Li et al. [LZDJ08] proposed an approach for handwritten text-line segmentation using level sets. Here, Gaussian filtering is used to estimate the probability density function (PDF) of pixel values and then level sets are initialized on these high PDF values. Growing and merging of level sets is then performed iteratively.

Goto and Aso [GA99] proposed a local linearity based method called Extended Linear Segment Linking ($ELSL$). This is able to detect and extract text lines in arbitrary orientations and curved text lines. In this method, a text line is treated as a set of short linear segments, which has some advantageous properties, such as; being tolerant of the local skew of the text line and being able to handle documents with a complex layout.

In another method, proposed by Hones and Litcher [HL94], line anchors are first found in the document image and then text lines are generated by expanding the line anchors. These methods cannot handle variable sized text, which is the main drawback of the methods.

O'Gorman proposed a method which can extract text blocks with a complex shape [O'G93]. In this method, connected components of black pixels are first found in document image, bounding boxes are created, and then the bounding boxes are connected to each other to form text line candidates using k-th nearest neighbor clustering. However, methods based on the simple merging of bounding boxes require broad blank spaces as separators between different text blocks. Thus, such a method cannot handle documents in which text blocks are densely arranged.

Loo and Tan [LC02] proposed a method using irregular pyramids for text line segmentation. This algorithm uses the inclusion of background information, concept of "closeness", density of a word region, majority "win" strategy and the directional uniformity and continuity between words in a sentence.



(a) (b)

(c) (d)

Figure 5.1: Examples of documents containing multi-oriented and curved text-lines. (a) and (b) Text images from newspapers (c) Map image (d) Camera based warped image.

Gatos et al. [GPN07] proposed an algorithm based on text line and word detection for warped documents. Horizontal smoothening is done to combine characters into words. Next, based on word rotation and translation according to upper and lower word baselines, document is de-warped. To do this, words are selected in a top down scanning fashion. For each word, neighboring connected components are searched. The connected component with smallest distance is grouped with the selected word. The same process is repeated until no nearest neighbor is found.

Bai et al. [BNS08] used a traditional perceptual grouping based algorithm for extracting curved line from logos and slogans. Chains of connected components are extended to the left and the right according to the local orientations of the text lines.

Recently, Bukhari et al. [BSB08] proposed a line segmentation approach for camera based warped documents using active contour models. The segmentation technique uses several active contour models (baby snakes) and their convergence.

Pal and Roy [PR04] proposed a head-line based technique for multi-oriented and

curved text lines extraction from Indian documents containing Bangla and Devnagari scripts. Since the head-line feature is missing in English, this method cannot be used for English line extraction.

In other work Pal et al. [PSC03] developed a system for English multi-oriented text line extraction estimating the equation of the text line from the character information. Main drawback of this method is that, it will not work for documents of curved text lines.

In Table 5.1, we summarize the state-of-the-art approaches for text line extraction methods. As discussed above, the main problems for segmentation of curved text lines in graphical document images are high degrees of curve due to geometric and annotative nature. Hough transform based text line detection methods [FK88] will not be applicable in such documents due to curve-nature of text lines. Many previous curved text lines segmentation techniques use either connected component itself or its top-bottom points for text lines detection using nearest neighbor criteria [PR04, BSB08, BNS08]. Considering top-bottom points of connected components of a text line will not be useful in graphical document, because, text lines are not in horizontal direction in maps always. Text lines are annotated in horizontal, vertical or even in curvi-linear way according to the suitability. Use of irregular pyramids [LC02] may also fail many times, as the inter-character spacing in words are not uniform always.

Table 5.1: Related work proposed on text line segmentation.

Method	**Key Feature in the Proposed Algorithm**
Li et al. [LZDJ08]	Level Sets
Goto and Aso [GA99]	Extended Linear Segment Linking
Hones and Litcher [HL94]	Line anchors
O'Gorman [O'G93]	Nearest neighbor clustering of connected components
Loo and Tan [LC02]	Irregular pyramid structure
Gatos et al. [GPN07]	Upper and lower baselines of word
Bai et al. [BNS08]	Local orientation
Bukhari et al. [BSB08]	Active contour models
Pal and Roy [PR04]	Head-line

5.1.3 Outline of the Approach

To handle documents of wide variations in terms of size, font, orientation, layout, etc., we propose a technique based on the foreground and background information of the characters in text line. Most researchers focus their attention only on foreground components and discard the background part (background part refers to those white areas of a binary image surrounding the actual black foreground). Here, we demonstrate how the background information can be utilized for multi-oriented line extraction.

The proposed approach is based on perceptual properties of text components. It finds the sequence of text characters using size and spatial information, and later, includes them in line according to the background information. Text is a sequence of cavities following a regular distribution in a smooth path. These cavities/background

information plays an important role in our proposed method and guides our algorithm to extract individual lines from the documents containing multi-oriented and curved text lines. We use the background portion obtained between two consecutive characters of a line. To get this background portion we apply the water reservoir concept. Water reservoir is a metaphor to illustrate the cavity region of a component [PBC03]. The use of background information and its computation through the water reservoir concept guides our proposed algorithm to segment text lines. In the proposed scheme, at first, individual components are detected and grouped into 3-character clusters using their inter-component distance, size and positional information. Merging these 3-character clusters, a proximity graph is created to have larger clusters. Using inter-character background information, orientations of the extreme characters of a larger cluster are determined, and based on these orientations, two candidate regions are formed from the cluster. Finally, with the help of these candidate regions, individual lines are extracted.

The organization of the rest of the chapter is as follows. A brief discussion on water reservoir concept used for line extraction is given in Section 5.2. The line extraction procedure is detailed in Section 5.3. We demonstrate our proposed algorithm on a variety of datasets including graphical documents and camera based documents in Section 5.4. Conclusion and future work are presented in Section 5.5.

5.2 Water Reservoir Concept

The water reservoir principle, originally proposed by Pal et al. in [PBC03] is as follows. If water is poured from a side of a component, the cavity regions of the background portion of the component where water will be stored are considered as reservoirs of the component. Some of the water reservoir principle based features used in our proposed scheme are as follows.

- Top (Bottom) reservoir: By top (bottom) reservoirs of a component we mean the reservoirs obtained when water is poured from the top (bottom) of the component. A bottom reservoir of a component is visualized as a top reservoir when water is poured from top after rotating the component by 180°.
- Left (Right) reservoir: If water is poured from the left (right) side of a component, the cavity regions of the component where water will be stored are considered as left (right) reservoirs. A left (right) reservoir of a component is visualized as a top reservoir when water is poured from the top after rotating the component by 90° clockwise (anti-clockwise).
- Water reservoir area: The area of a reservoir is defined by the area of the cavity region where water will be stored. The number of points (pixels) inside a reservoir is computed and this number is considered as the area of the reservoir.
- Water flow level: The level from which water overflows from a reservoir is called the water flow level of the reservoir (see Fig.5.2).
- Reservoir surface width: The width of the reservoir at the flow-level of the reservoir is the reservoir surface width. In Fig.5.2, AC is the surface width of

the reservoir.

- Mid-Point of water flow surface: This is defined as the mid-point of the reservoir surface width. In Fig.5.2, M is the mid-point of the water flow surface.

- Height of a reservoir: By height of a reservoir, we mean the depth of water in the reservoir.

These background regions based features obtained using the water reservoir concept help our line extraction scheme.

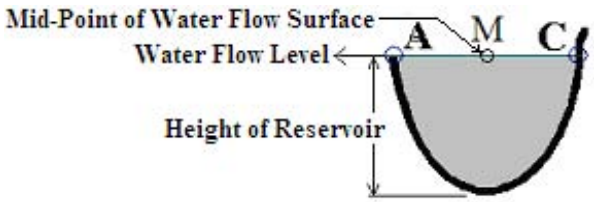


Figure 5.2: A top water reservoir and its different features are shown. The water reservoir is marked in grey.

5.3 Text Line Extraction

An English text line can be divided into 3 zones: upper zone, middle zone or busy zone and lower zone. The busy zone for an English text line is the zone between the mean-line and the base-line as shown in Fig.5.3(a). Busy zone is shown for curve line in Fig.5.3(b). This height information is important for the line extraction algorithm. It is used in our approach to determine the search zone (explained later) for the multi-oriented text line extraction.

In our line detection algorithm, we assumed that the inter-character distance in a word of a line is smaller than the distance of this line from its neighbouring lines. The proposed line detection process is divided into 4 steps. These steps are

1. Initial 3-character clustering.

2. Grouping of 3-character clusters to form larger cluster.

3. Candidate point selection of large clusters and

4. Extension of large clusters for line extraction.

The flow chart of text line segmentation is given in Fig.5.4. The main part of the line segmentation approach is illustrated within the dashed rectangle. In the following sections, we detail each step.

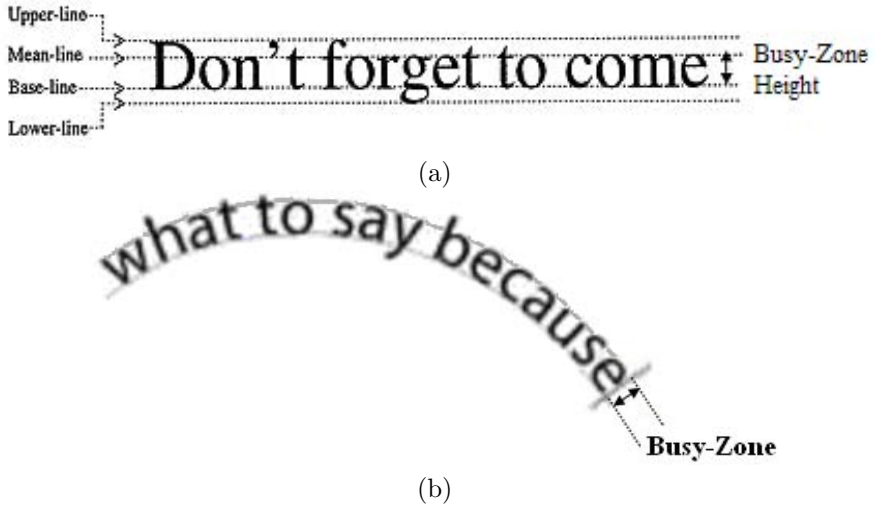


Figure 5.3: Different parts of a text line shown in (a) straight line and (b) curve line.

5.3.1 Initial 3-Character Clustering

For character clustering, at first, component labeling is performed. Let, V_i^C be the center of minimum enclosing circle (MEC) of the component C_i. The size (l_{Ci}) of C_i is computed from the radius of MEC of the component.

For each component (say, C_1) we find two nearest components using a boundary-growing algorithm (discussed later). Let the two nearest components of C_1 be C_2 and C_3. The components C_1, C_2 and C_3 will form a valid 3-character cluster if they satisfy the following:

1. Similar in size and
2. Locally linear

The size similarity is tested based on the components' sizes. The component C_1 will be similar in size to its neighbor component C_2, if

$$l_{C2}/1.5 \leq l_{C1} \leq 1.5 \times l_{C2}$$

Local linearity is tested based on the angular information of V_1^C, V_2^C and V_3^C. If the angle $\angle V_2^C V_1^C V_3^C$ formed by the points V_2^C, V_1^C and V_3^C is greater than T_{ang} then we assume these 3 components are linear in nature. T_{ang} ($150° \leq T_{ang} \leq 180°$) is empirically chosen from experimental results.

Initial 3-character clustering of the image Fig.5.5(a) is shown in Fig.5.5(b). Each of these 3-character clusters is marked by circular ring. From the figure, it is to be noted, there is no cluster among the components (o-p-e) and (r-l-i) because they fail to satisfy the angular criteria. The clusters (r-s-C), (s-C-r) and (n-g-R) are not formed because the inter-character spacing is large.

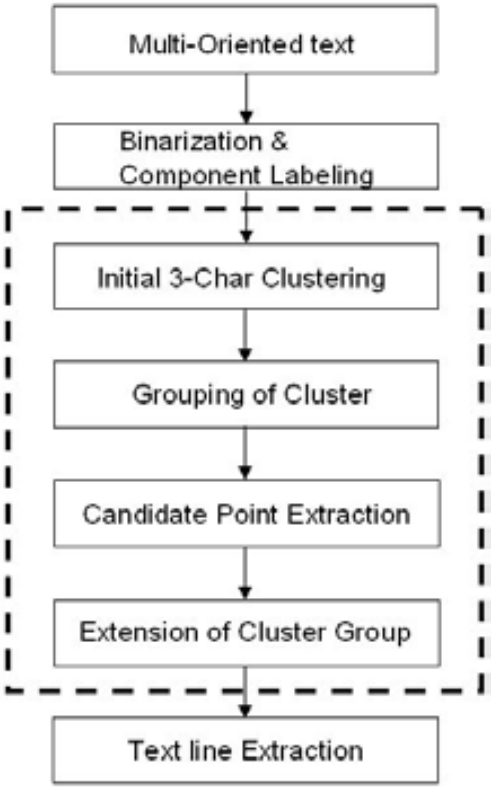
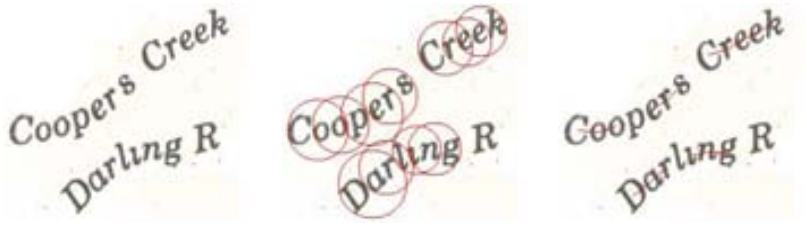


Figure 5.4: Flow chart of our approach for curved line extraction



(a) (b) (c)

Figure 5.5: (a) Image shows different words from a map. (b) Initial clusters are marked by grey circular rings. (c) Large clusters are shown by a grey line.

Boundary growing is performed as follows. The external contour points of each character component are detected by a contour tracing algorithm. These contour points are expanded outwards iteratively by one pixel (8 pixel neighbor configuration) for boundary growing. Number of iterations to find nearest components is decided considering the size of the characters of the word. It is decided as $I_C = q \times l_C$. The multiple factor q is selected based on the type of dataset. In text documents, q is

set to 4. If document contains graphical lines, the value of q is set to 6. Here, text characters are more sparse compared to normal text documents. These values are set by the experiment. If a character component does not touch other components after I_C iterations of boundary growing, then no neighbor is considered to be found for the character.

5.3.2 Grouping of Initial Clusters

From the initial clustering, several 3-character clusters are obtained which will be grouped together by growing. From the second text-line of Fig.5.5(b), we have different 3-character clusters such as (D-a-r), (a-r-l), (l-i-n) and (i-n-g). A graph analysis is performed to grow these clusters. A graph (G_{cl}) is initialized with all these components of different clusters, where components of all the clusters are considered as nodes. An edge between two component nodes is added if they are from the same initial cluster.

Sometimes, by adding edges in such a way, a node may have 3 or more edges. This situation occurs when two or more text-lines cross each other or they are very close. The nodes having 3 or more edges are not considered in this step. For illustration, see Fig.5.6. Here, the character 'h' of the word '*Algorithm*' has two neighbors 't' and 'm' from this word. Similarly, the character 's' of word '*analysis*' has neighbors 'i' from this word and 'h' from the word '*Algorithm*'. After grouping into initial clusters, 'h' will have 3 edges which are connected to its 3 neighbors ('h-t', 'h-m' and 'h-s'). This node 'h' is marked for removal and is not considered for initial cluster grouping, since it may generate erroneous results in line extraction.

Figure 5.6: Example of text when a character may have 3 neighbors. Here, the character 'h' has 3 neighbouring characters 't', 'm' and 's'.

The angle of each node (of degree 2) is calculated with respect to two connected nodes. If the angle is less than T_{ang}, this node is also marked for removal. For example, in Fig.5.5(c), the component 'l' of (D,a,r,l,i,n,g) does not satisfy the angle criterion with its neighbor components 'r' and 'i'.

Thus, if for a node, either number of edges is greater than 2 or its angle with neighbor characters is less than T_{ang} (explained earlier), then the node is removed and all the edges of the corresponding node are deleted from G_{cl}. Thus, local linearity criterion between each of 3 character clusters is validated. Because of removal of some

edges from the graph, G_{cl} will be split into sub-graphs or a set of components. In each of these sets, we will have a chain of components which are locally linear. In other words, paths of characters are searched in the graph according to piecewise linearity criterion. Each sub-graph is considered as a large cluster.

These large clusters represent different parts of text lines. These clusters are grown to merge other clusters in the same text line. Since, the clusters in the same line can be in different orientations, we use the cluster growing in local curvi-linear direction. The continuation path of each cluster is followed by extending two extreme sides of the cluster. Hence, the orientation of the extension is computed from the extreme characters of the cluster group. For this extension purpose, we select candidate points from both ends of each cluster to follow the orientation. The selection of candidate points and the extension of the cluster are described in the following.

5.3.3 Candidate Point Selection

Using inter-character background information, we decide the orientation of the extreme characters of a cluster group. For each cluster group we find one pair of characters from both of the two extreme sides of the cluster. Let, 'NG' and 'IN' be two such pairs of extreme characters of a cluster group and these pairs are shown in the 1st row of Fig.5.7. To find background information, the water reservoir concept is used. To do so, first convex hull of each character is found (a convex hull is shown in the 2nd row of Fig.5.7) and this is done to fill up the cavity regions of the character, if any. Next the resultant components are joined (shown in 3rd row of Fig.5.7) by a straight line through their center of MEC (Minimum Enclosing Circle). This joining is performed to make the character pair into a single component to find the water reservoir in the background part between them.

Subsequently, the water reservoir area of this joined character is computed in 8 directions at 45° intervals as shown in Fig.5.8. Computation of the water reservoir when water is poured in the D_7^W (D_3^W) direction is equivalent to the computation of top (bottom) reservoir. The areas of water reservoirs in direction D_3^W and its opposite direction D_7^W are added to get the total background area with respect to the orientation $D_3^W D_7^W$. This is done for other orientations $D_1^W D_5^W$, $D_2^W D_6^W$ and $D_4^W D_8^W$. The orientation in which the maximum area is found, is detected and water flow-lines of the corresponding reservoirs are computed. The mid-points of the water flow-lines of the two reservoirs are the candidate points. For illustration, see the last four rows of Fig.5.7, where water reservoirs in four orientations ($D_1^W D_5^W$, $D_2^W D_6^W$, $D_3^W D_7^W$ and $D_4^W D_8^W$) are shown. From the figure, it can be noted that, the maximum reservoir area is found for the pair 'NG' in $D_3^W D_7^W$ (top-bottom) directions whereas for the pair 'IN', the maximum reservoir area is obtained in the $D_1^W D_5^W$ (left-right) direction. This is because of the different orientations of these two extreme pairs of the cluster "INTRODUCING".

For a pair, we get two candidate points and an orientation. The candidate points of the character pairs 'NG' and 'IN' are shown in Fig.5.9. The orientation is obtained from the directions in which we get a maximum reservoir. For example, if we get a maximum reservoir area in the D_3^W and D_7^W directions, then $D_3^W D_7^W$ is the orientation. This orientation represents the direction of the extreme characters of a cluster,

and it helps us to extend the cluster group for text line extraction. Generally, cluster groups should be extended in perpendicular to this orientation to obtain its neighbouring clusters of a text line. We also compute the distance between two candidate points and this distance gives the height information of the busy zone of the text line.

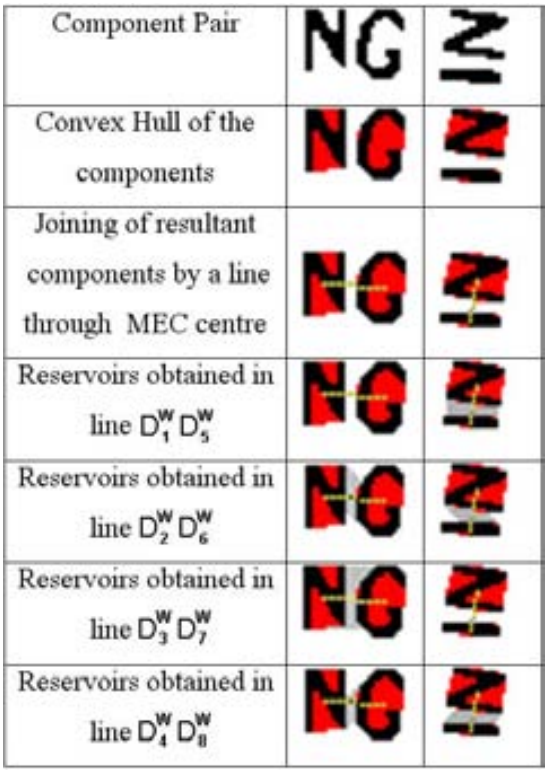


Figure 5.7: Water reservoir computation is shown on two pairs of characters. The convex hull is marked by red color and the water reservoir area is marked by grey.

5.3.4 Extension of Cluster Groups

For each extreme character pair of a cluster group, we can know its candidate points and orientation. Let two candidate points of one extreme side of a cluster be M_{cl}^1 and M_{cl}^2 and the oriented direction be D_{cl}. The distance between $M_{cl}^1 M_{cl}^2$ is noted and this distance is called height info (HI). We find a line which is perpendicular to D_{cl} and passing through the mid-point of M_{cl}^1 and M_{cl}^2. Let this line be XY and we call it the estimated line of extension. Note that, this line is detected based on the information of the extreme characters of a cluster and as a result, our line extraction scheme gives good results. Now, the line XY is extended in an outward direction until it reaches the bounding box of the cluster group. The point where the extended line meets the bounding box is noted and we call it a key point. This is done for the other parts of the other extreme sides of the cluster and we detect another key-point. So,

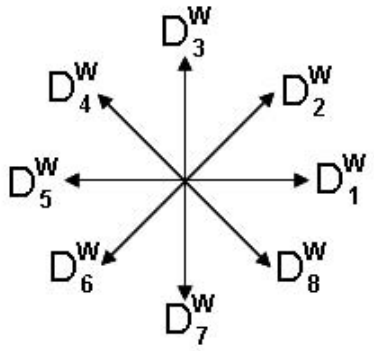


Figure 5.8: Directions of water reservoir computation.

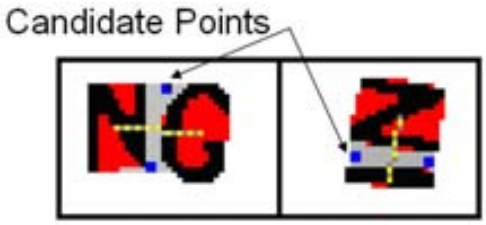


Figure 5.9: Candidate points are marked by dot.

for a cluster we have two key-points. Key points of the cluster "INTRODUCING" are shown in Fig.5.10(c) and marked by K_{cl}.

Now, for each extreme side of a cluster we have the following: (a) estimated equation of the line of extension and (b) a key-point (K_{cl}). For cluster extension to extract a line, we generate a candidate region for each key point of a cluster. Candidate region generation is performed as follows.

Generation of candidate regions

Estimated equation of line of extension is extended up to a distance $TR = q \times HI$ in the outwards direction from the key point K_{cl}. The path obtained during this extension is called as extended candidate path of K_{cl} of the cluster. The value of q is discussed in Section 5.3.1. A rectangular mask of length $q \times HI$ and width of HI is placed on the extended candidate path in such a way that the mask will be divided into two halves by the extended candidate path. This region of the rectangular mask is considered as the candidate region of the cluster for the key point K_{cl}. In a similar way, the candidate region of the cluster for the other key point is computed. Candidate regions for two key points of a cluster "INTRODUCING" are shown in Fig.5.10(c) and they are marked by dashed line.

The reason to choose a TR equal to $q \times HI$ is as follows. In printed text HI generally represents the busy-zone height of a text line. So, if we assume TR as $q \times HI$ it is most

likely that a part of the neighboring cluster will fall in one of the candidate regions of the current cluster. Using the candidate regions, lines are extracted.

Line Extraction

Candidate regions are used for line extraction and line extraction is performed as follows. Let U_S be the set of candidate clusters and isolated characters of an input image. We use a bottom-up approach for line extraction and the approach is as follows. First, an arbitrary cluster (say, topmost left cluster) is chosen from U_S and a line-group L_g is formed using this cluster. Two candidate regions are detected from this cluster. For each line-group we maintain two candidate regions (CR): left and right CR. Next, we check whether there exists any extreme characters of a cluster or individual component whose portion falls in these CRs of the line-group. From U_S, we get a cluster Cl of which an end character falls in CR. If the orientation of that end character of Cl and that of corresponding CR is similar and their corresponding HI are also similar, then we include it in L_g. The candidate regions of Cl are noted and the CRs are modified accordingly. The CRs are modified as follows. If the right (left) candidate region of Cl falls in the left (right) CR of L_g then the left (right) candidate region of Cl is assigned as left (right) CR of L_g. Sometimes a portion of some isolated characters of may also fall in the CRs. We also include such isolated characters in L_g. If a portion of an isolated component falls in any CR of L_g, the component will be selected to join the line-group L_g, if the size similarity and linearity condition discussed in Section 5.3.1 are satisfied. If an isolated character is included in L_g, then the corresponding CR is updated considering the included character as the extreme character of the line-group. The extension of this line-group continues as both sides, till it does not find any component or cluster in any CR or it reaches the border of the image. Algorithmic steps of the extension of clustering group of the proposed scheme are given in Algorithm 2.

Components clustered into a single line-group are the members of a single text line. To get other text lines we follow the same steps and finally we get N_T number of line-groups if there are N_T text lines in a document. In pre-processing of line extraction, small symbol components (‘.’, ‘,’ etc.) and background noise were filtered out based on their aspect ratio and pixel density. The isolated symbols are later restored after line extraction according to the proximity relation. To include these isolated components in the line-group of their respective text lines we use a boundary growing technique explained (See Section 5.3.1). An isolated component is grown along its border until it touches any component of a line. The isolated component is included within the line, which it touches first. The whole process is described in Fig.5.11.

5.4 Experimental Results

For the experimentation of the present work, we considered data from different sources e.g., maps, newspapers, magazines, seal documents, etc. as well as synthetic data generated through the computer. The datasets and the metric for performance evaluation are discussed in the following sub-sections.

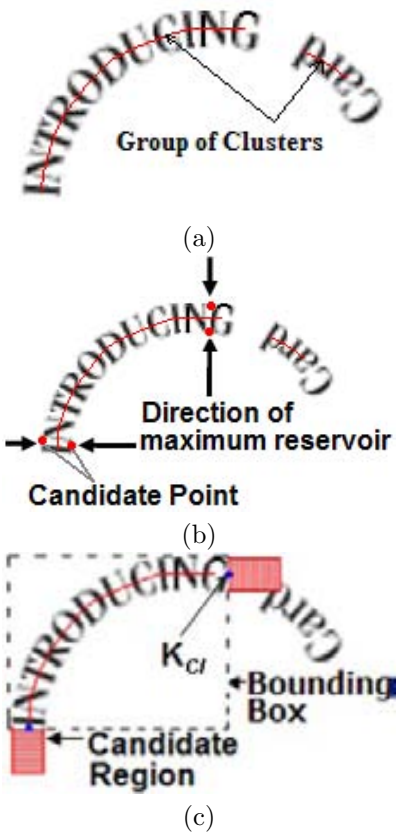


Figure 5.10: Example of candidate region detection from a cluster. (a) Two clusters "INTRODUCING" and "Card" are shown. (b) Candidate points of the two extreme pairs of characters for the cluster "INTRODUCING" are shown. (c) Key points and candidate regions of cluster "INTRODUCING" are shown. Key points are marked by 'K_{cl}'. The candidate region is marked by dashed-line box.

Metric used for Experiment

To check visually whether a text line is extracted correctly or not, we render all components that are clustered in an individual line by a single color. The color of different text lines is selected with a random function generator. To give an idea about different ranges of accuracy of the system on different types of documents, we divide the accuracy into three categories: (a) 100% (b) 95-99.9% and (c) $\leq$ 95%. The accuracy of the line extraction module is measured according to the following rule. If out of N_C components of a line, M_C components are extracted in favor of that line by our scheme, then the accuracy for that line is $(M_C \times 100)N_C\%$. So if all components of a text line are extracted correctly by the proposed algorithm, we say that the accuracy of the line is 100%. Different datasets, used in our experiment and their results are discussed as follows.

Algorithm 2 Extension of cluster group (G_c).

Step 1: Let M_{cl}^1 and M_{cl}^2 be the candidate points for one extreme side of a cluster (G_c) and D_{cl} be the orientation of line obtained by joining M_{cl}^1 and M_{cl}^2. Let HI be the distance between M_{cl}^1 and M_{cl}^2. An equation XY is estimated perpendicular to D_{cl} and passing through the mid-point of M_{cl}^1 and M_{cl}^2.
Step 2: The intersection point of XY and bounding box of G_c is detected to find the Key-Point (K_{cl}) as shown in Fig.5.10(c).
XY is extended up to length $TR = q \times HI$ in the outwards direction from the key point K_{cl}, q is the multiplying factor of neighbor search area to get candidate region.
Step 3: A Candidate Region (CR) of rectangular size ($TR \times HI$) is considered at K_{cl} as shown in Fig.5.10(c).
Step 4: If extreme character of another cluster falls in CR and the orientation of the candidate points of the cluster is also similar to the orientation of D_{cl} of G_c, then the cluster is included with G_c. Instead of a cluster, if an isolated character falls in CR, then it is also included in G_c. Update G_c to get new M_{cl}^1, M_{cl}^2 and D_{cl} from the resultant cluster. If no cluster is found to include with G_c then 'exit'. Else goto Step 1.

5.4.1 Text Documents

Our first experiment includes degraded text documents where text lines are in horizontal directions. In this scanned dataset, the text lines are parallel to each other. Due to degradation, characters in text-lines are broken sometimes. We considered 50 such documents to test. Some of the results of text line separation in such images are shown in Fig.5.12. Detailed performance accuracy is given in Fig.5.20.

5.4.2 Warped Documents

Camera-captured document images due to its unconstrained hand-held, usually contain several types of degradations which are not common in scanned images, like perspective and geometric distortions. Thus, camera-captured document images contain non-straight text lines with high-degrees of curve in different directions. These warped document images reduce human readability as well as OCR accuracy when traditional OCR engines developed for scanned documents are used (See Appendix A.3 for details).

From this dataset, we considered 30 images containing text data. Some of them include graphical diagrams also. As our system can take care of graphical diagrams, we used these documents without any other pre-processing and evaluated our approach on these documents. Some of the results of curve text line separation are shown in Fig.5.13.

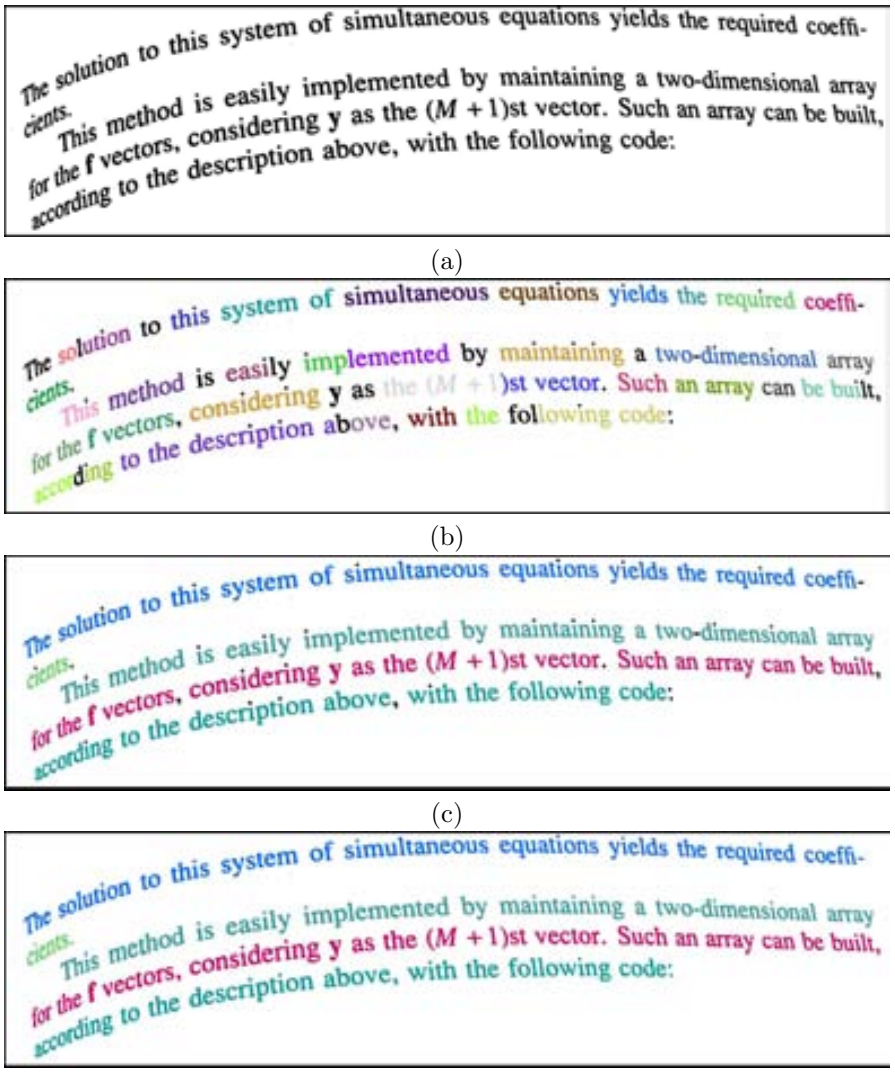


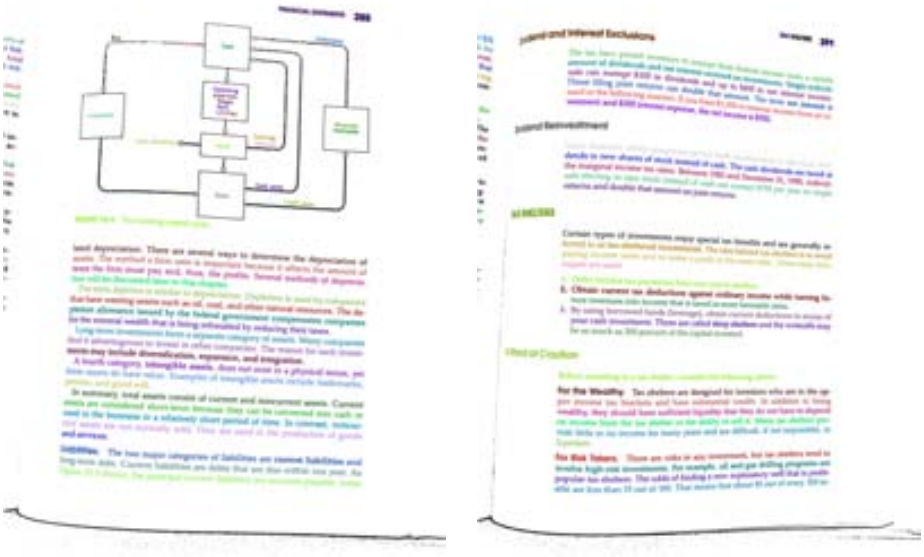
Figure 5.11: Different steps of our proposed algorithm are shown with a given camera based warped document shown in (a). (b) Large clusters. (c) Large clusters are merged to form text lines using water reservoir concept. (d) Other punctuations/small components are included in their respective text lines.

5.4.3 Graphical Documents

We have considered geographical maps and electrical diagram images to test our method. There are total 20 maps and 6 electrical diagrams considered. Images were digitized by a flatbed scanner at 300 dpi. The synthetic maps are also included (See Appendix A.5.1 for more details). These images contain text lines of different scale

(a) (b)

Figure 5.12: (a) and (b) Normal text document.



(a) (b)

Figure 5.13: (a) and (b) Camera-based warped text documents.

and orientation. Some real geographical maps are selected from historical archive. See Fig.5.14, where a portion of the document is shown. These images contain text printed in dark color compared to the background.

Text Layer Extraction from Color Image: To obtain the text layer from color image, we get foreground information by converting the color image to a gray-level one and apply a threshold to this image for segmenting. The conversion to gray level is

performed by transforming the RGB color model to YIQ model, where the luminance channel (Y) represents the gray-level image. It is achieved by the conversion equation,

$$Y = 0.299 \times R + 0.587 \times G + 0.114 \times B$$

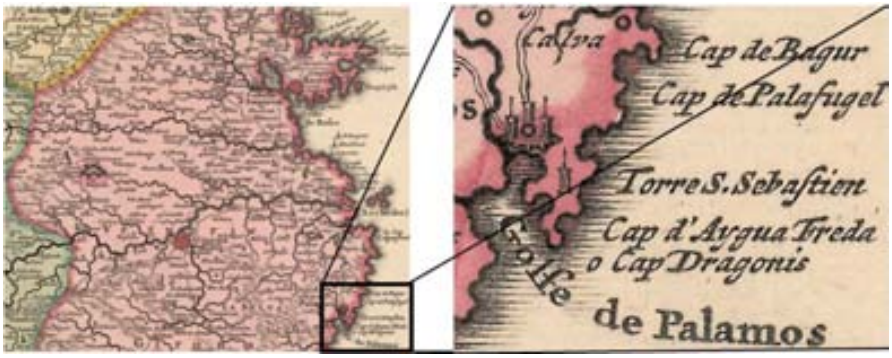


Figure 5.14: A real historical map showing its curvi-linear text lines in a zoomed window.

Next, by applying an adaptive threshold to the gray image (implemented in OpenCV[1]), we separate the foreground region. In maps, long graphical lines touch with text characters in some places. To extract these text character components, a simple heuristic based on connected component analysis is applied to remove long components. This is discussed in Chapter 4 (Section 4.2). The resultant image of text characters is fed to our system for line segmentation.

Due to connected component based analysis, some text components that are touched by graphical lines may be removed. Thus, all text characters may not be extracted using this method. Also, some small graphical objects which are not text components appear in the document. To get an idea of our text line extraction results in graphical documents, some of the images are shown in Fig.5.15.

Most of the errors in graphical documents occurred due to missing of character components in their corresponding lines. It is noticed that, if some characters are in middle of a word, then our algorithm can extract the rest of the characters and group them in a single line. For example in Fig.5.15(b) the word "MEDITERRA_EAN SEA" is recovered though some of its characters like 'N' in the middle are missing. Using some post-processing steps such as top-bottom line fitting of the text-line, the missing characters of the words could be found. We get some false positives in maps due to the presence of small components (non-text) in a line. In the electrical diagram dataset, the errors occurred when words were located very close together. As we are not considering the graphical information which is separated from the text portion, the boundary/box information of the text is ignored. Due to this reason, the words from different text blocks are sometimes combined together.

[1] www.intel.com/research/mrl/research/opencv/

(a) (b)

Figure 5.15: (a) and (b) Text lines from two different maps are shown.

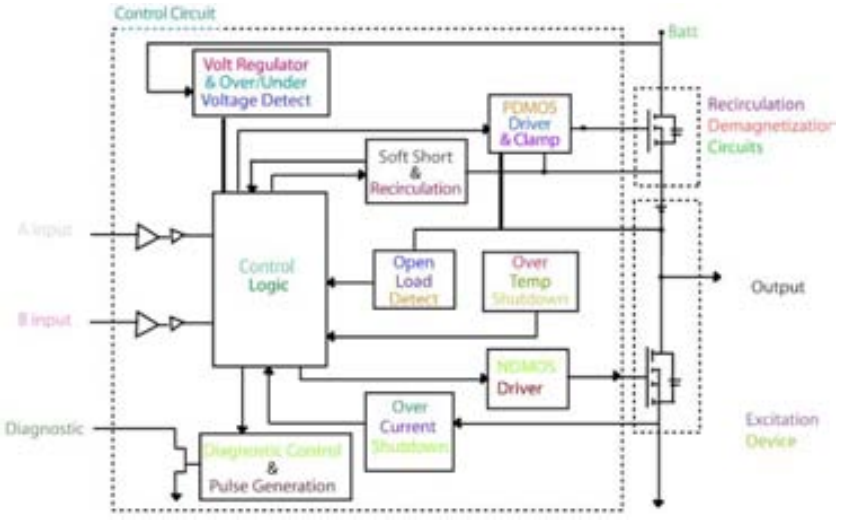


Figure 5.16: Extracted text lines from electrical diagram.

5.4.4 Seal Documents

For experiment with real noisy data, documents containing seals were considered. These were collected mainly from historical archive. In these documents, text lines are in different orientations and they overlap with seal many times. The documents were digitized by a flatbed scanner at 200 dpi. In these documents, the characters are frequently touching due to background noise, which creates problems for the

extraction of isolated text characters (See Appendix A.5.3 for details).

We tested a database of 50 documents containing seals of English characters. Some of the results are shown in Fig.5.17 and Fig.5.18. Fig.5.17 demonstrates the text line extraction in portion of seal documents. In Fig.5.18, we show the separate lines in isolated seal images. In seal dataset, text line segmentation is affected mainly due to broken text components in the seal and the presence of long graphical lines over text regions. Hence, the accuracy is low compared to other datasets. Many text lines are over segmented and could not be joined together with the proposed system. Sometimes, errors occurred when a cluster end is very close to another cluster, which is a member of a different text line.

(a) (b)

Figure 5.17: Part of seal documents.

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

Figure 5.18: Isolated seal symbols.

5.4.5 Direction Sign Boards

Our next experiment includes camera-based scenes containing direction signs. The images are captured in color tone with 3 mega pixel digital camera. The images

are captured in such a way, that the text lines in the direction board are in affine transform (See Appendix A.4 for details). We have considered 10 different direction boards for this experiment.

To obtain the text image, we get foreground text layer by converting the color image to a gray-level one and applying an adaptive threshold to this image (See Section 5.4.3). As our system does not include any algorithm to remove scene portions from the image, we have many false positives due to their presence. We show some results in the following. When a text line is composed of two characters only (e.g. "30" in Fig.5.19(c)), our approach can not extract them. It is due to the assumption that there will be at least 3 characters in each line. And our initial character clustering step fails to use them. It is to be noted that in Fig.5.19(d)-(e)-(f) we have shown our line segmentation approach in 3 different affine transformations of a natural scene image of Fig.5.19(c).

5.4.6 Discussion and Error Analysis

The digitized images may contain spurious noise points, small break points and irregularities on the boundary of the characters leading to undesired effects on the system. Since our method is based on cavity, our method may not work if there is a broken part in the character. In this case, neighborhood component selection will not be proper due to the mismatch in size similarity feature. Hence, direction from the water reservoir concept cannot determine the candidate region properly and errors occur. Hence we used a method to join the broken parts using the algorithm due to Roy et al. [RPC04]. If there is a small broken part in the component this algorithm can join the broken part and our proposed method work well.

We show in Fig.5.20, the text line segmentation accuracy in different datasets used in our experiment. In the datasets of normal scanned text documents, the text lines are segmented very well. In CBDAR dataset, the warped lines are segmented properly and separated from graphical portions if they exist. The orientation of text line information can be useful in future for de-warping the document. In maps, when the characters of a word are distantly spaced, then our line extraction method sometimes fails to join the cluster ends due to threshold selection. In direction board images, we have got many false positives due to improper foreground extraction from color scene image. A proper color segmentation scheme and text layer separation approach would help to improve the performance.

5.5 Conclusion

We have presented a new approach for text line segmentation. A key characteristic of our approach is the use of foreground and background information. The background information is taken care using water reservoir approach which guides our approach to follow the text line. The approach is based on perceptual laws of text characters in a line and it is easy to apply.

We have demonstrated our approach of invariance towards rotation, scale, affine to real world images. We demonstrated the robustness of this approach using different datasets with high degree of curvi-linearity. For this purpose, we considered camera

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

Figure 5.19: Two direction signs in (a) at a distance apart showing distinct text lines in (b). A direction board (c) and its 3 different affine transformed images are shown in (d), (e) and (f).

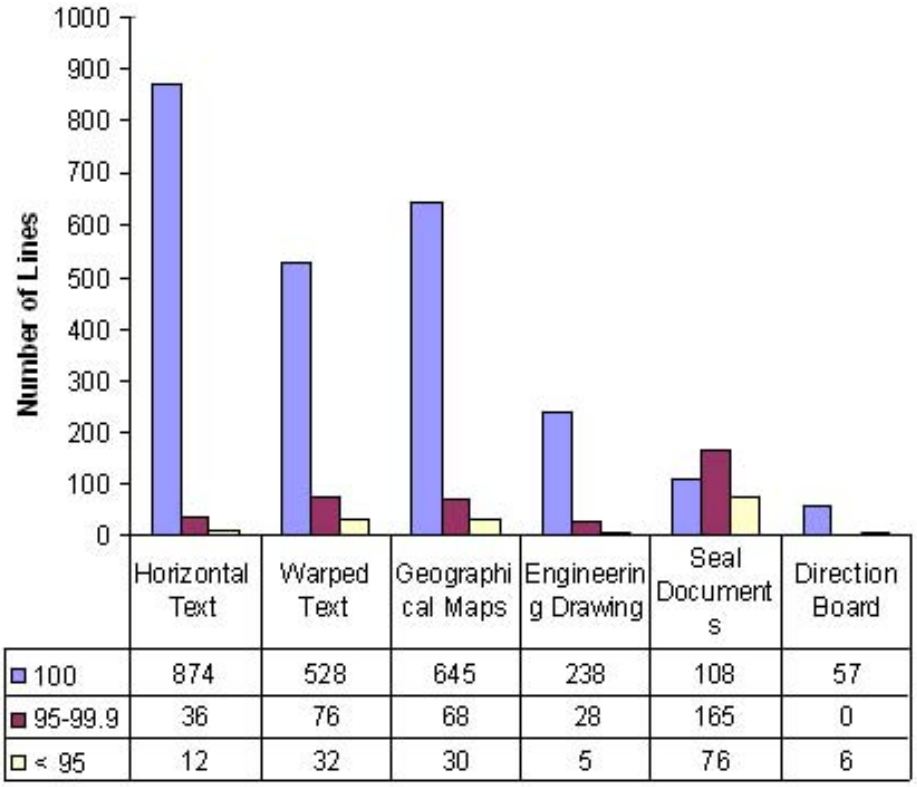


Figure 5.20: Text line segmentation accuracy of our proposed method from different documents.

based documents, graphical documents (map, electrical diagram) and seal documents. Our experimental results show that the approach can be used without de-warping or rotating the documents. One of the significant advantages of the proposed method is its flexibility. Our scheme is independent of font, size and style of the text characters.

Till this point, we have presented techniques to extract text information from graphical documents. In the next chapters, we will present two different approaches of graphical document retrievals. These approaches work based on user query namely: textual word and seal. In the next chapter, the document retrieval methodology based on user query word will be discussed.

Chapter 6

Query Driven Word Retrieval in Graphical Documents

In Chapter 5, we presented an approach for text line extraction using foreground /background information. In this chapter, we move one step forward and search graphical words on the fly by segmenting and validating them at the same time. The proposed approach uses recognition results of individual components to form character pairs with the neighboring components. An indexing scheme is designed to store the spatial description of components and to access them efficiently. Given a query text word (ASCII/unicode format), the character pairs present in it are searched in the document. Next the retrieved character pairs are linked sequentially to form character string. Dynamic programming is applied to find different instances of query words. A string edit distance is used here to match the query word as the objective function. Experimental results show that, the method is efficient to locate a query word from multi-oriented text in graphical documents.

6.1 Introduction

In the last chapters, we have studied different state-of-the-art approaches for different modules of graphical document interpretation. We have also presented different algorithms towards the analysis and segmentation of text information in graphical documents of multi-scale and multi-orientation environment. For the interpretation of graphical documents, text characters are separated and recognized. To include those characters in text lines, we have used perceptual rules. In this chapter, we tackle this problem with a different approach. Here, we move one step forward and search graphical words on the fly by segmenting and validating them at the same time.

As discussed in Chapter 1, with the rapid progress of research in document image analysis and understanding many applications are coming up to manage the paper documents in electronic form to facilitate indexing, viewing, extracting the intended portions, etc. These applications include documents that to be digitized and stored in

a database. In machine readable documents searching, the retrieval of documents is performed according to user queries which consist of collection of words. The records in documents are indexed with the traditional Optical Character Recognition (OCR) techniques which recognize character-by-character.

In an attempt to avoid the tedious process of scanning and OCR involved in obtaining testing collection, word spotting looks for different instances of the query image in document. In word spotting, given a single instance of a query word image, the system has to return a ranked list of segmented locations where the queried word is probable to be found. Here, the segmentation and ranking are performed using a matching process between query and target images by image based features. When a query word is given in textual format, such word is converted to image format and spotting is done accordingly. For this purpose, the textual query is translated into one or more word images (using different fonts/noise models) and later features from each of these images are compared with target images for retrieval [MMS03]. In graphical documents, this methodology may not be applicable, because characters may present in different fonts in a single document and their inter-character spacing is not known a priori. Also, due to curvi-linearity of the characters in a word, generation of such synthetic words may not be possible. Using symbolic encoding of text characters in such documents and then use of approximate string matching [Tak01] of such character group can be a good strategy to view ranking queries in such unstructured documents.

6.1.1 Motivation

As explained in previous chapters, the text characters appear in graphic rich documents along with graphical objects such as engineering drawings, boundary lines, rivers, etc. The interpretation of graphical documents does not only require the recognition of graphical parts but the detection and recognition of multi-oriented text characters and strings.

In graphical documents, text lines appear frequently in different orientations other than the usual horizontal directions. Sometimes, the text lines are curvi-linear. The purpose of such orientation and curvi-linearity is to annotate the location of graphical objects. Thus, the inter-character spacing differs according to annotation style. The OCR systems available commercially can not be applied in handling them. The problem for detection and recognition of such text strings is many-folded. Hence, the recognition of character strings/words in such document is difficult due to the usage of multi-oriented and multi-scale environment. We show a portion of map in Fig.6.1 to illustrate the problems. It can be seen from Fig.6.1, the string “Mahabharat Lekh” is oriented in different directions in two different instances. In another word “INDIA”, it can be seen that the inter-character spacing is much larger compared to that in other words.

6.1.2 Related Work

The presence of graphical objects (such as long lines) and their overlapping with text strings sometimes make text information retrieval difficult. Indexing of words in such

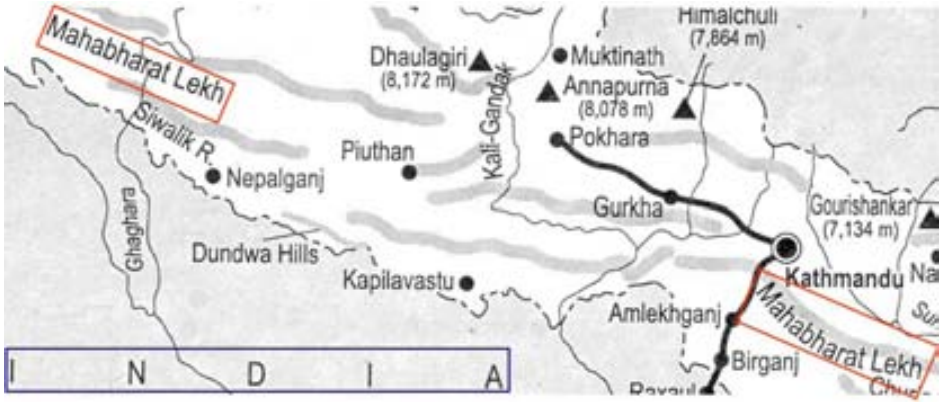


Figure 6.1: Example of a map shows different orientation of a string "Mahabharat Lekh". The strings are marked by box. Another string "INDIA" which demonstrates the inter-character spacing variation, is marked by a rectangular box.

documents needs to take care of text/graphics separation and to handle curvi-linear strings.

Deseilligny et al. [DMS95b] proposed an algorithm based on dynamic programming which enables to construct optimal strings by using constraints on orientation homogeneity. A systematic connected-character search is employed at each extremity of detected strings. This searching is done to find a character with the similar orientation. Furthermore, it requires precise information about the font used during the drawing process.

Leyk et al. [LBW06] developed a multi-step recognition process for extracting compound forest cover information from manually produced scanned historical topographic maps. Based on salient features in the images, different steps like, character recognition, graphical line detection and structural analysis of forest symbols are carried out. The text characters are considered here in horizontal direction.

As we presented already in previous chapters, there are many works in different sub-stages towards graphical document retrieval system.

Algorithm due to Fletcher and Kasturi [FK88] used simple heuristics based on the characteristics of text characters for text/graphics separation. Directional mathematical morphology approach has been used by Luo et al. [LAD95] for separation of character strings from maps. Tan et al. [TN98] illustrates a system using *Pyramid* structure.

For the recognition purpose of multi-oriented characters from engineering drawings, Adam et al. [AOC$^+$00] used Fourier Mellin Transform. Parametric eigen-space method is used by Hase et al. [HSYS03]. Xie and Kobayashi [XK91] proposed a system for multi-oriented English numeral recognition based on angular patterns.

Goto and Aso [GA99] proposed a local linearity based method to detect text lines in English and Chinese documents. Hones and Litcher [HL94] finds line anchors in the document image and then text lines are generated by expanding the line anchors. Loo and Tan [LC02] proposed a method using irregular pyramid for text line segmentation

in such documents.

But, there are not many published work with a complete system which can deal with word searching in graphical documents. Most of these existing work [DMS95b, AOC+00] proposed in the literature considered a fixed set of document with a single font of characters for their experiment. Detection and retrieval of curvi-linear word in such complex documents are not discussed. One of the main reasons could be the complexity, which grows incrementally from character segmentation to text word retrieval.

6.1.3 Outline of the Approach

The main contribution of this chapter is to present a word indexing architecture in graphical documents to facilitate searching of query word which can take care overlapping/touching cases. We use the spatial information of labeled connected components as the high level descriptors. A generic architecture is proposed to index such spatial information. Given a query text, the system extracts positional knowledge from the query and use the same to search the possible zones to generate hypothesis. These hypotheses are further reduced to find the probable locations of a word. Spatial arrangement of character components drives our system to locate text word in such non-structured environment. The system works on multi-scale and multi-rotation environment.

We label the connected components of a document as alpha-numeric text characters. For each component (text character) we find its n-nearest neighbors. The locations of all character pairs are stored in a character pair table. Given a query word, the code information is used to locate the hypothesis. The query is split into sequence of pairs of characters. From the character pair table, we find the locations of all character pairs in the documents. Next, a dynamic programming algorithm is applied to find possible combination of words by joining the character pairs. String edit distance is used to match the query word as the objective function.

The rest of the chapter is organized as follows. In Section 6.2, we describe the indexing of spatial representation of character components. The word retrieval process is detailed in Section 6.3. The experimental results are presented in Section 6.4. Finally, conclusion is given in Section 6.5.

6.2 Indexing of Character Components

The architecture of our word retrieval system in graphical documents is based on CC labelled as text characters. Again, text characters are too local and very generic to be used for matching primitives. Due to natural spatial arrangement of text characters among them, we characterize the connected component information in the document based on their spatial arrangement among each other. These spatial information are indexed efficiently to access them later. In Chapter 2, we have detailed about isolated character recognition. Labelling of individual characters in multi-scale and multi-orientation environment is used for word retrieval process in graphical documents. The architecture of component indexing is described as follows.

6.2.1 Component Neighborhood and Pairing

To represent the spatial arrangement, the components are grouped in a list of component pair. A component forms character pair with each of the components selected from its neighbor. These component pairs are selected based on their proximity and size similarity. Each component is associated to its n-nearest components which are nearest from its boundary. The neighbor components are decided using boundary growing algorithm from the boundary of the corresponding component (See Chapter 5 (Section 5.3.1)).

For the size similarity, at first the size (l_{Ci}) of individual components (C_i) is computed. The size (l_{Ci}) is computed from the radius of the MEC of the component. The size similarity between two components is tested based on the components' sizes [CK09]. A component C_1 will be similar in size to its neighbor component C_2 if Eq.6.1 is satisfied.

$$l_{C2}/1.5 \leq l_{C1} \leq 1.5 \times l_{C2} \tag{6.1}$$

These components pairs are stored after removing duplicity. To do so, a graph data-structure is used. Formally, a proximity graph $G_p = (V, E)$ is created for these component pairs (here V = vertex and E = edges). An edge, $e_{ij} \in E$ is formed by component pair (C_i, C_j) if, components C_i and C_j are neighbor. See Fig.6.2(a), where the n-neighbor ($n = 4$) components of 'A' are shown by joining it with its neighbors 'L','N','T' and 'T' (according to distance).

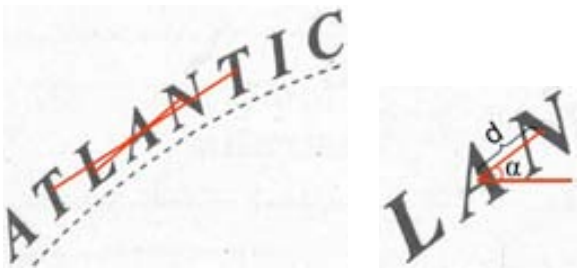
6.2.2 Indexing by Character Pair

The spatial feature of each character pair is encoded and recorded in a table called Character Pair table or CP-table. Encoded spatial feature is indexed using the component pair's recognition labels (ASCII/unicode value) which is performed by character classifier process. Let, L_i denotes the ASCII/unicode label of character component C_i. The character labels of each component pair are used to obtain the index key (Φ^W). A 2-dimensional Look-Up-Table (LUT) may be used to find Φ^W from each pair of character components' ASCII/unicode values (L_i, L_j) as shown by Eq.6.2.

$$\Phi^W_{ij} = (L_i, L_j) \tag{6.2}$$

The size of LUT depends on the total number of recognition labels used in the experiment. CP-table contains the spatial information of each character pair. The spatial information is encoded by distance, angle and location of the component pairs. We compute the CG (center of gravity) of each component for this purpose. These attributes are described as follows.

1. Distance: Let, V^C_i be the center of MEC of the component C_i. The distance (see, 'd' in Fig.6.2(b)) of component pair C_i and C_j is defined by Euclidean distance d_{ij} between their CGs.
2. Angle: The angle (α_{ij}) of line joining V^C_i and V^C_j with respect to x-axis is considered to take care of direction. See, 'α' in Fig.6.2(b)).
3. Location: Let, $P_i = \{x_i, y_i\}$ denotes the location of V^C_i. The locations of both components C_i and C_j of the pair are stored.



(a) (b)

Figure 6.2: (a) Character and its n-nearest (n = 4) neighbor are shown for the character 'A'. (b) A character pair and its spatial features like distance (d) and angle (α) are shown.

Thus, for each component pair C_i and C_j, the feature information f_{ij} is encoded as given below.

$$f_{ij} = (d_{ij}, \alpha_{ij}, P_i, P_j)$$

This encoded spatial feature f_{ij} is stored in CP-table using the index key Φ_{ij}^W. In Fig.6.3, we show the representation of the CP-table and the indexing architecture.

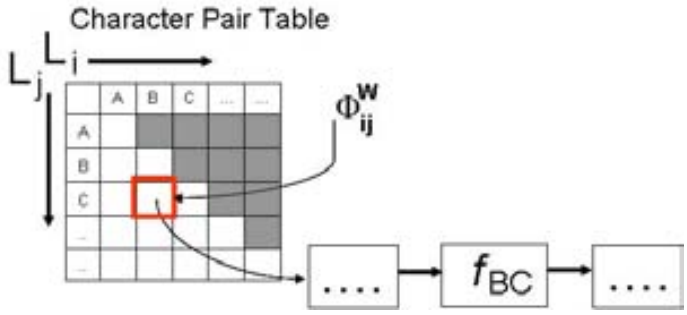


Figure 6.3: Indexing scheme using Character Pair table is shown. Spatial features of each character pair is attached using chaining.

It is to be noticed that, in CP-table, for each index key Φ_{ij}^W there may be multiple entries of spatial feature for presence of many similar character pairs (having similar codes L_i, L_j of components) in documents. The collision in CP-table is resolved by chaining.

6.3 Retrieval of Query Word

To retrieve a query word given by the user, we take into account the knowledge of the presence of character codes and their relative position information from the query word itself. The character codes are used in our system to search the possible location of them in the document. Finally a dynamic programming algorithm is used to locate different instances of the query word using their relative position information. This

process is described in two steps mainly knowledge extraction from query word and dynamic programming algorithm.

6.3.1 Knowledge from Query Word

From the query word, the system extracts the knowledge about character information. As discussed before, the character components in graphical documents are in multi-orientation environment. To search a query word in such documents, the ASCII/unicode values of characters of query word are converted to the label of multi-oriented characters used in our system. So, the ASCII/unicode character space is reduced to coded multi-oriented character space by $U^A \Rightarrow U^M$, where, U^M represents equivalence classes of rotationally similar characters. $|U^A|$ $(= 62)$ is the cardinality of alpha-numeric text characters and $|U^M|$ $(= 40)$ is the total number of unique multi-oriented character labels. The conversion is done by a look up table as in Eq.6.3, where, Ω represents the mapping function.

$$U_i^M = \Omega(U_i^A) \tag{6.3}$$

Thus, given a query word, for e.g. "Danube", the query will be changed to "Dauube" in multi-oriented character domain, since the characters 'u' and 'n' are grouped together.

Let, the query word Q be of character sequence $q_1 q_2 \ldots q_i \ldots q_l$, where l is the length of the string. By Eq.6.3, each character q_i is converted to multi-oriented character code L_i. Thus, Q is translated into a multi-oriented character sequence Q^M. From this character sequence Q^M, a list of character pairs (Q^P) is formed using different relative positions of the characters in the sequence. To avoid missing characters due to touching/overlapping or noise, different combination of character pairs are considered in the query.

Finally, the list Q^P is built with character pairs $\{q_i, q_{i+r}\}$ with different choices of relative position (r), where $i + r \leq l$. For the query "Dauube", the list Q^P will be formed with 'D-a', 'a-u', 'u-u' etc. for $r = 1$ and 'D-u', 'a-u' etc. for $r = 2$. These are shown in Fig.6.4.

Next, each character pair of list Q^P is used to generate the index key for CP-table and to retrieve the corresponding components' spatial information. The retrieved components will be joined sequentially to form a target string T^M. This sequential joining is performed using dynamic programming approach explained as follows.

6.3.2 Character Pairs Joining by Dynamic Programming

The core of a dynamic programming (DP) algorithm is the module that takes a string (query word) to incorporate contextual information and a list of the symbols (characters) composed of character pairs from the document image and returns a value that indicates the confidence that the word image (sequence of characters) represents the string. Given a query word, the primitive character pairs are joined and matched against the lexicon word so that the average cost is minimized using dynamic programming algorithm. A string edit distance algorithm "Levenshtein Distance"

	D	a	u	u	b	e
D						
a	D-a					
u	D-u	a-u				
u	..	a-u	u-u			
b	..	..	u-b	u-b		
e	..	..	..	u-e	b-e	

Figure 6.4: Character pairs from query word are used to create Q^P list.

has been used for such dynamic programming approach. This algorithm is described below.

Levenshtein Distance

Given a sequence of observations derived from an input word image and a database of similar descriptions of all possible words, one usually attempts to find a stored description that has the minimum edit-distance with respect to the input. Usually, Levenshtein's metric concept [Nav01] is used. Levenshtein distance (LD) is a measure of the similarity between two strings, which we will refer to as the source string (s) and the target string (t). The distance is the number of deletions, insertions, or substitutions required to transform s into t. The minimum edit-distance between two chains of symbols denoted by o1 $s_1 s_2 \ldots s_m$ and $t_1 t_2 \ldots t_n$ is the cheapest cost one needs to pay in order to transform one chain to the other using the operations of symbol deletion, substitution, or insertion of an extra one. This problem has a well-known dynamic-programming solution:

$$\begin{aligned} & d(s_1 s_2 \ldots s_m, t_1 t_2 \ldots t_n) = \\ & \texttt{min}\{d(s_1 \ldots s_{m-1}, t_1 \ldots t_{n-1}) + \texttt{sub}(s_m, t_n), \\ & d(s_1 \ldots s_{m-1}, t_1 \ldots t_n) + \texttt{ins}(s_m), \\ & d(s_1 \ldots s_m, t_1 \ldots t_{n-1}) + \texttt{del}(t_n)\} \end{aligned} \tag{6.4}$$

where $\texttt{del}(t_n)$, $\texttt{sub}(s_m, t_n)$, $\texttt{ins}(s_m)$ are the cost parameters for the three above-mentioned operations respectively. A cost parameter may be constant or a function of the symbol.

Joining of Character Pairs

The sequence of character pairs in the query word is used to spot the similar words in the document. Given, the character sequence of query word $\{q_1 q_2 \dots q_i \dots q_l\}$, the objective is to match it with a similar sequence of character components from the document. This matching can be written by following the Eq.6.5, where $\{C_{qi}\}$ represents the corresponding set of component character q_i. Ψ is the function that gives all the possibilities of query word matching. The solution is drawn from the sequence of character pairs which constitute the query word.

$$\Psi(q_1 q_2 \dots q_l) = (\{C_{q1}\}, \{C_{q2}\} \dots \{C_{ql}\}) \tag{6.5}$$

Using the knowledge of query character pair list Q^P and component pair list CP-table, the above Eq.6.5 can be modified according to successive sequence of pairs.

$$\Psi((q_1 q_2), (q_2 q_3) \dots (q_{i-1} q_i) \dots (q_{l-1} q_l)) \Rightarrow$$

$$(C_{q1} C_{q2}), (C_{q2} C_{q3}) \dots (C_{qi-1} C_{qi}) \dots (C_{ql-1} C_{ql})$$

Given a query character pair $(q_{i-1} q_i)$, the CP-table is searched for possible locations of the corresponding components. Two consecutive component pairs will be selected for joining if both of them have a common component and their spatial relationship are compatible. For example, the component pairs (C_i, C_k) and (C_k, C_j) are linked together if the spatial relation between them f_{ik} and f_{kj} are similar. This similarity is compared by their size, linearity and distance (Section 6.2.2). Size of components l_{Ci}, l_{Ck} and l_{Cj} are measured for similarity as given in Eq.6.1. Linearity of component pairs is checked by the angles of the components in the string. This is performed by checking local linearity and it is formulated by Eq.6.6. The distance equation (Eq.6.7) between two component pairs assures the inter-character spacing is same in the word. Here, T_α^W and $T_d W$ are fixed according to the experiment result.

$$\|\alpha_{ik} - \alpha_{kj}\| < T_\alpha W \tag{6.6}$$

$$\|d_{ik} - d_{kj}\| < T_d W \tag{6.7}$$

After linking the components through this constrained merging, the character clusters are passed to a dynamic programming approach to find the query word. The Levenshtein distance between two strings "Query word" and "Sequence of component pairs" is used to measure similarity. For our application, the value of cost functions deletion, substitution and insertion are considered same and the value is set to 1. Here, we did not need more sophisticated cost function, as it served our purpose.

Finally, words in the document are ranked on the basis of the similarity with query word by comparing the expanded string through component pairs. The rejection of a character pair string is performed if the distance between sequences is greater than a threshold $T_L (= l/3)$. The choice of the threshold value is set up through experimental results.

In Fig.6.5, we sketch the complete block diagram of the query retrieval process in graphical documents. As explained, the labelled text characters are indexed first according to the pair formation with its neighbor. These are stored in CP-table.

Next, the character sequence from the query word is searched in the CP-table. The best sequence is obtained using a matching (Levenshtein Distance) with the query word.

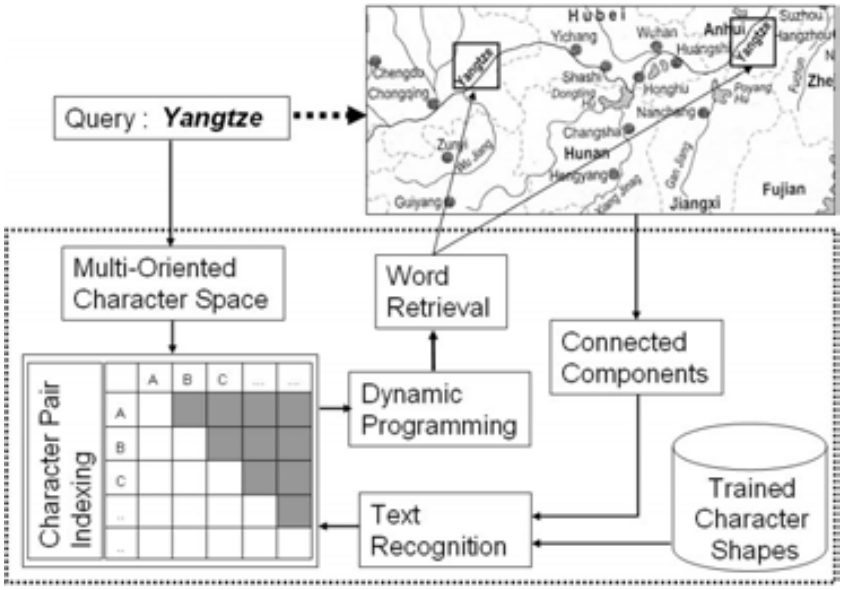


Figure 6.5: Different steps of the word retrieval system.

6.4 Experimental Results

We have evaluated word indexing methods in the database of graphical documents which are in multi-orientated and multi-size environment. To evaluate the word retrieval performance, the real data is collected from maps of 2 different resources namely, a set of maps from geography books and a collection of historical archives. The map images are from geography books. Some examples of this dataset are shown in Fig.6.1 and Fig.6.7. The average size of these images is 1200 $\times$ 1200. There are approximately 35-40 words in each document. The other set of real maps is selected from a historical archive of old maps. See Fig.5.14 of Chapter 5, where a portion of the document is shown. These images contain text printed in dark color compared to the background. The foreground text components are obtained in an automatic way as discussed in Chapter 5. We also use the synthetic dataset of maps (See Appendix A.5.1). We show a part of a synthetic map in Fig.6.6. Table 6.1 shows the experimental outline. We have used precision-recall to measure the word retrieval result.

Table 6.1: Experiment outline.

Dataset	map documents (real + synthetic)
Objective	To test the robustness of document retrieval using query word
Metric	Precision-Recall

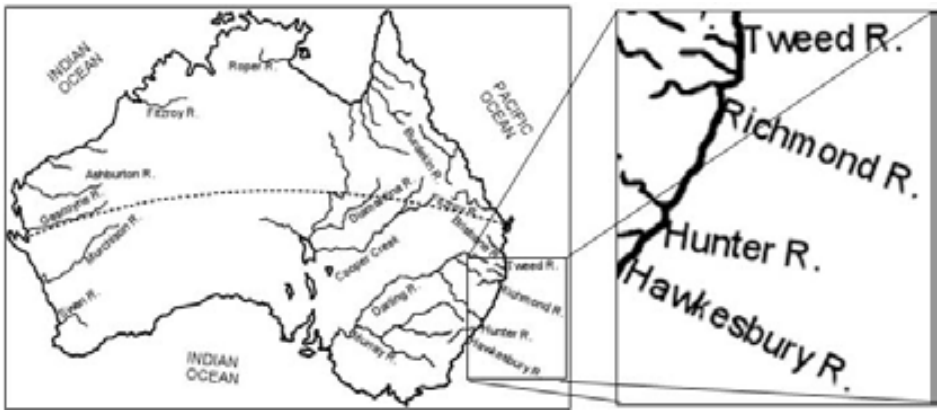


Figure 6.6: Parts of map from a synthetic documents showing orientation and touching of characters.

6.4.1 Performance of Word Detection

In our experiment on indexing of words in graphical documents, we tested our scheme on 60 query words (40 real and 20 synthetic). There are total 10 real and 10 synthetic documents. We have found that the total number of instances of such query words in the documents are 42 (real maps) and 110 (synthetic maps). The characters present in documents are of different size and orientation. Their fonts are also different in these documents. We used angle based features and SVM to classify such text characters.

We show in Fig.6.7 the word detection result of our approach in a real map image. The location of a query word is detected in this document by our character-pair indexing approach. The detection of a word is performed without any prior knowledge of the word orientation within document. To visualize, we draw a minimum enclosing box of the component clusters in that particular location as shown in Fig.6.7. It is appreciable to see that our approach performs well when there exists text overlapping with long graphical objects. In Fig.6.7, the word "OCEAN" at the top position contains a touching character 'E' with graphical lines. We have detected them correctly. At the bottom of the document, it is appreciable that the query word is detected in spite of orientation of that word.

Given a query word and a map document shown in Fig.6.8(a), the retrieved cropped images are ranked in Fig.6.8(b). Here, we have tested with the query word "SEA". The ranking is done based on string edit distance. In this image, there are total 4 different instances of "SEA". We have extracted 3 of them with distance 0 and missed one which is below the word "CASPIAN". The word is not extracted

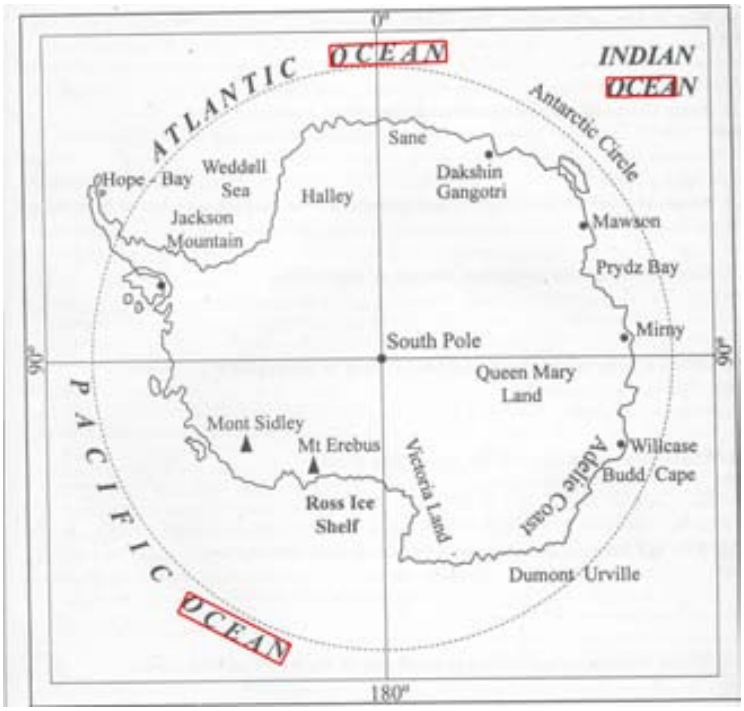


Figure 6.7: A map is searched with query word "OCEAN" and such results are marked by thick rectangular boxes.

due to insufficient extracted characters. Character 'S' is not segmented properly due to its touching with long line. And character 'A' is wrongly recognized as 'V'. We also received one false positive, which is ranked fourth (marked as (4) in Fig.6.8(b)) because of similarity (distance 1). It is appreciable from the ranking of these images, how overlapping/occlusion does affect the ranking process.

In Fig.6.9, we show some results of word retrieval using a few query words. The results show the advantage and robustness of our approach with different kinds of problem in graphical documents. Fig.6.9(a) shows the retrieval of word when inter-character spacing is more than normal spacing. Fig.6.9(c) shows a word where touching characters are present. Fig.6.9(d) and (e) demonstrate how the method is tolerant to the graphical line touching. In Fig.6.10, we show the retrieval results in historical images.

The detection accuracy is affected mainly due to broken text characters in the document and the presence of long graphical lines over text regions. Due to noise when characters are broken and the broken components of a character could not be joined through preprocessing, the character labeling process might be wrong, and hence the word retrieval process is sometimes erroneous. In Fig.6.11 we show some images which we could not extract using our system, because of improper character segmentation (See Fig.6.11(a) and (c)) and presence of many touching characters (See

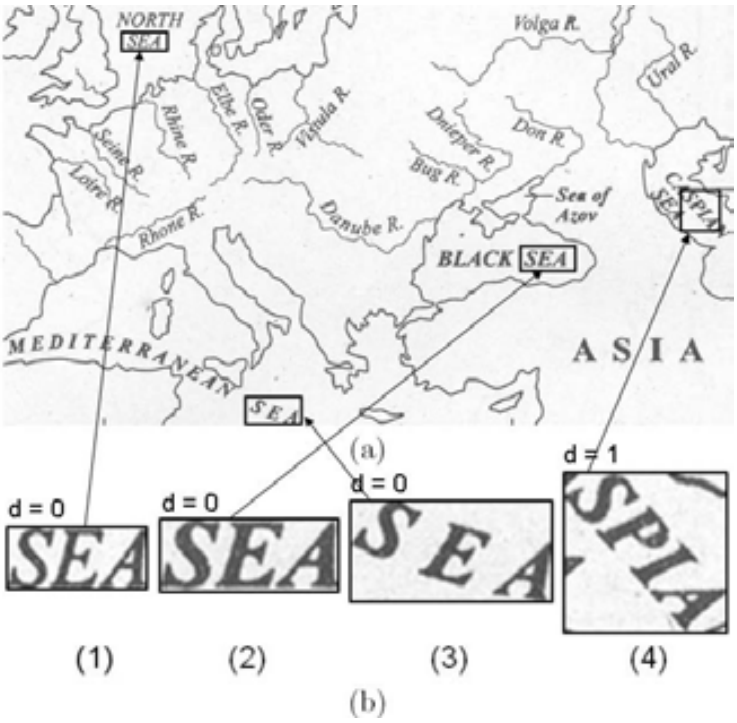


Figure 6.8: (a) Example of map where the query word "SEA" is searched (b) Ranking of search result based on the query word "SEA".

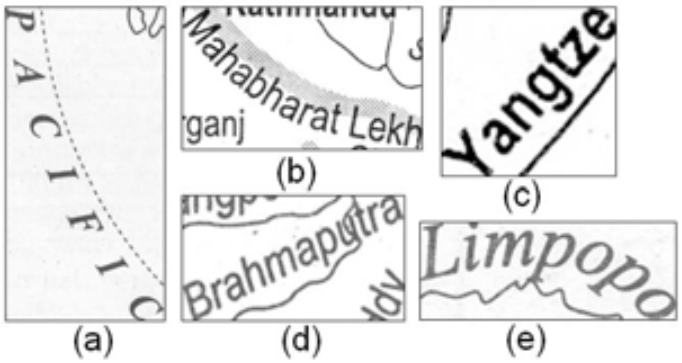


Figure 6.9: Some results of our method based on query word (a) PACIFIC, (b) Mahabharat Lekh (c) Yangtze (d) Brahmaputra (e) Limpopo.

Fig.6.11(b)). In Fig.6.11(c), the characters 'R' were not segmented properly due to noise. In Fig.6.12, we show some erroneous results using this system. In Fig.6.12(a), due to similar character sequence we have obtained a part of word "SOUTHERN" using the query string "NORTHERN". Fig.6.12(b) shows the retrieval result with query

Figure 6.10: Some results of our method in color historical images based on query word (a) MANRESA, (b) FIGUERAS (c) ARAGONIA.

word "Narmada" because of rotation invariance nature of the character recognition labels. Here, component 'h' is mislabeled as 'r'. Thus, using rotation invariant classes, the component sequence is recognized as "armad" which is similar to "Narmada".

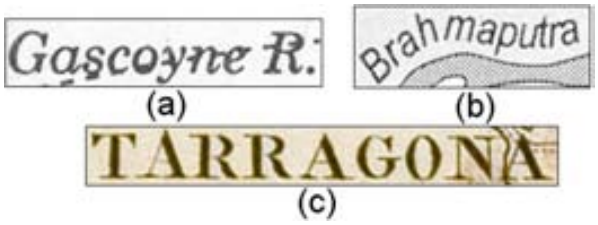


Figure 6.11: Some words in the documents which could not be retrieved using our method.

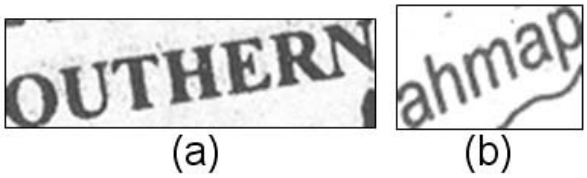


Figure 6.12: Some erroneous results of our method based on query word (a) NORTHERN, (b) Narmada.

To evaluate the performance of the system with a query word against the retrieved word images, we use common ratio of precision (P) and recall (R). The computation of precision and recall is discussed in Chapter 4. The retrieved word images, the system results are ranked by the cost obtained in dynamic programming algorithm. The result of each word image is considered as relevant or not depending on the ground-truth of the data.

In Fig.6.13, we show the performance of word retrieval system in different datasets. The precision and recall table is computed for real and synthetic datasets according to different choice of string edit distance. In synthetic maps the precision is obtained upto 89% when the recall is 81%. Then the performance is degraded due to more relaxing

of string distance comparison. Word retrieval accuracy in real image database is poor compared to that of synthetic image database. Lack of proper character segmentation from color map images is one of the reasons of low accuracy. In Table 6.2, we show the overall performance of the system with the best threshold distance $T_L(= l/3)$ used in the experiment.

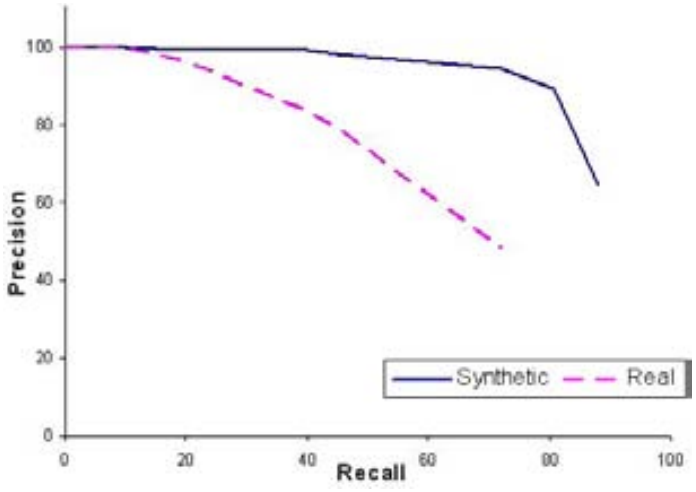


Figure 6.13: Precion-recall curve of word retrieval.

Table 6.2: Word spotting results

Type	Query Words	Total Instances	Detected Words	Correct Words
Synthetic	20	110	151	97
Real	40	42	62	30

6.5 Conclusion

We have presented a word searching algorithm based on spatial arrangement of character components in graphical documents. Labeled multi-oriented text characters are used to generate high level local feature to take care of word detection in complex graphical documents. The text label of these connected components and their pair information are used to index local spatial information of them. Relative positional information of text characters in a string is used for this purpose and hypothesis were generated based on that.

We tested our method in a database of graphical documents containing characters in touching/overlapping environment. In retrieval of words from document images, all components of a document pass through a recognition process which is time consuming. In future, we plan to work on this performance issue. One advantage of our

approach is that, the system can detect the query word even if we do not extract all the text characters of a query string. There is not much effort in this area, so we hope it will be helpful for researchers in their future work.

In the next chapter, we will present an approach for the document retrieval based on the graphical symbol e.g. seal. The spatial relationship among the characters will be used for seal recognition and detection in complex documents.

Chapter 7

A GHT based Seal Detection Approach using Local Text Characters for Retrieval in Digital Libraries

This chapter deals with a particular application involving multi-oriented text character detection. It is focused on automatic detection of seal (stamp) from documents with cluttered background. Seal detection involves a difficult challenge due to its multi-oriented nature, arbitrary shape, overlapping of its part with signature, noise, etc. Here, a seal object is characterized by scale and rotation invariant spatial feature descriptors computed from recognition result of individual connected components (characters). The concept of Generalized Hough Transform (GHT) is used to detect the seal and a voting scheme is designed for finding possible location of the seal in a document based on the spatial feature descriptor of component pairs. The peak of votes in GHT accumulator validates the hypothesis to locate the seal in a document. Experimentation is performed in an archive of historical documents of handwritten/printed English text. Experimental results show that the method is robust in locating seal instances of arbitrary shape and orientation in documents, and also efficient in indexing a collection of documents for retrieval purpose.

7.1 Introduction

As stated in Chapter 1, one of the objectives of the thesis is to retrieve the graphical documents based on user's query. In Chapter 6, we presented a technique towards the detection of query words from graphical documents. In this chapter we will discuss a novel strategy of document image retrieval using graphical symbols containing text characters. "Seal" is one of such graphical entities that contains text characters along with a logo and a boundary surrounding it. Automatic detection of seal (stamp) in documents allows us to index an archive of documents efficiently.

Libraries and archives are generally interested in mass-digitization and transcription of their book collections. The objective is not only to preserve in a digital format of documents of high value, but also to provide access and retrieval services to wider users, and assist scholarly research activities. On the other hand, companies are interested in implementing digital mail-rooms to improve the efficiency of paper-intensive workflows and to reduce the burden of information processing of incoming mails, faxes, forms, invoices, reports, employee records, etc. By this digital mail-room, companies try an automatic distribution of incoming mails to their respective departments based on the content of electronic documents.

Content-based Document Image Retrieval (CBDIR) involves a search process where the user's request is a concept to be found. Traditionally, document retrieval is associated to textual queries. The records in typewritten and structured documents are indexed with the traditional high performing commercial OCR products. Once the image document is OCRed, the resulting ASCII information is compared with the query with a string search algorithm. The OCR techniques may fail in documents with high degree of degradation such as faxes due to compression or bilevel conversion, historical document databases which are often degraded due to aging or poor typing, or handwritten documents. For such type of documents, word-spotting [LT08, RM05] provides an alternative approach for index generation. Graphical symbols containing text characters can also be used for index generation of such documents. In this chapter text location in the particular layout of seals is used for retrieval purposes.

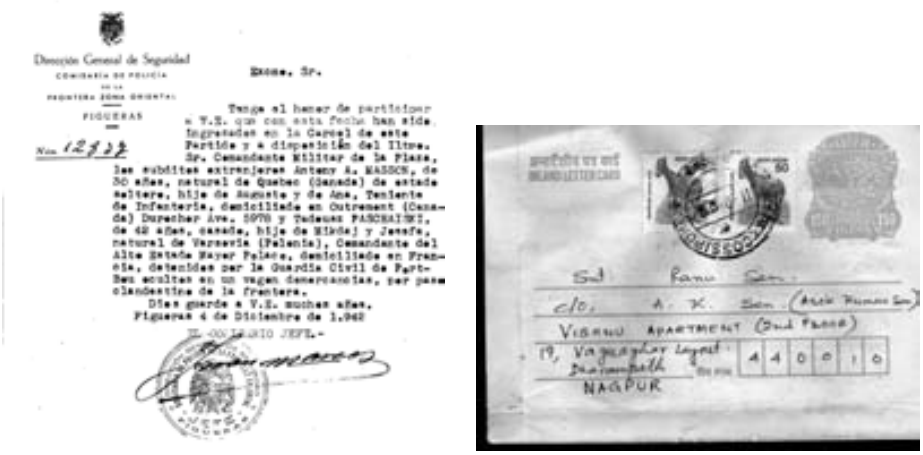
7.1.1 Motivation

Besides huge amount of text strings, some documents contain a large amount of information that is usually neglected and can provide richer knowledge e.g., administrative documents contain graphical symbols such as logos, stamps or seals. Searching in terms of symbol queries such as logos or seals allows the quick filtering or routing of the document to the right business process, archive or individual. Thus the detection of these graphical symbols in documents undoubtedly increases the performance of document retrieval [UMM98, ZD09]. Given a single instance of a symbol queried by the user, the system has to return a ranked list of segmented locations where the queried symbol is probable to be found. Traditional word-spotting methods which are devised to extract row/column-wise features from segmented image may not be useful for such purpose. Following the idea of word spotting, the problem of locating a graphical symbol in a document image is called *symbol spotting*. If we extend the idea of symbol spotting to a database of document images, i.e, a digital library, the problem is called *symbol focused retrieval* [RL08]. The arrangement of featured keypoints are used to retrieve document images [NKI05]. These methods are promising in detecting objects in general in terms of accuracy, time and scalability. Graphical objects such as symbols, seals, logos, etc. are synthetic entities consisting of uniform regions and these are highly structured [RL08, TL03]. These facts make geometric relationships between primitives as discriminative cues to spot symbols.

Indexing of documents can be performed based on graphical entities. A scheme on document indexing based on seal information is proposed in this chapter. Seals are complex entities consisting of mixed textual and graphical components. Information

obtained from seals could be used for efficient storage and retrieval of the documents. Seals generally have closed, connective contour surrounding text characters, and logos which indicate owner and usage of the seal [LLWZ07]. They bear some constant character strings to convey the information about the owner-organization and its locality. Besides, in many instances a seal contains some variable fields such as date [NPR07]. Date may provide the sending or receiving information of the document. Undoubtedly, automatic seal detection and recognition is an important stage to follow this task. It also allows to identify the document sources reliably.

Detection and recognition of seals is a challenging task, because seals are generally unstable and sometimes contain unpredictable patterns due to imperfect ink condition, uneven surface contact, noise, etc. [ZJD06]. Overlapping of seals with text/signatures, missing part of a seal, etc. are typical problems and add difficulty in seal detection. A seal instance may be affixed on any position within the document, that requires detection to be carried out on the entire document. Also a seal may be placed in arbitrary orientation. See Fig.7.1(a), where a historical archive document containing a seal is shown. The seal is overlapped here with part of signature and text regions. As a result some information of the seal is missing/illegible. We also show a postal document in Fig.7.1(b) which contains a seal overlapped with stamp images. It is to be noticed that, due to overlapping and noise, many text characters of the seal are missing.



(a) (b)

Figure 7.1: (a) A historical document with a seal. The seal is overlapped with the signature here. (b) A post-card image showing a seal with cluttered background.

7.1.2 Related Work

Detection of seals in documents can be treated as localizing objects in different pose. In many instances seals and text are in different colors, and to handle such document some researchers have studied the detection process based on color analysis [Fri03,

UM05]. In [Fri05] Frisch used fusion operator denoted as fuzzy integral for color seal extraction. Two different fuzzy measures were used in the computation of the fuzzy integrals on color channels of RGB color model. Ueda and Matsuo [UM05] assumed the pixels of the color cheque image constructed four clusters: seal imprint, characters, background and the superimposed area of characters and seal imprint. HSV color model has been used to perform the clustering.

In mass digitization process due to compression or bi-level conversion, color analysis can only solve a set of problems. A prior knowledge of the outer boundary shape of a seal (e.g. circular, rectangular, elliptical, etc.) is helpful to locate the seal in documents [ZJD06]. Accurate contour detection or shape estimation is needed for the extraction of such boundary. Moreover, a further seal recognition technique is needed here, because it is difficult to recognize the seal if text information within the seal is different although the boundary shape is similar.

Thus, many times a seal has been treated as a symbol and methodologies like segmentation by Delaunay tessellation [CW98] are applied for seal recognition. Hu et al. [HYZ+95] proposed a heuristic method to find the best congruent transformation between the model and sample seal imprint. A correlation based algorithm for seal imprint verification is also presented in [Hor01, HHYY96]. Chen [Che96] used correlation based block matching in polar coordinate system based on rotation-invariance features. Since, under the constraint of fixed scanning resolution, multi-scale invariance property is not necessary for seal detection in documents. He also proposed a method using contour analysis to find the principal orientation of a seal[Che03].

Matsuura et al. [MM02] verified seal image by using the discrete K-L expansion of the Discrete Cosine Transform (DCT). Next, a decision function was included to verify the seal model. Rotation invariant features by the coefficients of 2D Fourier series expansion of the log-polar image is presented in [MY04]. Lee and Kim [LK89] used the attributed stroke graph, obtained from the skeleton, for registration and classification of seals. Gao et al. [GDC95] used a verification method based on stroke edge matching.

In Table 7.1, we summarize the state-of-the-art approaches for seal detection and recognition methods. A seal involves rich textual structure with a bigger boundary frame and smaller text components. The seal recognition approaches explained above are mainly pixel-based shape descriptor which consider the seal as a whole object. In Fig.7.2(a) and Fig.7.2(b), we show two different seal images containing similar circular boundary frames but different text information. Due to their complex structure, methods using only boundary detection or orientation based matching etc. may not be useful to discriminate them properly. Also, when a seal contains variable field such as date, hard-matching approach [Che96] may not work. Thus we need a higher level of features (e.g. text character information) to provide more detailed description of seals.

7.1.3 Outline of the Approach

In this chapter, we propose a scheme on document indexing based on seal detection and text information has been used for this purpose. When imprinted, seal produces a fixed set of text characters with a logo (sometimes) to describe its identity. Al-

Table 7.1: Related work on seal detection and recognition.

Method	Main Feature of Algorithm	
	Detection	Recognition
Frisch [Fri05]	Fuzzy integral in RGB color model	-
Ueda et al. [UM05]	Pixel clustering in HSV color model	-
Zhu et al. [ZJD06]	Boundary contour detection	-
Chinag et al. [CW98]	-	Delaunay tessellation
Matsuura et al. [MM02]	-	K-L expansion of the DCT
Lee and Kim [LK89]	-	Attributed stroke graph

Figure 7.2: Examples of two seal images containing similar circular frame but different text information.

though the boundary frames are similar in some seals, they may contain different text information. We classify them as different types of seals. So, it is important to take care of text information knowledge in seal recognition purpose. Text characters in seal are of multi-rotations and multi-scale. The font styles vary according to the design by its owner/organization. Generally seals are affixed in documents that also contain a huge amount of text characters. Thus, text information within seal may be degraded due to the overlapping with text present in that document. In seal, the relative position between text characters is structured by its property. This is a key observation in our formulation. Features computed from spatial information of text characters within seal will allow us to detect the seal efficiently.

Generalized Hough Transform (GHT) is an extension of the "standard" Hough transform, used to detect shapes which cannot be described by an analytical equation, but can be characterized with discrete formulation. It roughly consists in explicitly listing all the points of the shape. Next, the position information (feature) of each point to a reference point is stored in a table. In the image, the feature of each point votes in the reference point stored in the table for those pixels having similar local information (usually the gradients). By detecting the local parts to vote for possible locations of a shape object, the peaks in voting space can be used for detection.

Our proposal of seal detection approach is based on the concept of GHT. Instead of feature points, the text characters in the seal are used as basic features for seal detection. We label the local connected components of a seal as alpha-numeric text characters. To obtain this, we employ multi-scale and multi-oriented text character recognition using a SVM classifier. The recognized text characters are used to provide high level descriptor in our approach. For each component (text character) we find its n-nearest neighbors. Next for each pair of components, we encode their relative spatial information using their distance and angular position. Given a query seal we compute the relative spatial organization for pair-wise text components within seal. These information are stored into a spatial feature bank. In an image, for each component pair, we use this spatial feature bank to retrieve query seal hypothesis. A vote is casted for possible location of the seal based on the matching of component pairs to ensure detection of partial seal objects. The locus of high number of votes (the peak) is the result of the seal spotting. Votes in GHT accumulator validates the hypothesis to locate the seal.

The main contribution of this chapter is to use of recognized local text components as high level descriptors and to generate hypothesis of the seal location based on spatial arrangement of these descriptors. Our approach is based on local text characters instead of only pixel-level descriptors to overcome distortion and affine-transform problems which are main drawbacks in existing approaches. Our approach does not need a previous segmentation, thus provides direct focus on seal detection. The approach is robust to detect seal in noisy, cluttered document. We can therefore conclude that combining individual character recognition with a SVM and relational indexing using GHT method, our work can be classified as a combination of statistical and structural approach.

The rest of the chapter is organized as follows. In Section 7.2, we explain the Generalized Hough Transform technique for general shape detection in brief. In Section 7.3, we describe the representation of the seal and its detection process. The experimental results are presented in Section 7.4. Finally conclusion is given in Section 7.5.

7.2 Generalized Hough Transform

Since, the approach proposed in the chapter is based on GHT, we first outline in this section the fundamentals of this technique.

Generalized Hough Transform (GHT) [Bal81] is the extension of the idea of the "standard" Hough Transform (HT), which is a feature extraction technique originally devised to detect analytical curves (e.g. line, circle, etc). The purpose of the technique is to find imperfect instances of objects within a certain class of shapes by a voting procedure. The modification of GHT enables this technique to be used to detect arbitrary objects (non-analytical curves) described with its model.

A look-up-table called R-table is used in GHT to store information concerning the template of the object to be detected. Given an object, the centroid is chosen usually as reference point (x_c, y_c) [BR92]. From this reference point, data from each point on the object's edge is recorded in R-table. Let, the line joining from a boundary

point (x, y) to the reference point makes an angle (α) with the x-axis. Also, 'r' be the distance from that boundary point to the reference point as shown in Fig.7.3. R-table is constructed to store (α, r) of each boundary point by indexing with it's gradient angle (ϕ) as shown in Table 7.2. From the pair (α, r), the reference point can be reconstructed.

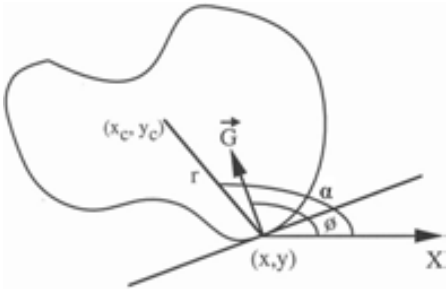


Figure 7.3: The geometry used to construct the R-table in GHT.

Table 7.2: The R-Table constructed for Fig.7.3

Gradient Angle	(α, r) pairs
ϕ_1	$(\alpha^1, r^1)_1, (\alpha^2, r^2)_1, \ldots (\alpha^n, r^n)_1$
ϕ_2	$(\alpha^1, r^1)_2, (\alpha^2, r^2)_2, \ldots (\alpha^n, r^n)_2$
.	.
.	.
ϕ_m	$(\alpha^1, r^1)_m, (\alpha^2, r^2)_m, \ldots (\alpha^n, r^n)_m$

Given a test image, an accumulator (A) is defined in the parameter space. The accumulator records the votes of the edge points to determine the most probable center of the prototype object. More specifically, the gradient angle ϕ_i which is obtained from each edge point (u_i, v_i) of the test image is utilized to retrieve corresponding entries of the R-table. The possible location of reference point in the parameter space is calculated by the following equation.

$$(x_{ci}, y_{ci}) = u_i + r(\phi_i)cos[\alpha(\phi_i)], v_i + r(\phi_i)sin[\alpha(\phi_i)] \tag{7.1}$$

In this equation, (x_{ci}, y_{ci}) denotes the location of the possible reference point. $r(\phi_i)$ and $\alpha(\phi_i)$ signify the magnitude and the angle of the position vector obtained from the R-table for index ϕ_i, respectively.

7.3 Seal Detection using GHT

Seals in documents are affected mainly due to noise, rotation, occlusion and overlapping. To take care of these problems, our approach is based on partial matching which is inspired by the Generalized Hough Transform (GHT) [Bal81]. Hough Transform (HT) is originally devised to detect analytical curves. GHT extends the idea of HT

for non-analytical curves. The edge information of a shape is used here to define a mapping from the orientation of an edge point to a hypothetical reference point of the shape and detect the shape based on the accumulation of these reference points. The purpose is to find similar instances of objects by a voting procedure. The voting procedure is carried out in a parametric space from which object candidates are localized as local maxima in the voting space. For indexing the relative position of edge points, the GHT uses a look-up table called R-table.

A classical pixel-based GHT may not be efficient for seal detection because two different seals having similar frame boundary (such as circular, rectangular, etc.) can't be distinguished properly. Also, the pixels of text characters are very local. Seals generally comprised of many text characters. Thus, missing of some text characters or inclusion of background text in a seal may not be tackled properly using such method. In our approach, text characters in a seal are considered as high level descriptors. Since, the spatial information between text characters will remain fixed, the relative position of them is used to vote for seal detection. In Chapter 2, we explained recognition technique for isolated characters. We use this approach for labelling the text characters here.

Here, we describe the architecture of our method with four key-parts namely: spatial information extraction, construction of R-table, seal recognition and detection approach. The former two allow to represent model shapes in a compact way. The latter two are the steps when a query is formulated into the image database. Let us further describe the above steps in the following subsections.

7.3.1 Feature Descriptor of a Seal Shape

Our representation model for a seal is based on the connected components (classified as text characters) within seal region. Again, text characters are too local and very generic to be used for matching primitives. Due to natural spatial arrangement of text characters among them, we characterize the seal information based on relative position of these local text character shapes. The attributes used in our approach are described below.

1. *Reference point of a seal (R^S):* Given a seal image, the center of the minimum enclosing circle (MEC) of the seal is considered as the reference point (R^S). Note that, the relative position of R^S will be fixed for each class of seal shapes even if the seal is rotated.

2. *List of component pair:* To represent the spatial arrangement of the components present in the seal a list of neighboring component pairs is built. These neighboring component pairs are selected based on their proximity. Each component is associated to its n-nearest components. The neighbor components are decided using boundary growing algorithm from the boundary of the corresponding component (See Chapter 5 (Section 5.3.1)).

   Formally, a proximity graph $G_p = (V, E)$ is created to store these neighboring component pairs (here V = vertices and E = edges). An edge, $e_{ij} \in E$ is formed by a component pair (C_i, C_j) if components C_i and C_j are neighbors.

See Fig.7.4(a), where the n-neighbor ($n = 5$) components of 'U' (according to distance) are shown by joining line, which are 'L','L','F','A' and 'T'.

From each neighboring component pair, the location of the reference point (R^S) of the seal is designed. The spatial information of each neighboring component pair with respect to R^S is done by simple basic features namely: distance and relative angle which are discussed as follows.

3. *Distance:* We compute the CG (center of gravity) of each component for this purpose. Let, V_i^G represents the CG of component C_i. In Fig.7.4(b), the distance d_{ij} is shown for two components 'A' and 'D' of the seal. To obtain scale invariance, the distance is normalized based on character pair sizes.

4. *Relative Angle:* Two angles are considered for each neighboring component pair for detection purpose. For each pair of components we may consider a triangle by V_i^G, V_j^G with R^S (see Fig.7.4(b)). Angle α_i is formed at V_i^G by $V_j^G (j \neq i)$ and R^S. These angles are denoted by α_i and α_j. In Fig.7.4(b), these angles are "$\angle R^S AD$", "$\angle ADR^S$" respectively.

(a) (b)

Figure 7.4: (a) Character and its n-nearest neighbor are shown for the character 'U'. (b) A neighboring character pair and their relative angles are shown in a seal. Here, R^S is the reference point of the seal.

Given a pair of neighboring connected components of a seal model (m), we compute the feature information for our approach. The ASCII character labels of components are found by the recognition step using SVM classifier. Let L_i, L_j denote the ASCII character label of character component pair C_i and C_j. The distance and relative angles are calculated through their spatial organization. Thus, for each component pair C_i and C_j of the seal (m), the feature information f_{ij}^m is obtained as given below.

$$f_{ij}^m = (L_i, L_j, d_{ij}, \alpha_i, \alpha_j, m)$$

The feature of all neighboring character pairs characterizes the spatial description of seal. These features are stored in a R-table for indexing purpose.

7.3.2 Construction of the R-table

The feature information of each neighboring character pair of a seal is computed and stored in a hash table called as R-table of GHT using an efficient way. The

construction of R-table is performed as follows. We encode the character label and distance information for each neighboring character pair. A 3-dimensional Look-Up-Table (LUT) is used to obtain the index key (Φ^S) from each pair of character components as shown by Eq.7.2. K^S is the Look-Up function used for this purpose.

$$\Phi^S_{ij} = K^S(L_i, L_j, d_{ij}) \tag{7.2}$$

R-table contains the location of the hypothetical center of the seal model. We store the angle information (α_i, α_j) as function of Φ^S in the R-table. In Fig.7.5 we show the representation of the seal model in a R-table.

In a seal all of the alpha-numeric character pairs (62×62) do not occur. So, keeping all combinations of character pairs in that 3-dimensional LUT needs a huge memory. To reduce the memory space, we split the R-table in 2 different look-up-tables: Character Pair (CP) table and Distance (D) table. In the Character Pair table, all combinations of alpha-numeric character pairs are computed. When we find a neighboring character pair (L_i, L_j) in a seal, a table (D_{ij}) for distance is linked with the neighboring character pair. The Distance table (D_{ij}) contains the angle information of corresponding character pairs. The size of CP table is $|L|^2$, where $|L|$ stands for the total number of text labels. Thus, the memory size of our R-table is $(|L|^2 \times D_l)$, where D_l denotes the quantization parameter of the Distance table.

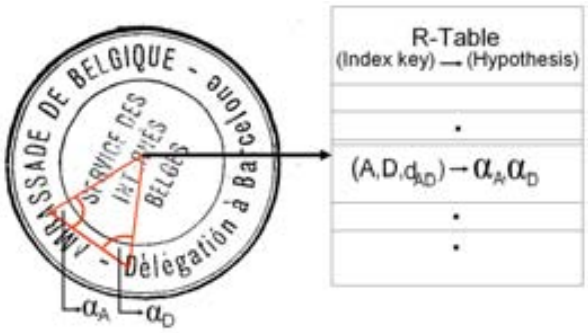


Figure 7.5: R-table representation from a seal model. Here, the triplet (A, D, d_{AD}) represents the index key for the character pair 'A' and 'D'. (α_A, α_D) are stored as hypothesis of the corresponding triplet of the index key.

For each index key in R-table, there may be multiple entries of angle information due to the presence of many similar neighboring character pair in seals. The collision in R-table is resolved by chaining. In Fig.7.6, we show the efficient data structure of the R-table. We also augment in R-table the model index along with angle information. We explain the R-table construction process in Algorithm 3. Finally, all neighboring character pairs of model seals are compiled and added to the R-table. Thus, an indexing scheme with all the seal models is constructed.

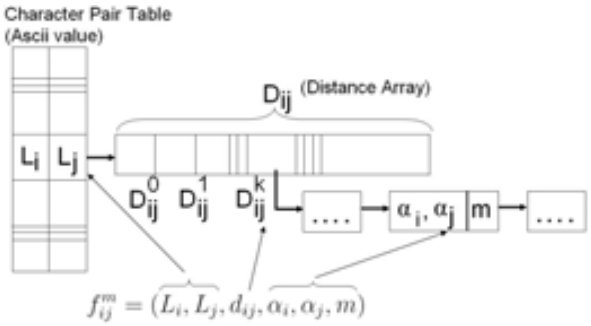


Figure 7.6: Efficient architecture of R-table and relation to feature information.

Algorithm 3 Seal description by local components

Require: A model seal object (m) composed of components list $C_1 \ldots C_t$ and a Reference point R_m^S

Ensure: Seal representation by spatial organization of local components

```
//Recognize character components
for all C_i of C_1 ... C_t do
    L_i ⇐ Text Label of C_i
end for
//Find center of gravity (CG) of components
for all C_i of C_1 ... C_t do
    V_i^G ⇐ CG of C_i
end for
//create list of neighboring component pairs
P_m ⇐ ∅
for all C_i of C_1 ... C_t do
    C_i1 ... C_in ⇐ n-nearest components of C_i
    for all C_j of C_i1 ... C_in do
        P_m ∪ (C_iC_j) if (C_iC_j) ∉ P_m
    end for
end for
//record spatial information
for all (C_iC_j) from P_m do
    d_ij ⇐ distance of (V_i^G and V_j^G)
    //Let Φ_ij^S be the index key
    Φ_ij^S ⇐ (L_i, L_j, d_ij)
    α_i ⇐ ∠R_m^S V_i^G V_j^G and α_j ⇐ ∠V_i^G V_j^G R_m^S
    R[Φ_ij^S] ⇐ R[Φ_ij^S] ∪ (α_i, α_i‖m)
end for
```

7.3.3 Isolated Seal Recognition

Although the final purpose is the retrieval of document images from a database, we present here GHT based approach for classifying isolated seal images to better assess

the performance of the recognition approach. To recognize isolated seal images, we first construct the R-table with a set of model seals. Next, the text components from the isolated query seal image are extracted using a text/graphics separation system (See Chapter 4). The local features are computed by spatial descriptor from each pair of characters in that seal as explained before. Here, the reference point is selected in the same way as the centre of minimum enclosing circle (MEC) of the seal. The character components provide high level spatial descriptor. For each neighboring character pair, we compute the index key (Φ^S) as given in Eq.7.2. Next, all angle information corresponding to each neighboring character pair and their distance are retrieved from the R-table. These angles are compared to the angle information for each neighboring character pair of query seal. If the difference of these two angles are less than T_α, we say a matching is occurred. The value of T_α ($= 1^\circ$) is set up empirically.

For recognition purpose, we create a table of all the model seals and initialize all with 0s. The counter of seal class is incremented whenever a matching entry of angle is found in the R-table. Finally, the classes are ranked in descending order by the number of voting. The model seal for which we get highest number of votes is selected as the recognized one for that corresponding sample.

7.3.4 Seal Detection in Document Database for Retrieval

To detect seals in a document image, at first a R-table is created from a set of predefined model seals. Model seals are segmented manually so that only text characters present in the seal part can be used in the R-table. The construction of the R-table is performed to record all spatial description of neighboring character pairs of each model seal as discussed in Algorithm 3. The constructed R-table will be used later to find hypothesis of the seal models in a document. For seal detection from an input document, we considered all characters paired with their n-nearest neighbors and based on information of the n-neighboring component pair using R-table, we cast vote for the hypothetical location of seal model. For a neighboring component pair (C_i and C_j) of the document, we compute L_i, L_j and distance d_{ij} between their CGs. Using this spatial feature corresponding to the neighboring component pair, the hypothetical location and seal model information are obtained. To achieve the hypothetical location, we obtain the index key Φ^S (see Eq.7.2) for each pair of text character. Next, we extract all the hypothesis of the spatial angle information and model index from R-table using Eq.(7.3). $R(\Phi^S_{ij})$ indicates the information of index key Φ^S_{ij} from R-table.

$$(\alpha_i, \alpha_j, m) \Leftarrow R(\Phi^S_{ij}) \tag{7.3}$$

These angle information (α_i, α_j) provide the location of hypothetical centre of the seal model (m). In our process, the voting is performed in pixel level. We show this process using a diagram in Fig.7.7. The voting mechanism accumulates evidences in the locations where we find the similar neighboring component pair having same spatial distance.

The presence of a particular seal on a document produces peaks in the GHT voting space. Since, we vote in pixel level, the peak is obtained by finding zones of high vote density. The more we find the matching of neighboring character pair descriptor,

the more accumulation of voting will be there. Finally, we detect the existence of the query seal using number of votes in local peaks of accumulation space. The presence of seal on a document is thus determined by searching total votes in each peak of the accumulator. The number of votes is normalized by the total number of spatial features of the corresponding seal model. If, f_m be the total number of spatial features of the query seal and f_d be the number of votes found in peak then we compute $100 \times f_d/f_m$. If this value is greater than T_m, then we accept the presence of the query seal in the document. The value of T_m $(= 30)$ is set up experimentally. In Fig.7.7, we show an example of voting procedure from our seal models. Fig.7.7(a) shows the construction of R-table from two seal models m_1 and m_2 with different neighboring character pairs. In Fig.7.7(b), we show how hypothetical locations are obtained from R-table from the character pairs of documents. Fig.7.7(c) shows a real output of voting regions after running the seal detection algorithm with a query seal m_1 of Fig.7.7(a).

7.4 Experimental Results

As application scenario, the approach described here has been applied to the historical archive of border records from the Civil Government of Girona[LKMS08]. It consists of printed and handwritten documents between 1940 and 1976. See Appendix A.5.3 for details. The documents were scanned at 200 dpi. These were in binary form due to the constraints of the digitization process. The documents are written in English and the seals posted on these documents also contain English text characters.

Our seal spotting method has been applied to retrieve the bundle of documents which got registered using similar seal. Two experiments namely: isolated seal recognition and seal detection have been designed to evaluate the performance of our approach. These are discussed as follows.

7.4.1 Performance of Isolated Seal Recognition

Though our objective is to detect the query seal in a given document, we will discuss here the pros and cons of our GHT based approach to recognize isolated seal symbols. We will also compare the performance of this local character based approach with other pixel based approaches, found in literature. For the experiment, we have created a database of isolated seals to test the seal recognition approach irrespective of isolated character recognition. This database contains 220 isolated seal images of 19 classes. The seals are extracted automatically from the archive using large components extraction method. This method may detect large components such as logo, signature, etc. Next, a manual checking is done to consider only the seal shapes. The seals are of different shapes and rotations. We noted that in our database there are mainly 3 types of seal shape, e.g. circular, elliptical and rectangular. It is to be noted that the seals contain different number of text characters for their description. The numbers of text characters present in each type of 19 classes of seal are detailed in Fig.7.8. Maximum (minimum) characters present in the seal of class 14 (3) and it contains 83 (16) characters. Different shapes of seals in the database are shown with different color in this figure. Table 7.3 shows the experimental outline. 5-fold

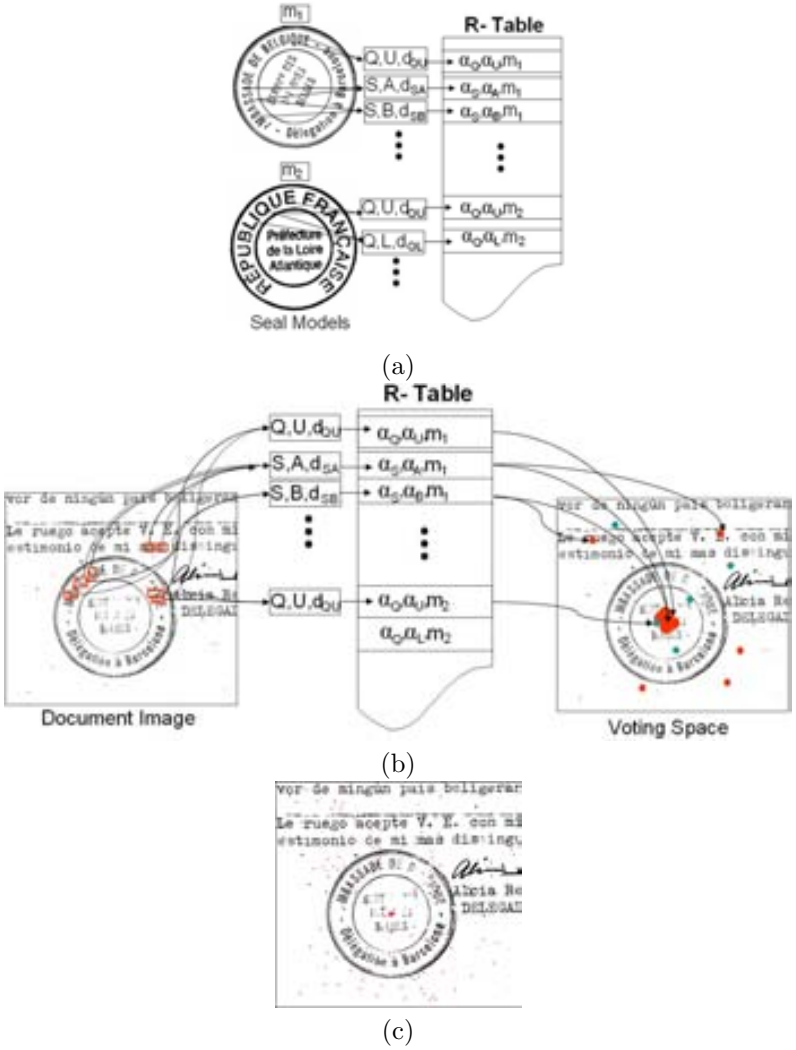


Figure 7.7: (a) Construction of R-table from two seal models m_1 and m_2 for GHT. (b) Hypothesis of Reference Point of seals are obtained from models' R-table. Red and green colors in voting space indicate the hypothesis of model m_1 and m_2 respectively (designed synthetically). (c) Voting areas (Red color) in document after running the seal detection algorithm with the query seal m_1 of (a).

cross validation scheme is used to evaluate the result. We have done two different experiments to evaluate our approach. One considering seal as a whole symbol and general pixel based symbol classification techniques are applied for this purpose. The other one considering GHT approach. Local text characters are used for providing the spatial information of the seal. We detail these two experiments in the following sections.

Table 7.3: Experiment outline.

Dataset	220 isolated seal images of 19 classes
Objective	To test the robustness of isolated seal recognition
Metric	5-fold Cross Validation

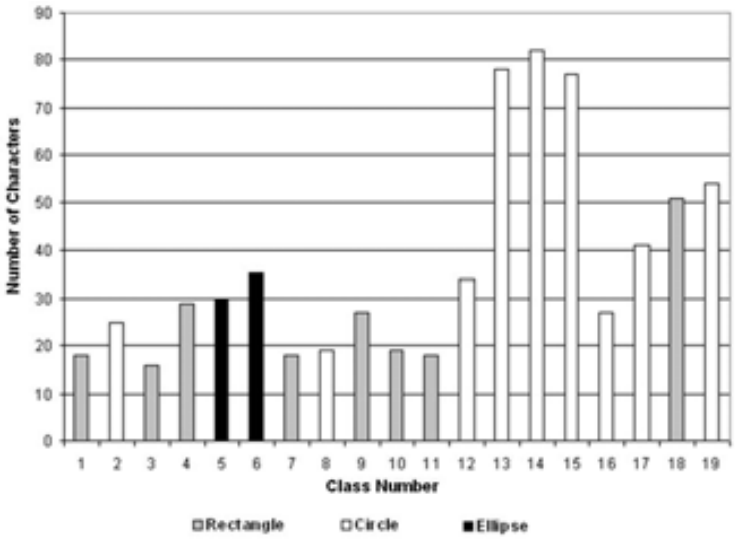


Figure 7.8: Number of text characters present in 19 different classes of seals are shown.

To get an idea of the quality of data used in our experiment, some samples of the data are shown in Fig.7.9. In Fig.7.9(a) we show different seal data of having similar outer boundary shape (circular). From the experiment we noted that our method can tackle this kind of images. It can be seen that although these seals have similar circular boundary, the text information within them is different. They contain a set of text characters along with a symbol (sometimes). Text strings are designed in different orientations and sizes. In Fig.7.9(b) we show a class of oval-shaped seal having rotation and noise (missing seal information or presence of other text characters from document).

Pixel based classification approach

To assess the usefulness of text information in seals, initially we performed a direct global classification test to the seal dataset at pixel level as features. In this experiment text identification are not used to evaluate the effectiveness of our GHT based approach which is discussed later. Here, we consider the seal as a symbol of texture only and different feature descriptors are applied to recognize such texture base symbols. The classification is performed using different rotation invariant shape descriptors. To get the idea of recognition results of different rotation invariant de-

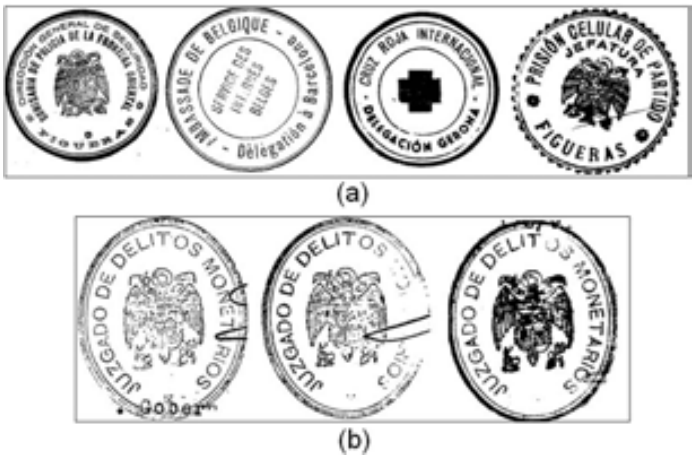


Figure 7.9: (a) Different types of circular seal. (b) Different noise level in a single class of oval-shaped seals.

scriptors, we present a comparative study on this dataset with different descriptors like (a) ART [RCB05], (b) Hu's Moments [Hu62], (c) Zernike (Zn) moments [KH90a], (d) Fourier-Mellin [AOC$^+$00], (e) Angle based approach and (f) Shape Context [BMP02]. The features obtained from each shape descriptor are fed to a classifier for matching. A model (prototype) seal for each class is selected. These models are chosen according to cross-validation scheme. We have used Euclidean distance based measures for this purpose. We measure the distance for each data from each model. Next, we assign the level of the model to the data according to minimum distance. In Table 7.4, we show the recognition accuracy obtained using different shape descriptors at different coarse level (up to 5 choices). From the experiment we noted that Angle based approach gives better results than ART, Hu, Fourier Mellin and Shape context. We also note that Zernike moment feature descriptor performs better than other descriptors in fine level (top 1 choice). When we increase the number of top choices, the difference of recognition accuracy between Angle-based approach and Zernike moment decreases. When 5 top choices are considered, angle based approach gives better result than Zernike (see Table 7.4).

Table 7.4: Comparison of seal (as a whole) recognition accuracy of different approaches.

Number of Top Choices	ART	Hu moment	Zernike moment	Fourier Mellin	Angle based Approach	Shape Context
1 choice	$< 5\%$	$< 10\%$	50.22%	$< 5\%$	36.65%	24.89%
2 choices	$< 5\%$	$< 10\%$	68.72%	$< 5\%$	60.18%	56.61%
3 choices	22.74%	10.42%	73.93%	$< 5\%$	69.19%	63.42%
4 choices	27.01%	19.43%	79.14%	$< 10\%$	73.93%	70.36%
5 choices	30.33%	24.64%	83.41%	12.32%	83.89%	78.57%

For one top choice, the highest performance of seal recognition considering seal as a whole symbol is only 50.22% and this is obtained from Zernike moment. These low performances using above descriptors are because a seal defines a symbol description with its rich internal textual structure. It contains a bigger boundary frame and smaller text components. Global pixel based shape descriptors are not enough to discriminate them properly. The confusion arises between the seal shapes which have similar boundary shapes. The text characters which convey the major information in a seal are not represented well in general pixel level shape descriptor.

GHT based classification approach

Next, we test the recognition of isolated seals using the GHT approach described earlier. The recognition is performed using local features computed from text characters (small components) of seal only. The text characters are labeled using SVM. The outer frame and other long graphical lines of seal are not considered. To have an idea about the effectiveness of our method on poor images, we show some noisy seal images in Fig.7.10, where our method performed correctly. We also have shown examples in Fig.7.11 where our method failed. Here, in the first seal image, many of the characters are overlapped with long graphical lines. After separating text characters using text separation algorithm, these text characters were not extracted, so the proper seal model is not found in this sample. In 2nd seal, many text characters are missing in the input sample. In 3rd seal, many characters are broken and overlapped with other text (it may happen while pressing the seal). Since our method is based on component level, if characters are broken the method may not work properly.

Figure 7.10: Some examples of isolated seals recognized correctly by our proposed scheme.

We show the seal recognition results using different number of top choices in Table 7.5. We have obtained 92.42% accuracy on taking only one top-choice of voting using

Figure 7.11: Some examples of seal where we got less votes using our approach.

Angle-based features. We obtained upto 98.58% accuracy using top 5 choices (coarse level recognition). We have also tested the seal recognition using text characters labelled by Hu moments and Zernike moments descriptors. These results are also shown in Table 7.5 to compare how different text recognition labelling can affect the seal recognition. To get the idea of the recognition accuracy according to different shapes of seal (though this boundary information is not considered in our approach) we show the results of our proposed method in Table 7.6. The recognition results depict that, the proposed approach is not sensitive to the text strings' global position, i.e. whether the strings are posted within seal in linear or curve way. As explained before, the recognition is done based on number of votes for each class. When we rejected 13.12% of samples based on less votes for recognized class, we got 95.31% accuracy using first choice only.

Table 7.5: Seal recognition accuracy using GHT by local connected components

Number of Top Choices	Angle-based Approach	Hu moment	Zernike moment
Only 1 choice	92.42%	62.12%	73.68%
Only 2 choices	96.68%	73.94%	82.61%
Only 3 choices	97.16%	80.56%	85.37%
Only 4 choices	98.11%	83.88%	88.56%
Only 5 choices	98.58%	87.21%	91.78%

Table 7.6: Recognition accuracy according to different boundary shapes

Rectangular	Circular	Elliptical
93.61%	91.74%	87.50%

From the experiment we noticed that most of the errors occur because of the presence of similar text string in some of the seals. The pair of seal models from which main confusion occurred are shown in Fig.7.12. Here, we see in both of the seals, the string "DIRECCION GENERAL DE SEGURIDAD" exist apart from other text lines. Though the font styles are different, we get similar information from the strings using SVM classifier. Due to noise, when other string characters are not recognized properly, we get similar voting for these models. The confusion rate obtained from these seals pair is found to be 1.37% when it is computed from overall samples of the dataset.

This confusion arises mainly due to unavailability of expected text information from seal. In this kind of images unlike pixel based systems our approach searches for the possible model seal where it finds maximum text character based spatial matching, which is another advantage of our approach.

Figure 7.12: Model of seal pairs where confusion occurred.

7.4.2 Performance of Seal Detection Result

To test the seal detection approach, we have constructed a database with 370 documents containing seals of English text characters. The database sometimes contains document images in upside down way. The seals are posted in different orientations and in different locations of these documents. There were missing of some seal information many times by noise or overlapped signature. Sometimes, due to posting seal on text document, additional text characters of document are also placed inside the seal region. We have considered additional 160 documents in this dataset which do not have the seals. Thus, our seal dataset consisted of 530 documents having different complexity. Table 7.7 shows the experimental outline. We have used Precision-Recall metric to measure the performance of the result.

Table 7.7: Experiment outline.

Dataset	370 documents containing seals + 160 documents without seal
Objective	To test the robustness of document retrieval using seal detection
Metric	Precision-Recall

In our experiment, here at first, we have provided some qualitative results to show how query seals are detected in documents with our approach. Next, we evaluate the performance of seal detection in the dataset of our experiments with precision-recall curve. We compared the seal detection results using 3 different multi-oriented text descriptors namely : Angle-based approach, Zernike moments and Hu moments. We have also discussed the performance of seal detection method based on the selection of different number of neighboring character pairs. We also provide a comparative result with patch based descriptors like SIFT [Low04] and shape context [BMP02]. Finally, we give an idea of the computation time of different methods we tested here.

In Fig.7.13, we show seal detection results of our approach when four different documents are tested. We used angle based features and SVM to classify text characters. As explained before, the location of a query seal is detected in these documents

by our GHT approach. To visualize the detection results, we draw a circle of radius of MEC of the query seal in that particular location. It is appreciable to see that our seal recognition performs well when there exists signature or other text overlapping with seal region. Here, Doc#1 and Doc#2 have two different seals though they share similar shape structure (circle). Using our approach, we correctly distinguish them and label the associate seal model with it. In Doc#3, it is shown that the query seal is detected in the document in spite of 180 degree rotation of the document. In Doc#4, a date was affixed during stamping and because of this, seal region contained additional alpha-numeric characters. We noted that, our approach can detect the seal from these documents having variable date field in the seal region.

Figure 7.13: Detection of seals in documents

Given a query seal image, a ranked list of retrieved documents from our database are shown in Fig.7.14. The ranking is done based on voting in accumulation array. It is understandable from the ranking of these images, how noise, overlapping and occlusion do affect the ranking process. Here, we show five results of the ranked documents, retrieved by querying a seal model. We see, in Fig.7.14(S1_3), a seal was affixed 2 times and they have some overlapped portions. In Fig.7.14(S1_5), the approach could detect the seal although there were many text characters missing.

Another advantage of our method is that since it works considering only labeled text characters, we can use this to detect arbitrary shape of seal.

Figure 7.14: Ranking of different documents of a seal model.

The detection accuracy is affected mainly due to broken text characters in the seal and the presence of long graphical lines over text regions (see Fig.7.15). In Fig.7.15(a), only a few characters are extracted after text/graphics separation algorithm because of broken text. Thus we got fewer votes. The seal of Fig.7.15(b) is also difficult to recognize for human. Due to noise when characters are broken and the broken components of a character could not be joined through preprocessing, the character labeling process might be wrong. Hence less accurate will be the seal detection because, there will be fewer voting accumulation due to small number of text character pairs.

Figure 7.15: Two documents having very low votes by our approach.

To evaluate the performance of the system with a query seal against a collection of documents, we use common ratio of precision (P) and recall (R) for evaluation of retrieval of documents. In Chapter 4, we explained the computation of precision (P) and recall (R). In Fig.7.16, we compare the performance of seal detection system using two different text character shape features, namely angle-based approach, Zernike moment and Hu moment. The precision is obtained 100% using these features during the recall value until 20%. Then the performance on Hu and Zernike moments based seal retrieval got degraded. The reason of lower performance is the poor recognition of text characters. We compared Precision, Recall and F-ratio using both these approaches. The F-ratio is the harmonic mean of precision and recall and it is computed as follows.

$$F = 2 \times \frac{precision \times recall}{precision + recall}$$

In Table 7.8, we show the comparative result of overall performance using these two character descriptors.

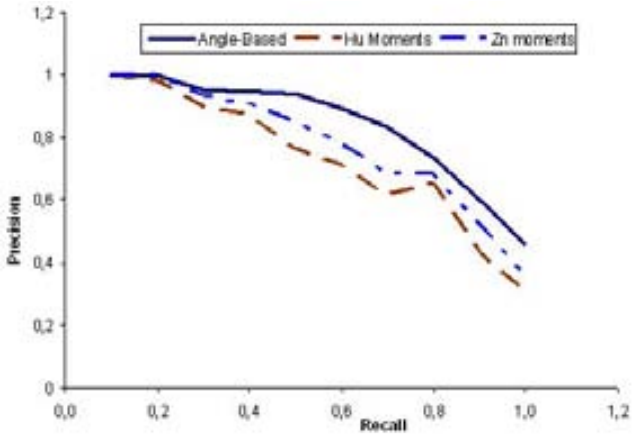


Figure 7.16: Precision-recall curve of document retrieval using different character recognition features.

Table 7.8: Performance of seal detection system

Shape Descriptor	Precision	Recall	F-Ratio
Angle-based	91.66%	91.91%	91.78%
Hu moment	85.35%	83.97%	84.65%
Zernike moment	89.78%	89.23%	89.50%

It is mentioned in Section 7.3.4 that a list of neighboring character pairs are found from each text component using its N nearest neighbors. In Fig.7.17 we show the average precision and recall plot of a query seal model in the whole database using different number of neighbor component selection of our proposed approach. The selection of number of neighbor character is checked because, the spatial organization of the neighboring character pair depends on this criterion and based on each character pair, the location of hypothetical reference point is voted. So, the more we select the number of neighboring character pair for each character, the higher the performance of the document retrieval system.

As discussed before, the documents in the historical dataset are sometimes degraded and as a result the text characters are broken or touched with graphical lines. If the local text characters are not segmented or labeled correctly, our method may not work properly. When we use less number of neighboring character pairs, it may fail due to improper hypothesis generation. Hence, a valley appears in the precision-recall curve (see Fig.7.17). When the choices of neighboring characters are more, the system is benefited from the other text characters (far neighbour) and the proposed approach is able to improve its performance based on the spatial relationships of other characters. Hence, with more neighboring character pairs, our system has more

strength to detect the seal in such documents and thus the valley slowly vanishes.

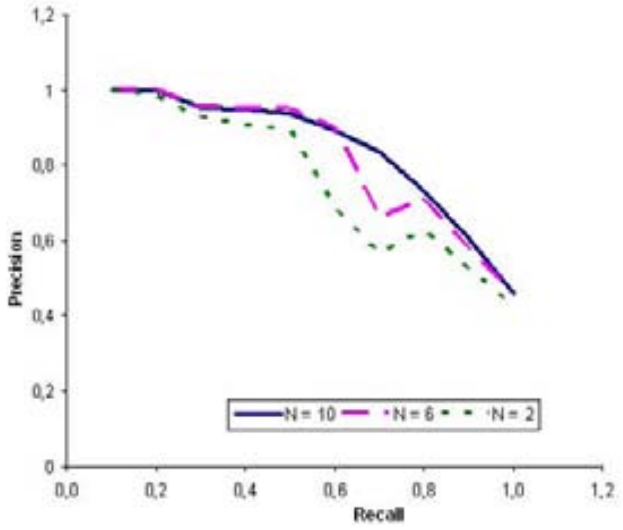


Figure 7.17: Precision-recall curve of document retrieval using different choice of neighbor.

We have compared our result with object detection methods like SIFT [Low04] and shape context [BMP02]. In Fig.7.18 we show the average precision and recall plots of seal detection results obtained from our database. The procedures to detect seals using these descriptors are designed as follows. A Harris corner detector is used to detect the invariant key points. RANSAC fitting algorithm [FB81] is then used to filter some false matches. From the result, it is noticed that the seal detection approach based on local character recognition shows better precision accuracy than other methods till 90% recall of the dataset. After that, recognition of text characters within seal fails due to noise. So, the precision accuracy curve of seal detection of angle based approach falls rapidly. Shape context did not provide good results in the experiment.

The performance of seal detection approach using text component extraction found better than patch based descriptors like SIFT. It is to be noted that using text character based seal detection scheme, we also have a semantic layer among the text characters in the seal. The text string found using recognized characters can provide a description of seal which is an added advantage of our approach.

Finally, we give an indication about the system's computation time. When testing the 530 documents in seal dataset for a query seal, the proposed method spent approximately 3.4 seconds on an average to detect seal in each document (on a PC of 2.00-GHz T6400 Intel Core 2 Duo Processor). In Table 7.9 we compare the approximate processing time to run the same using different approaches.

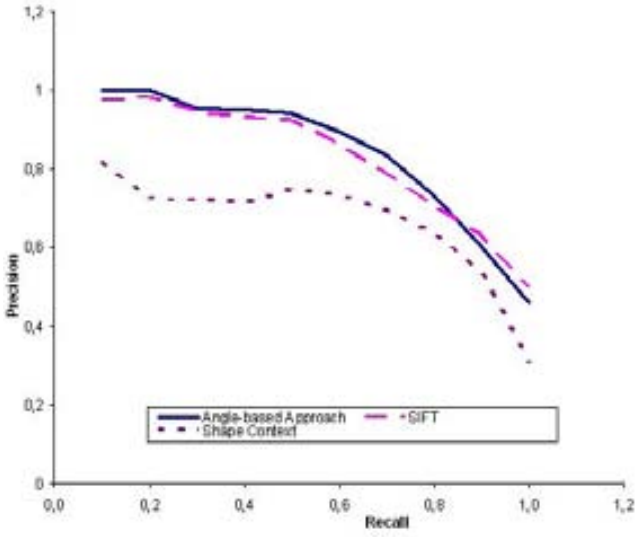


Figure 7.18: Precision-recall curve using different methods.

Table 7.9: Computation time (in seconds) of seal detection system

Patch-based Descriptor		Text-based Descriptor		
SIFT	Shape Context	Hu moment	Zernike moment	Angle-based Approach
2.1	2.7	3.3	3.9	3.4

7.5 Conclusion

In this chapter, we have presented a seal detection approach based on spatial arrangement of seal content. A query seal is translated into a set of spatial feature vector using its text character information. These features are used later to locate a seal of similar content in documents. We have used a multi-oriented and multi-scale text character recognition method to generate high level local feature to take care of complex multi-oriented seal information. The recognition result of these character components within seal region is used to generate local spatial information to classify and detect the seal. Relative positional information of text string characters is used for this purpose and hypothesis were generated based on that.

The existing approaches of seal detection consider the seal as a whole object. Use of rotation invariant features [MY04] to recognize whole seal may not work properly for the seals which are having similar boundary frames, frequent missing of text characters or inclusion of background text characters. The approaches based on seal frame detection [ZJD06] can not recognize the seal type in the document. Also, color based approaches [UMM98] do not work for degraded binary images. Our approach of seal detection based on GHT of local text characters overcomes such problems efficiently. Moreover, our approach can detect the seal even if all the characters from

the seal are not extracted.

We tested our method in a database of multi-oriented seal documents containing noise, occlusion and overlapping. In retrieval of document images from database, all components of a document pass through the recognition process and hence it is time consuming. So, the improvement of the performance in terms of time could be achieved if a pre-processing method (run-length smoothing) is applied to remove non-seal information from the document. Also the use of boundary-shape information (circular, rectangular, etc.) of seal may improve it further.

Chapter 8

Conclusions

In this chapter we summarize the contributions of this thesis to the text character extraction and recognition problem and also to the application of retrieval of graphical documents based on recognition of text characters. Next, we present discussion and some limitations of the proposed approaches. Finally some possible future directions and some improvements of the proposed methods are discussed.

8.1 Contributions

In this thesis, we have introduced a framework for text character information extraction. We also presented two applications of graphical document retrieval based on such text information. As explained in Chapter 1, our work has been motivated by the problem of text extraction and retrieval of graphical documents. A lot of interest is made worldwide for mass digitization of document collections and their storage in digital libraries. It results the digital repositories become rich in information if these information are semantically accessible. From a methodological point of view, the main challenges appear in the difference in the structure of the graphical documents and simple text documents.

The commercial OCR techniques may fail in graphical documents where text characters are oriented and designed according to subject's interest. In graphical documents, text lines appear frequently in different orientations other than the usual horizontal directions. Also, text and graphical line overlap many times. To cope up these problems, we have contributed towards the interpretation and retrieval of graphical documents along this thesis. Let us briefly summarize these contributions.

First, we have presented a novel feature shape descriptor based on angle information for **text character recognition** in graphical documents. The proposed system does not depend on the skew of the document and handles multi-oriented text as well as normal text. Size and rotation invariant features are considered for the recognition of multi-sized and multi-oriented characters.

As part of the OCR process, touching character segmentation is a major task for

achieving higher character recognition rates. Character segmentation is thus applied to word images for this purpose. We have proposed a scheme for **touching character segmentation** of recognizing touching strings in multi-oriented and multi-sized environment. The algorithm at first segments the touching character into primitives and then finds the best sequence of model character shapes based on dynamic programming algorithm using these primitive segments.

Next, a combination of bottom-up and top-down approach has been investigated in the context of text **character separation/spotting** in graphical documents. Here, at first, we grouped the connected component present in the document. Connected component analysis, skeleton information, geometrical features, etc. are used to identify text and symbol from graphics. We have adapted SIFT for this application to locate multiple instances of text characters which are overlapped with graphical lines in different poses such as scale and rotation.

We have presented a new approach for **text line segmentation**. A key characteristic of our approach is the use of foreground and background information for this purpose. Background information is usually neglected by researchers. In our approach, the background information is taken care using water reservoir concept which guides our process to follow the text line. The approach is based on perceptual laws of text characters in a line and it is easy to apply. We have demonstrated the invariance nature towards rotation, scale, and affine transform to real world images. One of the significant advantages of the proposed method is its flexibility. Our scheme is independent of font, size and style of the text characters.

Next, we have proposed a **word searching** algorithm based on spatial arrangement of character components in graphical documents. The text labels of connected components and their pair information are used to index local spatial information from them. Relative positional information of text characters in a string is used for this purpose and hypothesis were generated based on that. One advantage of our approach is that the system can detect the query word even if we do not extract all the text characters of a query string.

Finally, we have presented a **seal detection** approach based on spatial arrangement of seal content using GHT. A query seal is translated into a set of spatial feature vector using its text character information. These features are used later to locate a seal of similar content in documents. The character components within a seal region are used to generate local spatial information to classify and detect the seal. Relative positional information of text string characters is used for this purpose and hypothesis were generated based on that.

8.2 Discussions

In this thesis we have made contributions in different stages of text analysis methods and the application of such methods for document retrieval system from a collection of graphical documents.

In the text information extraction part of the thesis we presented four different stages: character recognition, touching n-character segmentation in multi-oriented environment, text character segmentation from overlapped graphical lines and curvi-

linear text line extraction. These methods improve the state-of-the-art approaches to improve overall interpretation in graphical documents.

Multi-oriented isolated character recognition approach has been used to drive different modules in our thesis. We have tested this approach on character datasets with two different Indian scripts namely: Devnagari and Bengali, besides Latin text. We also presented the efficiency of this approach by evaluating with MPEG-7 dataset and handwritten music symbols for its competence.

Although many techniques are available for segmentation of horizontal touching characters, to the best of our knowledge there is no work towards multi-oriented touching character segmentation. We have compared the segmentation approach with different character recognition approached such as Angle based approach, Fourier Mellin transform. Though, the work has been done in multi-rotation environment, the angle of inclination of the touching string has been used to arrange the primitive segments in sequential order. When facing real data arising from different orientations of characters in a string, this assumption restrict the possibility of flexibility and the performance of the character segmentation system drops.

For separating text characters in graphical documents, our system combines bottom-up and top-down procedures to take care of this complex problem. It must be said that, although experimental results show that the touching character spotting has good performance, it has some limitation when it is applied to very degraded documents. Obviously, detection of the long graphical line separation suffers when the image is degraded. In this kind of images, the thinning technique may not produce good skeleton image. Hence, it may produce many small segments. In addition, SIFT approach produces many false alarms which need other post-processing steps for elimination.

In our proposed methodology of text line separation, the assumption was taken that the text is present in foreground layer of the documents. Thus, the images were binarized by adaptive threshold to convert the image into two-tone images. If the image is in reverse contrast, i.e. black background and white foreground, our present method can not handle. Our methods, thus, depend on efficiency of binarization approach. Also, our system will not work well with color information in the document. We have converted color images to binary images using a typical color to gray image conversion technique and used the binary images for different experiments. More focused color segmentation methods, precisely for documents can be useful for better results. Most of the errors in our approach of text line segmentation are due to over-segmentation of the characters in the text line. Many text lines are over segmented and could not be joined together with the preprocessing system we have used [RPC04].

One of the novelty of this thesis is the detection of words on-the-fly in graphical documents by segmenting and validating them at the same time. There are not many published works with a complete system which can deal with word searching in graphical documents. It is due to several limitations when trying to apply in large collections of graphical documents. Most of these existing work [DMS95b] proposed in the literature considered a fixed set of document with a single font of characters for their experiment. Though our approach has limitations, we believe it is a step forward in the analysis of graphical document applications.

The seal detection approach deserves its novelty due to the complexity of the

problem. The existing approaches of seal detection consider the seal as a whole object. Use of rotation invariant features [MY04] to recognize whole seal may not work properly for the seals which are having similar boundary frames, frequent missing of text characters or inclusion of background text characters.

We have formulated our approach to be scalable and adaptable to process large collection of documents. It does not need a previous segmentation, thus provides direct focus on seal detection. The approach is robust to detect seal in noisy, cluttered document. We can therefore mention that combining individual character recognition with a SVM and relational indexing using GHT method, our work can be classified as a combination of a statistical and a structural approach.

The document retrieval performance is affected mainly due to broken text characters in the document and the presence of long graphical lines over text regions. As our methods are based on text character recognition, due to noise when characters are broken and the broken components of a character could not be joined through preprocessing, the character labeling process might be wrong. Hence the applications of document retrieval based on characters are sometimes erroneous.

8.3 Future Lines of Research

The research carried out in this thesis has opened new perspectives and scope of improvements in some of the points that are discussed below.

Text/graphics analysis is difficult problem in graphical documents. Thus, coordination between different levels of analysis is an important issue to research further. The use of **context information** may solve this problem. The data transfer between different steps of document analysis will improve the interpretation result. If a word is partially extracted then this extracted information would be helpful in finding the search area to look for other missing parts.

Since similar characters are grouped together in same class, the exact ASCII code of character is not obtained from our character recognition process. It affects the overall word retrieval process. It is because, the total number of characters is reduced to the size of that of multi-oriented characters. Also, due to the rotation invariance nature, the orientation information of character is missing. The **orientation direction** may provide better information for the character alignment, hence that of character string. Because, characters appear in string by nature. Thus, considering the direction of recognized character, the context information will be helpful to interpret string better.

Though we have used two different Indian scripts namely Devnagari and Bengali other than English script for performance evaluation of character recognition, these languages have not been used in other methods of our thesis. Different methodologies for text/graphics analysis need to be tested on **graphical documents of different scripts**.

For example, in India, a seal document may contain words in two or more language scripts. So, multi-script OCR is necessary to process these seal images. For OCR development of such multi-script seal, it is necessary to separate different script words before feeding them to the OCRs of individual scripts. We plan to work on multi-

script separation for such graphical documents.

Color information is often used to describe text and graphical symbols in graphical documents. As mentioned, our methods depend on character segmentation in foreground layer of the document. If the foreground text information is in different color, our method may not handle them properly. Proper color separation will be helpful to handle them better. Thus, efficient color segmentation methods in such documents will improve the overall performance of interpretation.

Performance of document retrieval in terms of **time complexity** should be improved. As our document retrieval approaches depend on character recognition module that is time consuming. There is a need to focus on reducing time by efficient use of the recognition approach.

Finally, as it is mentioned in the Appendix, we have constructed our own data from maps, electrical diagrams, etc. In addition, even we have used quite large databases, the methods should be tested on mass data in order to assess their **transference to real systems**. The scalability of the proposed methods should be checked by analyzing other kind of technical documents such as electronic diagrams, mechanical designs, architectural diagrams, etc.

In this thesis work, we have proposed two different applications of retrieval approaches based on text character information. We are planning to extend this work for more robust document retrieval on archive of documents or document routing in administrative departments. We believe that, our work will encourage the activities of document retrieval based on graphical symbol (seal) and query text in archive of graphical documents printed in other languages and English in particular.

Appendix A

Databases

During the research of this work, several databases have been used, namely multi-oriented isolated character symbols, touching characters, graphical documents, camera based warped documents, seal database. Some of these datasets have been created for validating the proposed methodologies. In this Appendix, we will explain in detail all the datasets used in our experiments.

A.1 Multi-Oriented Isolated Characters

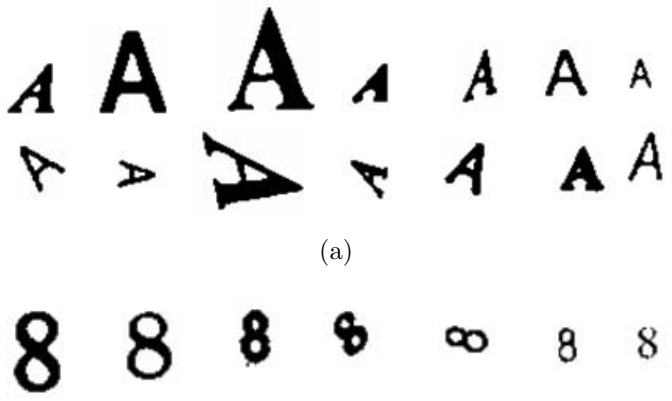
A.1.1 English Text Character

For the experiment of English character recognition, we considered data from two different sources. One source of data is from graphical documents (e.g. map, seal documents) and its size is 8,250. An automatic text separation method [RVL+07] has been used to extract characters from graphical documents towards the development of this dataset. In Fig.A.1, we show sample images of different styles and fonts of English letter 'A' and digit '8'. Second part of the data was the data used in Pal et al. [PKRP06] and the size of this data is 18,232. This collection is mainly from journals, magazines, newspaper, advertisements and computer printouts. The text characters are of arbitrary rotation and scale in both the datasets.

A.1.2 Text Characters from Indian Scripts

The alphabet of the modern Bengali script consists of 11 vowels and 39 consonants. These characters may be called basic characters. Out of the 49 basic characters of Devnagari script, 11 are vowels and 38 are consonants. The basic characters of Bengali and Devnagari scripts are shown in Fig.A.2(a) and Fig.A.2(b) respectively. Writing style in both the scripts is from left to right. The concept of upper/lower case is absent in Bengali and Devnagari scripts.

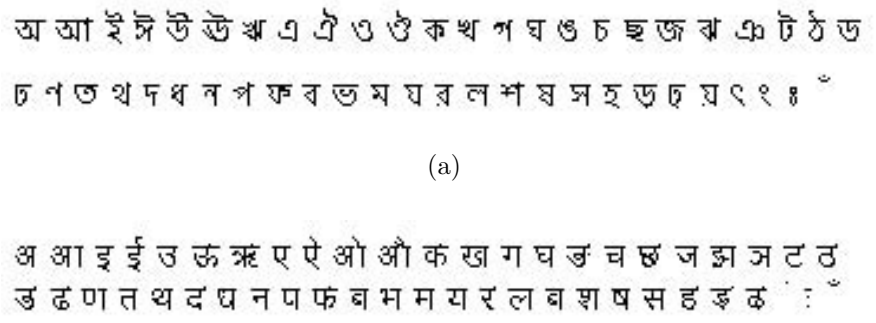
In both Bengali and Devnagari scripts a vowel following a consonant takes a mod-



(a)

(b)

Figure A.1: Different styles and fonts of (a) letter 'A' and (b) digit '8'.



(a)

(b)

Figure A.2: Basic characters of (a) Bengali and (b) Devnagari alphabet are shown. First eleven are vowels and rest are consonants in both the alphabets.

ified shape. Depending on the vowel, its modified shape is placed at the left, right (or both) or bottom of the consonant. These modified shapes are called modified characters. Examples of modified character are shown in Fig.A.3. A consonant or vowel following a consonant sometimes takes a compound orthographic shape which we call as compound character. Compound characters can be combinations of two consonants as well as a consonant and a vowel. Compounding of three or four characters also exists in these two scripts. In both the script forms, there are about 280 compound characters [CP98].

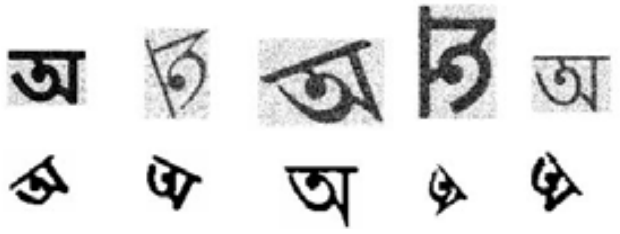
Languages like Hindi, Nepali, Sanskrit and Marathi are popularly written in Devnagari while Bengali, Assamese and Manipuri languages are written in Bengali script. Moreover, Hindi and Bengali are the national languages of India and Bangladesh, respectively. Also, Hindi is the third most and Bengali is the fifth most popular

Bangla vowels	আ	ই	ঈ	উ	ঊ	ঋ	এ	ঐ	ও	ঔ
Modified shape	া	ি	ী	ু	ূ	ৃ	ে	ৈ	ো	ৌ
When attached to consonant क	का	कि	की	कु	कू	कृ	के	कै	को	कौ
Devnagari vowels	आ	इ	ई	उ	ऊ	ऋ	ए	ऐ	ओ	औ
Modified shape	ा	ि	ी	ु	ू	ृ	े	ै	ो	ौ
When attached to consonant क	का	कि	की	कु	कू	कृ	के	कै	को	कौ

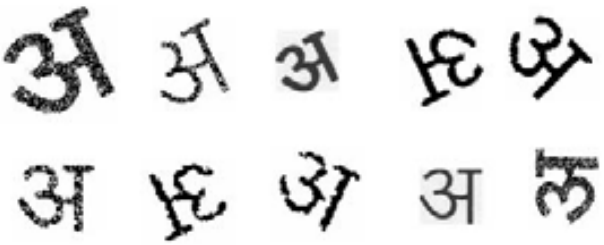
Figure A.3: Examples of Bengali and Devnagari modified characters

language in the world [CP98].

We constructed the dataset of multi-oriented Bengali and Devnagari characters. Documents of different orientations from newspapers and computer printed materials of Bengali and Devnagari scripts are collected. The documents contain four popular fonts of both Bengali and Devnagari scripts. Different font sizes like, 12, 16, 20, 26, 30, 36 and 40 point-size characters are considered. The size of the dataset is 7,874 for Bengali and 7515 for Devnagari characters. Some of these data are degraded using Gaussian noise. We show some of the samples in Fig.A.4.



(a)



(b)

Figure A.4: Some examples of isolated characters from Indian language (a) Bengali (b) Devnagari.

A.2 Multi-Oriented Touching Characters

For the experiments of touching character segmentation, a database of touching components is constructed using real as well as synthetic data. The real data is collected from maps, newspapers and magazines. They are digitized in grey tone with 300 dpi. Otsu binarization method [Ots79] is applied to convert them into two-tone images. A text separation method [RVL+07] has been used to extract characters from documents, and the groundtruth has been generated manually. There are total 250 real words present in the database.

The synthetic data have been generated using the system described in [DPV+08]. This database is composed of single-word images with different scales and orientations, with corresponding groundtruth at character level. Two different fonts namely, Arial and Times New Roman fonts and 250 words for each font are used to generate the datasets. The data are produced at first in vector graphics form with the corresponding ground-truth. Next, vector graphics data are rasterized to obtain the test images. In each word, this data generation method looks for the pairs of successive characters, and makes them connected according to a Boolean value. Overlapping between characters is controlled using a Gaussian function. The words are selected from a dictionary (of 52 country names), with random scaling and rotation parameters and using predefined fonts. Each word consists of 7-8 characters in average.

(a)

(b)

Figure A.5: Some examples of n-touching characters in multi-orientation environment (a) Real images (b) Synthetic images.

A.3 Camera Based Warped Documents

Camera-captured document images due to its unconstrained hand-held, usually contain several types of degradations like perspective and geometric distortions, which are not common in scanned images. Thus, camera-captured document images contain non-straight textlines with high-degrees of curve in different directions. These warped document images reduce human readability as well as OCR accuracy when traditional OCR engines developed for scanned documents are used.

The camera based documents are taken from the CBDAR 2007 document image dewarping contest [SB07]. CBDAR 2007 dataset contains 102 grayscale and binarized document images of pages from several technical books captured by an off-the-shelf hand-held digital camera in a normal office environment. In Fig.A.6, we show two warped documents.

Figure A.6: Some examples of CBDAR 2007 dataset.

A.4 Direction Boards

We have used a dataset of direction board images, captured in color tone with 3 Mega Pixels digital camera from university campus. The images contain text lines, printed in a dark color compared to the background. These images contain several types of degradations like perspective and geometric distortions. There are total 20 different direction signs which contain 5 text lines in average. The images are captured in such a way, that the text lines in the direction board are with perspective and geometric distortions. In Fig.A.7, we show some direction board images.

Figure A.7: Some examples of direction board indications.

A.5 Graphical Documents

We considered documents of maps, electrical diagrams, seal documents in our collection for the experiment on graphical documents. These are discussed below.

A.5.1 Map Documents

Real Maps

The real data is collected from maps of 2 different resources. One set of maps are from geography books. The data were digitized by a flatbed scanner at 300 DPI. The average size of these map images are 1200 × 1200. We have considered 20 different real geographical maps for our work. There are approximately 35-40 words in each document.

The other set of real maps are selected from historical archive of project "Cartography". See Fig.A.9, where a portion of the document is shown. These images contain text printed in dark color compared to the background.

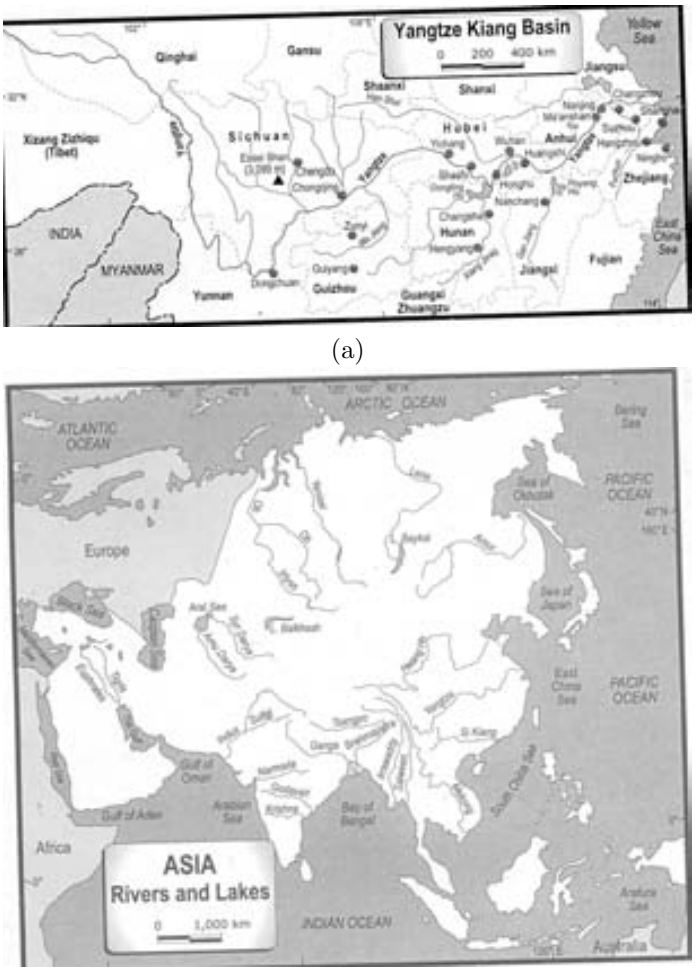


Figure A.8: Some examples of real maps in gray tone.

Synthetic Maps

The synthetic maps are generated using an automatic system described in Delalandre et al. [DPV+08]. The backgrounds (graphical lines, boundaries etc.) of these maps are taken from a few real maps and the text words of country/river names are placed in the foreground using a set of rules. The graphical long lines i.e. geographical borders and rivers in these maps are kept fixed. The text portions are randomly placed and oriented to generate different synthetic documents. Arial font is used for annotation purpose in these maps. The long graphical lines are overlapped with text in many places. The arrangement of characters in text strings are of both linear and curvilinear. There are total 200 maps in the datasets. We show two maps with different background in Fig.A.10.

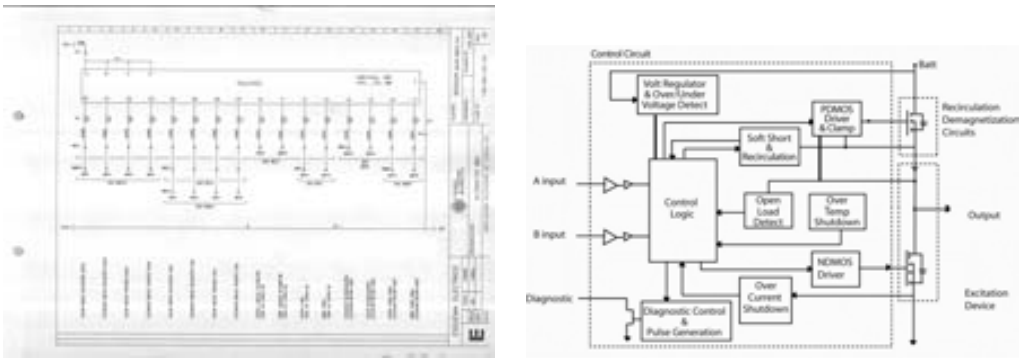
Figure A.9: Portion of a color map from historical archive.

(a) (b)

Figure A.10: Some examples of synthetic maps in binary image.

A.5.2 Electrical Diagrams

The dataset of electrical diagrams consisted from the collection of NISSAN dataset of CVC and some diagrams from CBDAR dataset. These documents are digitized in grey tone with 300 dpi. Average size of the images are 2500×3500. We collected 15 such diagram images to test our method. To have an idea, we show some images in Fig.A.11.



(a) (b)

Figure A.11: Some examples of electrical diagrams dataset.

A.5.3 Seal Documents

We have considered a historical archive of documents containing seal documents. The archive contains border records from the Civil Government of Girona[LKMS08]. It consists of printed and handwritten documents between 1940 and 1976. The documents are related to people going through the Spanish-French border. These documents are organized in personal bundles. For each one, there is a cover page with the names of people whose information is contained in this record. In each bundle there is very diverse documentation, so we can find safe-conduct to cross the border, arrest reports, information of professional activities, documents of prisoners transfer to labor camps, medical reports, correspondence with consulates, telegram, etc. This documentation has a great importance in the studies about historical issues related with the Spanish Civil War and the Second World War. From the digital archive of scanned document images it is interesting to extract information regarding people (names, nationality, age, civil status, dates, location of arrest, etc.) that can be used for indexing purposes. Depending on the location of office of registration, the documents contain different types of seal. The documents were scanned in 200 dpi and these are in binary form due to the constraints of the digitization process. Some examples of documents containing seals are shown in Fig.A.12.

In our database of seal documents, there are mainly 3 types of seal shapes, e.g. circular, elliptical and rectangular. Text information is annotated in straight or curve way according to the shapes of seals. The seals are posted in different orientations and in different locations. There is missing of seal information due to noise or overlapping signatures. Sometimes, due to stamping on to text documents, many additional text characters of the document itself are also present inside the seal region.

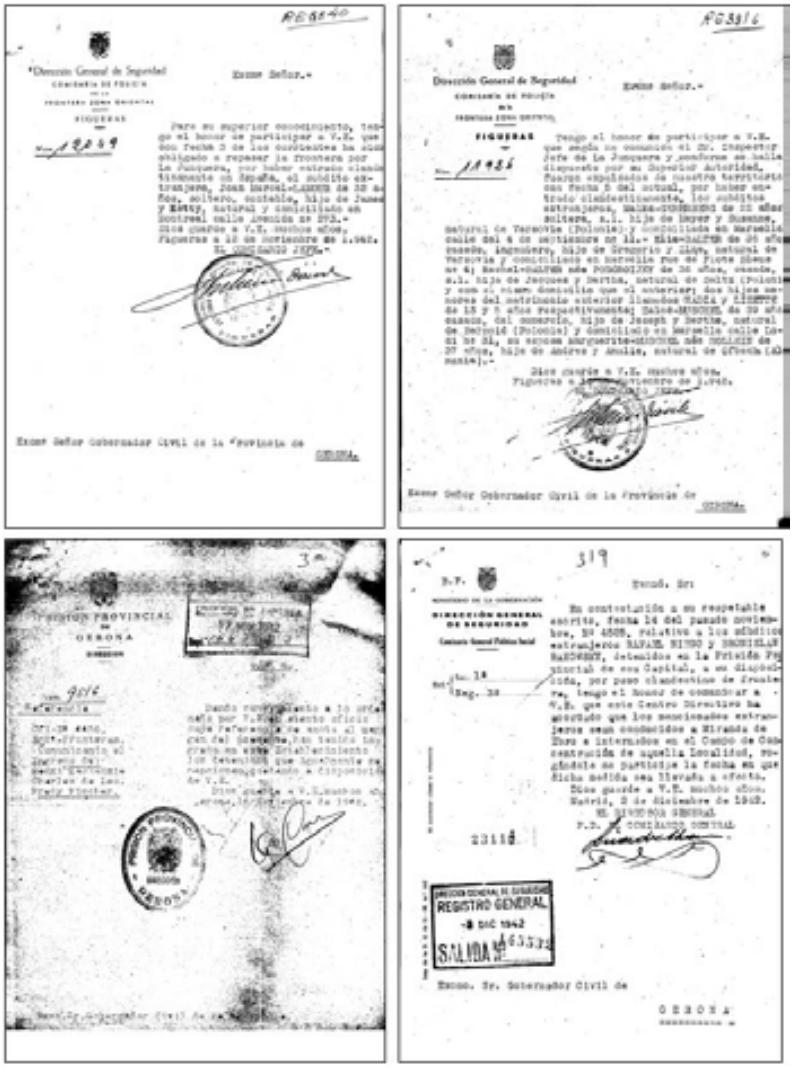


Figure A.12: Some examples of documents containing seal symbols.

Bibliography

[AOC+00] S. Adam, J. M. Ogier, C. Carlon, R. Mullot, J. Labiche, and J. Gardes. Symbol and character recognition: application to engineering drawing. *International Journal on Document Analysis and Recognition*, 2000.

[AW02] M. Ahmed and R. Ward. A rotation invariant rule-based thinning algorithm for character recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(12):1672–1678, Dec 2002.

[Bal81] D. H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111–122, 1981.

[Bis09] A. F. Bisquerra. *Writer Identification by a Combination of Graphical Features in the Framework of Old Handwritten Music Scores*. PhD thesis, Universitat Autonoma de Barcelona, 2009.

[BLGT06] M. Bicego, A. Lagorio, E. Grosso, and M. Tistarelli. On the use of SIFT features for face authentication. In *Computer Vision and Pattern Recognition Workshop*, page 35, USA, 2006.

[BM95] M. Burge and G. Monagan. Extracting words and multi part symbols in graphics rich documents. In *Proceedings of International Conference on Image Analysis and Processing*, 1995.

[BMP02] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(4):509–522, April 2002.

[BNS08] N. N. Bai, K. Nam, and Y. Song. Extracting curved text lines using the chain composition and the expanded grouping method. In *Document Recognition and Retrieval*, USA, 2008.

[BR92] D. Baumann and S. Ranka. The generalized hough transform on an mimd machine. *Journal of Undergraduate Research in High-Performance Computing*, 2:1–8, 1992.

[Bre02] M. Breeding. Preserving digital information. *Information Today*, 19, 2002.

[BSA91] S. O. Belkasim, M. Shridar, and M. Ahmadi. Pattern recognition with moment invariants: a comparative study and new results. *Pattern Recognition*, 24:1117–1138, 1991.

[BSB08] S. S. Bukhari, F. Shafait, and T. M. Breuel. Segmentation of curled textlines using active contours. In *8th IAPR Workshop on Document Analysis Systems*, pages 270–277, 2008.

[Bur98] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[BW97] H. Bunke and P. S. P. Wang, editors. *Handbook of Character Recognition and Document Image Analysis*. World Scientific, 1997.

[CC99] T. C. Chang and S. Y. Chen. Character segmentation using convex-hull techniques. *International Journal on Pattern Recognition and Artificial Intelligence*, 13(6):833–858, 1999.

[CDOM09] M. Coustaty, S. Dubois, J. Ogier, and M Menard. Segmenting and indexing old documents using a letter extraction. In *Workshop on Graphics Recognition*, pages 142–149, 2009.

[Che96] Y. S. Chen. Automatic identification for a chinese seal image. *Pattern Recognition*, 29:1807–1820, 1996.

[Che03] Y. S. Chen. Registration of seal images using contour analysis. In *Scandinavian Conferences on Image Analysis*, pages 255–261, 2003.

[CHTB94] W. B. Croft, S. M. Harding, K. Taghva, and J. Borsack. An evaluation of information retrieval accuracy with simulated ocr output. In *In the 3rd Symposium of Document Analysis and Information Retrieval*, pages 115–126, 1994.

[CK09] A. Clavelli and D. Karatzas. Text segmentation in colour posters from the spanish civil war era. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 181–185, Barcelona, Spain, 2009.

[CL] C. Chang and C. Lin. *LibSVM*.

[CP98] B. B. Chaudhuri and U. Pal. A complete printed bangla ocr system. *Pattern Recognition*, 31:531–549, 1998.

[CT01] R. Cao and C.L. Tan. Text/graphics separation in maps. In *Proceedings of International Workshop on Graphics Recognition (GREC)*, Canada, 2001.

[CW98] J. Y. Chiang and R.C. Wang. Seal identification using the Delaunay tessellation. In *Proceedings of National Science Council of ROC (A)*, volume 22, pages 751–757, 1998.

[CW00] Y. K. Chen and J. F. Wang. Segmentation of single- or multiple-touching handwritten numeral string using background and foreground analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1304–1317, 2000.

[DC06] D. B. Dhar and B. Chanda. Extraction and recognition of geographical features from paper maps. *International Journal on Document Analysis and Recognition*, pages 232–245, 2006.

[DH72] R.O. Duda and P. E. Hart. Use of the hough transform to detect lines and curves in pictures. *Communications of the Association for Computing Machinery*, 15:11–15, 1972.

[DIP92] G. Dimauro, S. Impedovo, and G. Pirlo. From character to cursive script recognition: Future trends in scientific research. In *Proceedings of International Conference on Pattern Recognition*, volume 2, page 516, 1992.

[DMS95a] M. Deseilligny, H. Le Men, and G. Stamon. Characters string recognition on maps, a method for high level reconstruction. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 249–252, 1995.

[DMS95b] M. P. Deseilligny, Hervé Le Men, and G. Stamon. Character string recognition on maps, a rotation-invariant recognition method. *Pattern Recognition Letters*, 16(12):1297–1310, 1995.

[Doe98] D. Doermann. The indexing and retrieval of document images: A survey. *Computer Vision and Image Understanding*, 70(3):287–298, 1998.

[DP73] D. Douglas and T. Peucker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *The Canadian Cartographer*, 10(2):112–122, 1973.

[DPV+08] M. Delalandre, T. Pridmore, E. Valveny, H. Locteau, and E. Trupin. Building synthetic graphical documents for performance evaluation. *Revised Selected Papers of Workshop on Graphics Recognition (GREC), Lecture Notes in Computer Science (LNCS)*, 5046:288–298, 2008.

[DVPK10] M. Delalandre, E. Valveny, T. Pridmore, and D. Karatzas. Generation of synthetic documents for performance evaluation of symbol recognition and spotting systems. *International Journal on Document Analysis and Recognition*, online first articles, May 2010.

[FB81] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[Fen91] R. Fenrich. Segmentation of automatically located handwritten words. In *2nd International Workshop on Frontiers on Handwriting Recognition*, pages 33–44, 1991.

[FK88] L. A. Fletcher and R. Kasturi. A robust algorithm for text string separation from mixed text/graphics images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(6):910–918, 1988.

[FNK92] H. Fujisawa, Y. Nakano, and K. Kurino. Segmentation methods for character recognition from segmentation to document structure analysis. In *Proc. IEEE*, volume 80, pages 1079–1092, 1992.

[Fri03] A. S. Frisch. The fuzzy integral for color seal segmentation on document images. In *International Conference on Image Processing*, volume 1, pages 157–160, 2003.

[Fri05] A. S. Frisch. Color seal extraction from documents: Robustness through soft data fusion. *Journal on Applied Signal Processing*, 13:2146–2152, 2005.

[FSN+95] M. Flickner, H. Sawhney, W. Niblack, J. Ashley, Q. Huang, B. Dom, M. Gorkani, J. Hafner, D. Lee, D. Petkovic, D. Steele, and P. Yanker. Query by image and video content: The qbic system. *Computer*, 28:23–32, 1995.

[GA99] H. Goto and H. Aso. Extracting curved lines using local linearity of the text line. *International Journal on Document Analysis and Recognition*, 2:111–118, 1999.

[GDC95] W. Gao, S. Dang, and X. Chen. A system for automatic chinese seal imprint verification. In *Proceedings of International Conference on Document Analysis and Recognition*, volume 2, pages 660–664, 1995.

[GPN07] B. Gatos, I. Pratikakis, and K. Ntirogiannis. Segmentation based recovery of arbitrarily warped document images. In *Proceedings of International Conference on Document Analysis and Recognition*, 2007.

[Gra72] R. Graham. An efficient algorithm for determining the convex hull of a finite point set. *Information Processing Letters*, 1:132–133, 1972.

[HAT96] S. He, N. Abe, and C. L. Tan. A clustering-based approach to the separation of text strings from mixed text/graphics documents. In *Proceedings of International Conference on Pattern Recognition*, pages 706–710, 1996.

[HHYY96] R. Haruki, T. Horiuchi, H. Yamada, and K. Yamamoto. Automatic seal verification using three-dimensional reference seals. In *Proceedings of 13th International Conference on Pattern Recognition*, volume 3, pages 199–203, 1996.

[HL94] F. Hones and J. Litcher. Layout extraction of mixed mode documents. In *IAPR Workshop on Machine Vision Applications*, pages 237–246, 1994.

[Hor01] T. Horiuchi. Automatic seal verification by evaluating positive cost. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 572–576, 2001.

[HS92] J. Hershberger and J. Snoeyink. Speeding up the douglas-peucker line-simplification algorithm. In *Proceedings of the 5th International Symposium on Spatial Data Handling*, volume 1, pages 134–143, Charleston, South Carolina, 1992.

[HSYS03] H. Hase, T. Shinokawa, M. Yoneda, and C. Y. Suen. Recognition of rotated characters by eigen-space. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 731–735, 2003.

[HT06] T. Hayashi and N. Takagi. A consideration on rotation invariant character recognition. In *World Automation Congress*, pages 1–6, 2006.

[HT10] T. V. Hoang and S. Tabbone. Text extraction from graphical document images using sparse representation. In *International Workshop on Document Analysis Systems*, pages 143–150, 2010.

[Hu62] M-K. Hu. Visual pattern recognition by moment invariants. *IRE Transactions on Information Theory*, IT-8:179–187, 1962.

[HYSS01] H. Hase, M. Yoneda, T. Shinokawa, and C. Y. Suen. Alignment of free layout color texts for character recognition. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 932–936, 2001.

[HYZ+95] Q. Hu, J. Y. Yang, Q. Zhang, K. Liu, and X. Shen. An automatic seal imprint verification approach. *Pattern Recognition*, 28:1251–1266, 1995.

[ITHK09] M. Iwamura, T. Tsuji, A. Horimatsu, and K. Kise. Real-time camera-based recognition of characters and pictograms. In *Proceedings of International Conference on Document Analysis and Recognition*, Spain, 2009.

[KH90a] A. Khotanzad and Y. H. Hong. Invariant image recognition by zernike moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:489–497, 1990.

[KH90b] A. Khotanzad and Y. H. Hong. Rotation invariant image recognition using feature selected via a systematic methods. *Pattern Recognition*, 23:1089–1101, 1990.

[Kit92] N. Kita. Object locating based on concentric circular description. In *Internation Conference on Pattern Recognition*, volume 1, pages 637–641, The Hague, 1992.

[KKS00] K. K. Kim, J. H. Kim, and C. Y. Suen. Recognition of unconstrained handwritten numeral strings by composite segmentation method. In *Proceedings of International Conference on Pattern Recognition*, pages 594–597, 2000.

[Kol03] A. Kolesnikov. *Efficient algorithms for vectorization and polygonal approximation*. PhD thesis, University of Joensuu, Finland, 2003.

[KPB87] S. Kahan, T. Pavlidis, and H.S. Baird. On the recognition of printed characters of any font and size. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9:274–288, 1987.

[KSM85] M. Karima, K. S. Sadhal, and T. O. McNeil. From paper drawings to computer aided design. *IEEE Computer Graphics Applications*, 5:24–39, 1985.

[KTYS94] F. Kimura, S. Tsuruoka, Y.Miyake, and M. Shridhar. A lexicon directed algorithm for recognition of unconstrained handwritten words. *ICICE Transactions on Information and Systems*, E77:785–793, 1994.

[LAD95] H. Luo, G. Agam, and I. Dinstein. Directional mathematical morphology approach for line thinning and extraction of character strings from maps and line drawings. In *Proceedings of International Conference on Document Analysis and Recognition*, volume 1, page 257, Washington, DC, USA, 1995. IEEE Computer Society.

[LBW06] S. Leyk, R. Boesch, and R. Weibel. Saliency and semantic processing: Extracting forest cover from historical topographic maps. *Pattern Recognition*, 39:953–968, 2006.

[LC02] P. K. Loo and C.L.Tan. Word and sentence extraction using irregular pyramid. In *Proceedings of IAPR Workshop on Document Analysis Systems*, pages 307–318, 2002.

[LCSS99] Z. Lu, Z. Chi, W. Siu, and P. Shi. A background-thinning based approach for separating and recognizing connected handwritten digit strings. *Pattern Recognition*, 32:921–933, 1999.

[LK89] S. W. Lee and J.H. Kim. Attributed stroke graph match-ing for seal imprint verification. *Pattern Recognition Letters*, 9:137–145, 1989.

[LKMS08] J. Llados, D. Karatzas, J. Mas, and G. Sanchez. A generic architecture for the conversion of document collections into semantically annotated digital archives. *Journal of Universal Computer Science*, 14:2912–2935, 2008.

[LLWZ07] H. Liu, Y. Lu, Q. Wu, and H. Zha. Automatic seal image retrieval method by using shape features of chinese characters. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pages 2871–2876, 2007.

[Low04] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[LS09] C. L. Liu and C. Y. Suen. A new benchmark on the recognition of handwritten bangla and farsi numeral characters. *Pattern Recognition*, 42:3287–3295, 2009.

[LSA94] S. Liang, M. Shridhar, and M. Ahmadi. Segmentation of touching characters in printed document recognition. *Pattern Recognition*, 27(6):825–840, 1994.

[LSC92] L. Lam, Lee S.W., and Suen C.Y. Thinning methodologies-a comprehensive survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(9):869–885, 1992.

[LT08] S. Lu and C. L. Tan. Retrieval of machine-printed latin documents through word shape coding. *Pattern Recognition*, 41(5):1816–1826, May 2008.

[Lu95] Y. Lu. Machine printed character segmentation - an overview. *Pattern Recognition*, 28:267–273, 1995.

[Lu98] Z. Lu. Detection of text regions from digital engineering drawings. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20:431–439, 1998.

[LVSM02] J. Llados, E. Valveny, G. Sanchez, and E. Marti. Symbol recognition: Current advances and perspectives. In *Graphics Recognition Algorithms and Applications*, 2002.

[LWT04] Y. Lu, Z. Wang, and C. L. Tan. Word grouping in document images based on voronoi tessellation. In *Proceedings of IAPR Workshop on Document Analysis Systems*, pages 147–157, 2004.

[LZDJ08] Y. Li, Y. Zheng, D. Doermann, and S. Jaeger. Script independent text line segmentation in freestyle handwritten documents. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(8):1313–1329, 2008.

[LZT07] L. Likforman, A. Zahour, and B. Taconet. Text line segmentation of historical documents: a survey. *International Journal on Document Analysis and Recognition*, pages 123–138, 2007.

[Man86] J. Mantas. An overview of character recognition methodologies. *Pattern Recognition*, 19(6):425–430, 1986.

[Mar10] N. K. Marcelo. *Dynamic Optimization of Classification Systems for Adaptive Incremental Learning.* PhD thesis, Ecole de technologie suprieure, Montreal, 2010.

[MHP07] M. Monwar, W. Haque, and P. P. Paul. A new approach for rotation invariant optical character recognition using eigen digit. In *Proceedings of Canadian Conference on Electrical and Computer Engineering*, pages 1317–1320, 2007.

[MM02] T. Matsuura and K. Mori. Rotation invariant seal imprint verification method. In *Proceedings of 9th International Conference of Electronics, Circuits & Systems, ICECS*, volume 3, pages 955–958, 2002.

[MMS03] S. Marinai, E. Marino, and G. Soda. Indexing and retrieval of words in old documents. In *Proceedings of International Conference on Document Analysis and Recognition*, page 223, Washington, DC, USA, 2003.

[MS01] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In *In Proceedings of the 8th International Conference on Computer Vision*, pages 525–531, 2001.

[MY04] T. Matsuura and K. Yamazakiv. Seal imprint verification with rotation invariance. *Circuits and Systems*, 1:597–600, 2004.

[Nag00] G. Nagy. Twenty years of document image analysis in pami. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(1):38–62, 2000.

[Nav01] G. Navarro. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88, 2001.

[NKI05] T. Nakai, K. Kise, and M. Iwamura. Camera-based document image retrieval as voting for partial signatures of projective invariants. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 379–383, Washington, DC, USA, 2005.

[NPR07] F. Nourbakhsh, P. B. Pati, and A. G. Ramakrishnan. Automatic seal information reader. In *International Conference on Computing: Theory and Applications*, pages 502–505, 2007.

[NSV92] G. Nagy, S. Seth, and M. Viswanathan. A prototype document image analysis system for technical journals. In *Computer*, volume 25, pages 10–22, 1992.

[O'G93] L. O'Gorman. The document spectrum for page layout analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:1162–1173, 1993.

[OLBS00] L. E. Oliveira, E. Lethelier, F. Bortolozzi, and R. Sabourin. A new segmentation approach for handwritten digits. In *Proceedings of International Conference on Pattern Recognition*, pages 2323–2326, 2000.

[Ots79] N. Otsu. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66, 1979.

[PBC03] U. Pal, A. Belaid, and C. Choisy. Touching numeral segmentation using water reservoir concept. *Pattern Recognition Letters*, pages 261–272, 2003.

[PKRP06] U. Pal, F. Kimura, K. Roy, and T. Pal. Recognition of english multi-oriented characters. In *Proceedings of International Conference on Pattern Recognition*, pages 873–876, 2006.

[PL92] S. J. Perantonis and P. J. G. Lisboa. Translation, rotation, and scale invariant pattern recognition by high-order neural networks and moment classifiers. *IEEE Transaction on Neural Networks*, 3:241–245, 1992.

[POYC01] H. C. Park, S. Y. Ok, Y. J. Yu, and H. G. Cho. Word extraction in text/graphic mixed image using 3-dimensional graph model. *International Journal on Document Analysis and Recognition*, 4:115–130, 2001.

[PR04] U. Pal and P. P. Roy. Multi-oriented and curved text lines extraction from indian documents. *IEEE Transactions on Systems, Man and Cybernetics - Part B*, pages 1676–1684, 2004.

[PSC03] U. Pal, S. Sinha, and B. B. Chaudhuri. English multi-oriented text line extraction. *Image Analysis, Springer Verlag, Lecture Notes on Computer Science (LNCS)*, 2749:1146–1153, 2003.

[RCB05] J. Ricard, D. Coeurjolly, and A. Baskurt. Generalizations of angular radial transform for 2d and 3d shape retrieval. *Pattern Recognition Letters*, 26:2174–2186, 2005.

[RL08] M. Rusinol and J. Llados. Word and symbol spotting using spatial organization of local descriptors. In *Proceedings of IAPR Workshop on Document Analysis Systems*, pages 489–496, 2008.

[RM05] T. Rath and R. Manmatha. Word spotting for historical documents. *International Journal on Document Analysis and Recognition*, 9:139–152, 2005.

[RPC04] K. Roy, U. Pal, and B. B. Chaudhuri. A system for joining and recognition of broken bangla numerals for indian postal automation. In *Indian Conference on Computer Vision, Graphics and Image Processing*, pages 641–646, 2004.

[RPLD09] P. P. Roy, U. Pal, J. Lladós, and M. Delalandre. Multi-oriented and multi-sized touching character segmentation using dynamic programming. In *Proceedings of International Conference on Document Analysis and Recognition*, Barcelona, Spain, 2009.

[RPLK08] P. P. Roy, U. Pal, J. Llados, and F Kimura. Convex hull based approach for multi-oriented character recognition from graphical documents. In *Proceedings of International Conference on Pattern Recognition*, Tampa, USA, 2008.

[RVL+07] P. P. Roy, E. Vazquez, J. Llados, R. Baldrich, and U. Pal. A system to segment text and symbols from color maps. *Revised Selected Papers of Workshop on Graphics Recognition (GREC), Lecture Notes in Computer Science (LNCS)*, pages 245–256, 2007.

[SB07] F. Shafait and T. M. Breuel. Document image dewarping contest. In *2nd international Workshop on Camera-Based Document Analysis and Recognition*, Curitiba, Brazil, September 2007.

[SLLC05] J. Song, Z. Li, M. R. Lyu, and S. Cai. Recognition of merged characters based on forepart prediction, necessity-sufficiency matching, and

character-adaptive masking. *IEEE Transactions on Systems, Man and Cybernetics - Part B*, 35:2–11, 2005.

[SLY+09] F. Su, T. Lu, R. Yang, S. Cai, and Y. Yang. A character segmentation method for engineering drawings based on holistic and contextual constraints. In *International Workshop on Graphics Recognition*, 2009.

[SMA00] S. Sato, S. Miyake, and H. Aso. Evaluation of two neocognitron-type models for recognition of rotated patterns. In *Proceedings of International Conference on Neural Information Processing*, pages 295–299, 2000.

[SRI99] T. Steinherz, E. Rivlin, and N. Intrator. Offline cursive script word recognition: A survey. *International Journal on Document Analysis and Recognition*, 2:90–110, 1999.

[SS95] N. W. Strathy and C. Y. Suen. A new system for reading handwritten zip codes. In *Proceedings of International Conference on Document Analysis and Recognition*, volume 1, pages 74–77, 1995.

[SSR10] T. Saba, G. Sulong, and A. Rehman. A survey on methods and strategies on touched characters segmentation. *International Journal of Research and Reviews in Computer Science*, 1:103–114, 2010.

[Tak01] A. Takasu. Document filtering for fast approximate string matching of errorneous text. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 916–920, 2001.

[TC02] D. M. Tsai and C. H. Chiang. Rotation-invariant pattern matching using wavelet decomposition. *Pattern Recognition Letters*, 23:191–201, 2002.

[TCS91] Y. Y. Tang, H. D. Cheng, and C. Y. Suen. Translation-ring-projection (trp) algorithm and its vlsi implementations. In *Character and Handwriting Recognition*, pages 25–56, Singapore, 1991.

[TJT96] O. D. Trier, A. K. Jain, and T. Taxt. Feature extraction methods for character recognition - a survey. *Pattern Recognition*, 29:641–662, 1996.

[TL03] K. Tombre and B. Lamiroy. Graphics recognition - from re-engineering to retrieval. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 148–155, 2003.

[TN98] C. L. Tan and P. O. Ng. Text extraction using pyramid. *Pattern Recognition*, 31(1):63–72, 1998.

[TTP+02] K. Tombre, S. Tabbone, L. Peissier, B. Lamiroy, and P. Dosch. Text /graphics separation revisited. In *Proceedings of IAPR Workshop on Document Analysis Systems*, pages 200–211, 2002.

[UIOK06] S. Uchida, M. Iwamura, S. Omachi, and K. Kise. Ocr fonts revisited for camera-based character recognition. In *Proceedings of International Conference on Pattern Recognition*, pages 1134–1137, 2006.

[UM05] K. Ueda and K. Matsuo. Automatic seal imprint verifiation system for bank check processing. In *International Conference on Information Technology and Applications*, volume 1, pages 768–771, 2005.

[UMM98] K. Ueda, T. Mutoh, and K. Matsuo. Automatic verification system for seal imprints on japanese banckchecks. In *Proceedings of International Conference on Pattern Recognition*, volume 1, pages 629–632, 1998.

[USI+07] S. Uchida, M. Sakai, M. Iwamura, S. Omachi, and K. Kise. Extraction of embedded class information from universal character pattern. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 437–441, 2007.

[Vap95] V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlang, 1995.

[VOB+08] E. Vellasques, L. S. Oliveira, A. S. Britto, Jr., A. L. Koerich, and R. Sabourin. Filtering segmentation cuts for digit string recognition. *Pattern Recognition*, 41(10):3044–3053, 2008.

[WCL94] S. S. Wang, P. C. Chen, and W. G. Lin. Invariant pattern recognition by moment fourier descriptor. *Pattern Recognition*, 27:1735–1742, 1994.

[WL06] J. Llados W. Liu. Graphics recognition. ten years review and future perspectives. *Revised Selected Papers of Workshop on Graphics Recognition (GREC), Lecture Notes in Computer Science (LNCS)*, 2006.

[WS94] D. Wang and S. N. Srihari. Analysis of form images. In *Document Image Analysis*, pages 1031–1051, 1994.

[WZH00] W. J. Williams, E. Zalubas, and A. O. Hero. Word spotting in bitmapped fax documents. *Information Retrieval*, 2:207–226, 2000.

[XK91] Q. Xie and A. Kobayashi. A construction of pattern recognition system invariant of translation, scale-change and rotation transformation of pattern. *Transactions of the Society of Instrument and Control Engineers*, pages 1167–1174, 1991.

[YW01] T. N. Yang and S. D. Wang. A rotation invariant printed chinese character recognition system source. *Pattern Recognition Letters*, 22:85–95, 2001.

[YY98] D. Yu and H. Yan. Separation of single-touching handwritten numeral strings based on structural features. *Pattern Recognition*, 31:1835–1847, 1998.

[YYH93] H. Yamada, K. Yamamoto, and K. Hosokawa. Directional mathematical morphology and reformalized hough transformation for the analysis of topographic maps. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(4):380–387, 1993.

[YYZ09] G. Yong, Z. Yan, and H. Zhao. Touching string segmentation using MRF. In *Proceedings of International Conference on Computational Intelligence and Security*, pages 520–524, 2009.

[ZD09] G. Zhu and D. Doermann. Logo matching for document image retrieval. In *Proceedings of International Conference on Document Analysis and Recognition*, pages 606–610, 2009.

[ZJD06] G. Zhu, S. Jaeger, and D. Doermann. A robust stamp detection framework on degraded documents. In *SPIE Conference on Document Recognition and Retrieval*, 2006.

[ZL04] D. Zhang and G. Lu. Review of shape representation and description techniques. *Pattern Recognition*, 37:1–19, 2004.

List of Publications

Journal Papers

1. U. Pal, P. P. Roy, N. Tripathy and J.Lladós. Multi-Oriented Bangla and Devnagari Text Recognition. *Pattern Recognition*, 2010 vol. 43, pages 4124-4136, 2010.

2. U. Pal and P. P. Roy. Multi-oriented and curved text lines extraction from Indian documents. *IEEE Transaction on Systems, Man and Cybernetics-Part B*, vol. 34, pages 1676-1684, 2004.

3. P. P. Roy, U. Pal and J. Lladós. Document Seal Detection using GHT and Character Proximity Graphs. *Pattern Recognition* (Minor Revision), 2010.

4. P. P. Roy, U. Pal, J. Lladós and Mathieu Delalandre. Touching Text Character Segmentation in Graphical Documents using Dynamic Programming. *International Journal of Document Analysis and Recognition* (Revision), 2010.

5. P. P. Roy, U. Pal and J. Lladós. Text Line Extraction in Graphical Documents using Background and Foreground Information. *International Journal of Document Analysis and Recognition* (Revision), 2010.

Book Chapters

1. P. P. Roy, E. Vazquez, J. Lladós, R. Baldrich and U. Pal. A system to segment text and symbols from color maps. *Revised Selected Papers of Workshop on Graphics Recognition (GREC), Lecture Notes in Computer Science (LNCS)*, pages 245-256, 2008.

2. P. P. Roy, U. Pal and J. Lladós. Touching Text Character Localization in Graphical Documents Using SIFT. *Revised Selected Papers of Workshop on Graphics Recognition (GREC), Lecture Notes in Computer Science (LNCS)*, pages 199-211, 2010.

Conference and Workshop Contributions

1. P. P. Roy, U. Pal and J. Lladós. Query Driven Word Retrieval in Graphical Documents. *In Proceedings of the 9th IAPR International Workshop on Document Analysis Systems (DAS)*, Boston, USA, pages 191-198, 2010.

2. P. P. Roy, U. Pal and J. Lladós. Seal Object Detection in Document Images using GHT of Local Component Shapes. *In Proceeding of ACM Symposium on Applied Computing (ACM - SAC)*, Sierre, Switzerland, 2009.

3. P. P. Roy, U. Pal, J. Lladós and M. Delalandre. Multi-Oriented and Multi-Sized Touching Character Segmentation using Dynamic Programming. *In Proceeding 10th International Conference on Document Analysis and Recognition (ICDAR)*, Barcelona, Spain, pages 11-15, 2009.

4. P. P. Roy, U. Pal and J. Lladós. Seal detection and recognition: An approach for document indexing. *In Proceeding 10th International Conference on Document Analysis and Recognition (ICDAR)*, Barcelona, Spain, pages 101-105, 2009.

5. P. P. Roy, U. Pal and J. Lladós. Touching Text Character Localization in Graphical Documents Using SIFT. *Workshop on Graphics Recognition (GREC)*, La Rochelle, France, pages 271-279, 2009.

6. P. P. Roy, J. Lladós and U. Pal. A Complete System for Detection and Recognition of Text in Graphical Documents using Background Information. *International Conference on Computer Vision Theory and Applications (VISAPP)*, Lisbon, Portugal, 2009.

7. P. P. Roy, U. Pal and J. Lladós. Recognition of Multi-Oriented Touching Characters in Graphical Documents. *6th Indian Conference on Computer Vision, Graphics & Image Processing (ICVGIP)*, Bhubaneswar, India, pages 297-304, 2008.

8. P. P. Roy, U. Pal, J. Lladós and F. Kimura. Convex Hull based Approach for Multi-Oriented Character Recognition from Graphical Documents. *International Conference on Pattern Recognition (ICPR)*, Tampa, USA, 2008.

9. P. P. Roy, U. Pal, J. Lladós and F. Kimura. Multi-Oriented English Text Line Extraction Using Background and Foreground Information. *Proceedings of the 8th IAPR International Workshop on Document Analysis Systems (DAS)*, Nara, Japan, pages 315-322, 2008.

10. P. P. Roy, U. Pal and J. Lladós. Morphology Based Handwritten Line Segmentation using Foreground and Background Information. *International Conference on Frontiers in Handwriting Recognition (ICFHR)*, Montreal, Canada, pages 241-246, 2008.

11. P. P. Roy, U. Pal and J. Lladós. Multi-Oriented Character Recognition from Graphical Documents. *2nd International Conference on Cognition and Recognition (ICCR)*, Mandya, India, pages 30-35, 2008.

12. J. Lladós, P. P. Roy, J. Rodríguez and G. Sánchez. Word Spotting in Archive Documents using Shape Contexts. *3rd Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA)*, Girona, Spain, pages 290-297, 2007.

13. P. P. Roy, E. Vazquez, J. Lladós, R. Baldrich and U. Pal. A System to Retrieve Text/Symbols from Color Maps using Connected Component and Skeleton Analysis. *Workshop on Graphics Recognition (GREC)*, Brazil, 2007.

14. P. P. Roy, J. Lladós and U. Pal. Text/Graphics separation in colour maps. *In Proceeding International Conference on Computing: Theory and Applications (ICCTA)*. Kolkata, India, pages 545-549, 2007.

15. U. Pal and P. P. Roy. Text Lines Extraction from Indian Documents. *In Proceeding 5th International Conference on Advances in Pattern Recognition (ICAPR)*. Kolkata, India, pages 275-289, 2003.

Internal Workshops

1. P. P. Roy, U. Pal and J. Lladós. Comparison of Seal Detection by Different Character Shape Features. *4th CVC Internal Workshop, Computer Vision: Progress of Research and Development*, Barcelona, Spain, pages 112-118, 2009.

2. P. P. Roy, U. Pal and J. Lladós. Detection and Recognition of Multi-Oriented TextSymbols in Graphical Documents. *3rd CVC Internal Workshop, Computer Vision: Progress of Research and Development*, Barcelona, Spain, pages 104-111, 2008.

3. P. P. Roy, J. Lladós and U. Pal. Printed Text Character Matching Using Discrete Circular Profile Signature in Graphical Documents. *2nd CVC Internal Workshop, Computer Vision: Progress of Research and Development*, Barcelona, Spain, pages 44-47, 2007.

4. P. P. Roy, J. Lladós and U. Pal. Text/Graphics Separation in Color Maps. *1st CVC Internal Workshop, Computer Vision: Progress of Research and Development*. Barcelona, Spain, pages 158-163, 2006.

Awards

Best student paper award of the 10th International Conference on Document Analysis and Recognition (ICDAR, 2009) based on the paper:

P. P. Roy, U. Pal, J. Lladós and M. Delalandre. Multi-Oriented and Multi-Sized Touching Character Segmentation using Dynamic Programming. *In Proceeding 10th International Conference on Document Analysis and Recognition (ICDAR)*, pages 11-15, 2009.

Final Acknowledgment

This work has been partially supported by the research Fellowship number 2007-FIO1032 (Fellowship from AGAUR) and Spanish projects CONSOLIDER-INGENIO 2010 (CSD2007-00018), TIN2008-04998 and TIN2009-14633-C03-03.