Logo detection, recognition and spotting in context by matching local visual features

Viet Phuong Le

Université La Rochelle

École Doctorale Sciences et Ingénierie pour l'Information

Doctoral Thesis

Logo Detection, Recognition and Spotting in Context by Matching Local Visual Features

Author:
Viet Phuong Le

Supervisor:
Prof. Jean-Marc Ogier

Co-Supervisor:
Assoc. Prof. Cao De Tran
Assoc. Prof. Muriel Visani
Dr. Nibal Nayef

A thesis submitted in fulfilment of the requirements for the degree of Doctor of Philosophy

in the

Laboratory Informatics, Image and Interaction
Department of Science and Technology

8 décembre 2015

PhD Thesis Committee:

Mr. Jean-Christophe LAPAYRE, Université de Franche-Comté, president
Mr. Jean-Yves RAMEL, Université de Tours, examiner
Mr. Bart LAMIROY, Université de Lorraine, examiner
Mr. Jean-Marc OGIER, Université de La Rochelle, member
Mr. Cao De TRAN, Université de Can Tho, Vietnam, member
Mrs. Muriel VISANI, Université de La Rochelle, member
Mrs. Nibal NAYEF, Université de La Rochelle, member

Abstract

The last decades have seen an explosion of the amount of digitized document libraries. In order to properly mine these documents, it is necessary to categorize as well as to retrieve them. In particular, logos are commonly used in documents, especially in business and administrative documents. It allows us to determine the source of the documents quickly and accurately, without any textual transcription and at a low cost. This brings about different interesting issues, such as logo detection, logo recognition, and logo spotting.

This thesis presents a logo spotting framework applied to spotting logo images on document images and focused on document categorization and document retrieval problems. The spotting method is formulated in terms of searching for matches between all interest points of query logo images and of document images. Interest points are extracted from images and are described under local feature vectors by local detectors and local descriptors. Experiments show that blob detectors and spectra descriptors are the most suitable description technique to represent logos.

We also present three key-point matching methods: simple key-point matching with nearest neighbor, matching by 2-nearest neighbor matching rule method and matching by two local descriptors at different matching stages. The last two matching methods are improvements of the first method. The combination with another local descriptor method should be preferred in applications where correct identification of the incoming documents is of more importance than the false alarms (documents which do not contain any logos). On the other hand, if we expect a faster method or intend to minimize the false alarms which may include misclassified documents, the method with 2-nearest neighbor matching rule should be considered.

In addition, using a density-based clustering method to group the matches in our proposed spotting framework can help not only segment the candidate logo region but also reject the incorrect matches as outliers. Moreover, to maximize the performance and to locate logos, an algorithm with two stages is proposed for geometric verification based on homography with RANSAC.

Since key-point-based approaches assume costly approaches, we have also invested to optimize our proposed framework. The problems of text/graphics separation are studied. We propose a method for segmenting text and non-text in document images based on a set of powerful connected component features. Then, Adaboosting with decision trees is used to classify the connected components. Our results show that the method is fast

and is efficiently able to discriminate text from non-text, including the text that appears within graphical zones.

To deal with the requirement of camera-based applications, we applied dimensionality reduction techniques to reduce the high dimensional vector of local descriptors and approximate nearest neighbor search algorithms to optimize our proposed framework. In addition, we have also conducted experiments for a document retrieval system on the text and non-text segmented documents and ANN algorithm. The results show that the computation time of the system decreases sharply by 56% while its accuracy decreases slightly by nearly 2.5%.

Overall, we have proposed an effective and efficient approach for solving the problem of logo spotting in document images. We have validated the performance of our approach on both standard and private databases, and have achieved significantly better performance than prior methods for logo spotting. We have designed our approach to be flexible for future improvements by us and by other researchers. We believe that our work could be considered as a step in the direction of solving the problem of complete analysis and understanding of document images.

Acknowledgements

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

I would like to express my deep and sincere gratitude to my supervisors Prof. Jean-Marc Ogier, Assoc. Prof. Cao De Tran, Assoc. Prof. Muriel Visani, and Dr. Nibal Nayef. Their understanding, encouragement and guidance have provided the 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 have learned a lot throughout the process. I would like to thank them again for all.

My special gratitude is to Project 165 of Vietnamese government for their financial support for my study in France. I also thank my colleagues in the Information Technology Center of An Giang Provincial Committee of the Party. They shoulder the work during my study period. Without all the assistance, I could hardly have had the opportunity for doctoral studies.

I could hardly finish this thesis without the greatest support from Mr. Kenneth Phillips for his concern and careful proofreading, from all my colleagues at L3i, University of La Rochelle for their help and sharing during my time there.

I would like to express my deepest gratefulness to my family and friends for their continuous support and encouragement. I can see their happiness and pride of my education achievements. They are parts of the very strong motivations for my education pursuit.

Most of all, I wholeheartedly thank my wife for her serious and careful reading my thesis, and for being always next to me, care about me, encourage and inspire me in good and bad times.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

A logo is a graphical mark used to identify a company, organization, product or brand. Logos are used to represent a company's name, a particular product or service and are used heavily in the marketing of products and services. Logos have become an integral part of a company's identity and a well-recognized logo can increase a company's goodwill.

A logo usually has a recognizable and distinctive graphic design, stylized name or unique symbol for identifying an organization. It is affixed, included, or printed on all advertising, buildings, communications, literature, products, stationery, vehicles, etc. Logo can be seen anywhere in the surrounding in our daily life, such as in the streets, supermarkets, on the products or services, on administrative documents, etc. Examples of different logos are shown in Figure 1.1.

FIGURE 1.1: Some figures illustrate that logos appear everywhere in our surrounding.

The last decades have seen an explosion of the amount of digitized document libraries. In order to properly index these documents, it is necessary to categorize as well as to retrieve them. Throughout the years, several document classification systems have been investigated based on OCR and analysis of text by natural language processing. However, OCR systems reach good performance only with typewritten and printed documents, and natural language processing depends greatly on the context. On the other hand, graphical objects in document images such as logos, stamps, photos, graphs and diagrams contain much important information. In particular, logos are commonly used in documents, especially in business and administrative documents. Figure 1.2 shows some examples of using logos in documents. It allows us to determine the source of the documents quickly and accurately, without any textual transcription and at a low cost. This brings about different interesting issues, such as logo detection, logo recognition, and logo spotting.

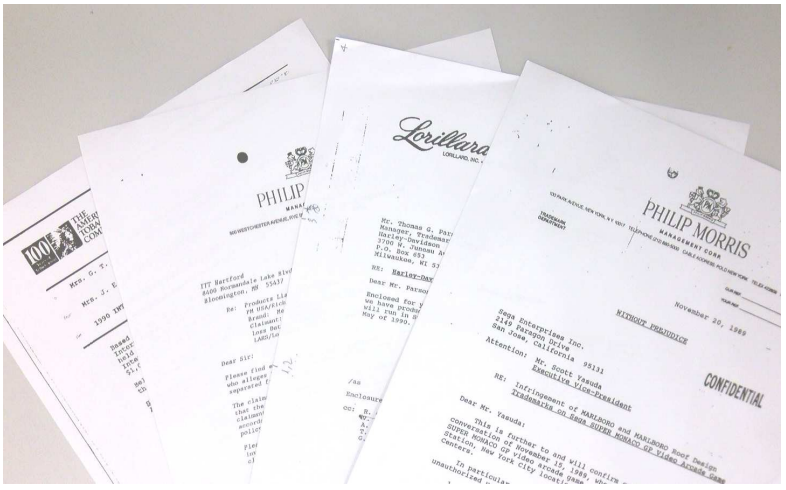
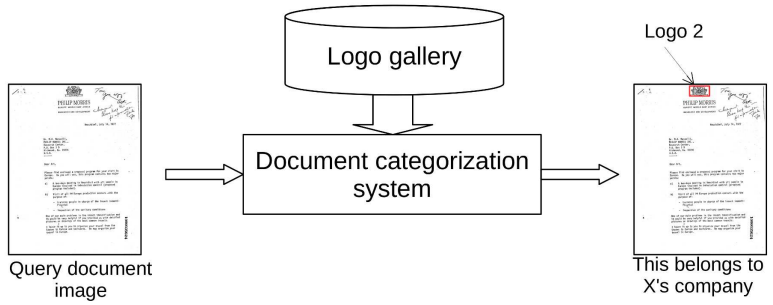


FIGURE 1.2: Logos are commonly used in documents, especially in business and administrative documents.

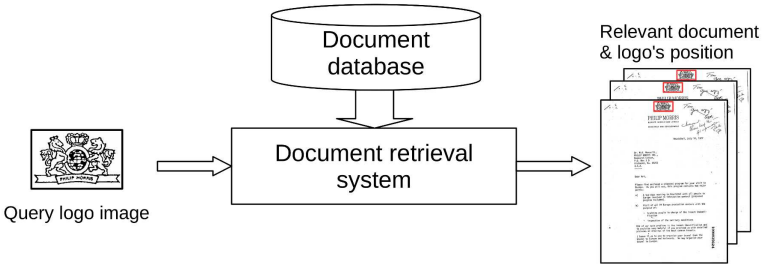
The logo spotting problem can be defined as the location of a set of regions of interest from a document image which are likely to contain an instance of a certain queried logo without explicitly recognizing it [4]. Whereas logo detection attaches special importance to detecting if logos occur on a document or not. Logo recognition focuses on recognizing an unknown logo image as one in a library (a set) of known logos. This usually refers to recognize pre-segmented (isolated) logos. However, it usually relates to previous segmentation. Meanwhile, logo spotting is able not only to solve logo detection and logo recognition problems but also can locate logos appearing on documents. In

addition, to avoid the paradox problem of the relationship between recognition and segmentation performance [1], logo spotting architectures should perform both recognition and segmentation simultaneously.

In the Document Image Analysis and Recognition (DIAR) field, two common applications related to document images are document categorization application and document retrieval application [6]. Document categorization application aims to categorize a query document based on given information. Hence, the given information is a set of logo images as a gallery, and the query document is then categorized following this logo gallery. Figure 1.3(a) shows the structure of document categorization application based on logo spotting. Similarly, Figure 1.3(b) shows the structure of document retrieval application. Therefore, a query logo image is the information to retrieve relevant documents in a set of documents based on logo spotting.



(a) Document categorization application



(b) Document retrieval application

FIGURE 1.3: Logo spotting applied to document categorization and retrieval applications.

Based on the discussion above, it can be seen that the main problem faced in the two common mentioned applications of DIAR is logo spotting. This has motivated the work to be presented throughout this thesis. More specifically, we have proposed a logo spotting approach and applied it in document categorization and retrieval problems. The proposed methods should be able to locate and recognize logos contained within

[1] This problem, presented by Sayre [5], concerned the relationship between recognition results and segmentation performance. In order to achieve good recognition results, the objects should be previously segmented; but to get a reliable segmentation, the objects should be previously recognized.

a database of complete document images. Moreover, the methods can work without a preliminary segmentation step.

1.1 Objectives and Contributions

The main objective of this work is to propose a framework for logo spotting in document images. The main spotting method within our framework is formulated in terms of a search for matches between all interest points of query logo images and of document images. Interest points are extracted from images and are described by local feature vectors which are computed by local detectors and local descriptors.

To this end, the problem will be tackled from different points of view and the mentioned objective can be detailed into the following points:

1. **Proposing methods for key-point matching**

   A method is proposed using key-point matching methods to find the best candidate match for each key-point from the image of the logo by identifying its nearest neighbor in the set of key-points from the document images.

   - *Simple key-point matching with nearest neighbor*: the best candidate match of a key-point is simply its nearest neighbor key-point using some distance function. The distance is computed in the description vector space with a suitable measure (e.g Euclidean, Hamming).
   - *Key-point matching with 2-nearest neighbor matching rule*: the false matches are removed based on the ratio of the first closest neighbor and the second closest neighbor.
   - *Key-point matching with post-filter by a second descriptor*: the removed matches which are rejected by the 2-nearest neighbor matching rule are re-examined using a second descriptor.

2. **Benchmarking well-known local feature detectors and descriptors**

   Although logo spotting has its own characteristics, the problem of locating logos in documents can be seen as a particular case of the object recognition problem from the Computer Vision field. Our objective is to test and to find the most suitable local feature detectors and descriptors for the problem of logo spotting.

3. **Applying the logo spotting approach on two applications: document categorization and retrieval**

The proposed framework with its main logo spotting method is applied to two applications: document categorization and retrieval with real document databases. A comparison is made between our proposed methods and state-of-the-art methods in the fields of the two mentioned applications.

4. **Separating text and non-text in document images**

   Document images are composed of text and non-text elements. Non-text can be halftones, drawings, mathematical formulas, tables, charts, logos, etc. Separating text and non-text in document images helps us to remove some superfluous elements as text in our context, and this could improve the performance of the methods. The main contribution of this part of the thesis is proposing a method to separate text and non-text in document images.

5. **Optimizing the proposed spotting framework**

   The objective point is centered on how to speed up the proposed framework. Therefore, a dimensionality reduction technique and approximate search computation are integrated into the framework. Moreover, separating text and non-text is also applied as the preprocessing step in the framework.

6. **Creating a database of scanned magazine image for logo spotting**

   Finally, beside testing our proposed spotting framework on a well-known database (Tobacco-800 database), we build a database with complex layout. The database is collected from scanned magazine images. We also create its ground-truth as bounding boxes of logo regions.

1.2 Thesis Organization

This thesis is organized in five chapters and two appendix, structured in two main parts.

Part I

The first part presents the main contribution of this thesis. A framework for spotting logos in document images using a key-point matching approach based on local feature description of logos is described.

- In Chapter 2, the state-of-the-art in logo spotting is reviewed. After a brief overview of the literature of information spotting as word spotting and symbol spotting, we divide this chapter into three sections, namely: the state-of-the-art

in logo description techniques; the approaches for logo detection, recognition, and spotting; and the local feature extraction.

- Chapter 3 presents a framework for spotting logos on document images by using key-point matching methods. First, the proposed framework for logo spotting is presented with four main steps: key-point extraction, key-point matching, key-point grouping, and geometric verification and localization. Logo and document representation is presented in Section 3.3. Section 3.4 describes three key-point matching methods: simple key-point matching method with nearest neighbor, key-point matching with 2-nearest neighbor rule, and key-point matching with post-filter by a second descriptor. We also discuss grouping the matching key-points issue, and geometric verification and localization in this chapter. Then, in the final part of this chapter, the effectiveness of this framework is demonstrated by experiments which compare the performances of different combinations of key-point detectors and descriptors.

- In Chapter 4, systems for document categorization and retrieval based on logo spotting are presented. We test our proposed framework on a set of real document images and logo images. In addition, an evaluation protocol for spotting is also discussed. The comparisons of performance of experiments on key-point matching methods and combinations of detectors and descriptors are presented in the first part of this chapter. This chapter also describes a proposed system for document categorization based on logo spotting while another system for document retrieval is presented in the following section. Both systems are built based on our proposed framework, and they are also tested on a large collection of real world documents using a well-known benchmark database of logos and show that our approach achieves better performances compared to state-of-the-art approaches.

Part II

The second part of this thesis, including Chapter 5 and Chapter 6, is devoted to the optimization of the logo spotting framework. Therefore, a text/graphics separation method is proposed to segment and then to reduce texts on documents. In addition, the dimensionality reduction problem and the approximate nearest neighbor problem are also discussed in this part.

- In Chapter 5, we first present an overview of the text/graphics separation approaches. Then, we focus on connected component extraction and a set of powerful connected component features is selected. In Section 5.4, we detail the proposed

method for separating text and non-text on documents using connected component features, discuss the evaluation protocol for our method on connected component-level, and pixel-level and describe the experiments. Finally, Section 5.5 is for the discussion and conclusion.

- Chapter 6 is centered on optimizing the logo spotting framework. We divide this chapter into two parts. First, the proposed framework is sped up using dimensionality reduction and approximate nearest neighbor algorithms to deal with a camera-based application. The second part presents the performance of the proposed framework using optimization. The text elements of documents are detected and removed using the proposed method in Chapter 5. Then, the separated documents are used to perform the document retrieval system.

Finally, in Chapter 7, we give a summary and concluding remarks about our work. We also discuss some possible future perspectives for logo spotting.

Databases used to perform the experiments are explained in Appendix A. For each database, we detail some of its characteristics including the number of elements, resolution, the number of classes. Some illustrations of each database are also presented.

Part I

Feature Matching-based Logo Spotting

Chapter 2

State-of-the-art in Logo spotting

2.1 Introduction

In this chapter, we will consider the architecture of logo spotting system in which the description issues and their suitable approaches will be discussed. The chapter is organized in six sections. First, we review the recent works of word spotting and symbol spotting in Section 2.2. Logo description categorization, and logo detection, recognition and spotting approaches will be discussed in Section 2.3 and Section 2.4. In Section 2.5, we briefly review popular local feature detectors and descriptors. We will finally give our discussion and summaries in Section 2.6 for our choices of the suitable approaches for spotting logo.

2.2 Information Spotting

Information spotting can be defined as the task of locating and retrieving specific information from databases without explicitly recognizing it. In the Document Image Analysis and Recognition field, that means that a spotting approach does not need to recognize all the objects on a document in the database but it has just to locate interest regions where the query object is likely to be found. For example, locating words in textual document images, identifying regions which likely to contain a symbol within documents, or locating logo regions on document images are different to applications to which researchers have devoted their efforts in the last years. In this section, we briefly review the existing work on word and symbol spotting and the rest of the chapter will focus on logo spotting.

2.2.1 Word Spotting

Besides quite an amount of word recognition approaches using OCR, research on word spotting techniques are also common in recent years. Rath and Manmatha [7, 8] presented a method for spotting historical handwritten words. The normalized projection profiles of segmented words are used as word signatures. Then, these signatures are seen as time series and are aligned using the Dynamic Time Warping (DTW) distance.

In another work, Lu and Tan [9] proposed a word shape coding based on character extremum points and horizontal cuts. Words are represented by simple digit sequences. Based on the frequency of the codes, the author defined several similar measures to retrieve documents written in the same language or on similar topics.

Llados and Sanchez [10] proposed a keyword spotting method based on the shape context descriptor in a voting strategy. First, words are represented by a signature formulated in terms of the shape context descriptor. They are then encoded as bit vectors codewords. Llados and Sanchez present a voting strategy to perform the retrieval of the zones of a given document containing a certain keyword.

Kise et al. [11] considered the word spotting problem. Since they focused their approach on Japanese documents, they found that a word can be formed by several Kanji characters. A query word within a document is then located by analyzing the characters density distribution within a document image.

2.2.2 Symbol Spotting

Symbol spotting refers to the problem of locating instances of a symbol within a line drawing. In other words, the task is to spot all the regions within the line drawing where a given query symbol is likely to be found [12–14].

In [15, 16], Rusinol et al. proposed different spotting methods using closed regions to detect interest regions for symbol spotting. In [15], each region is extracted and described by a two-center bipolar coordinate system. Then, a hashing table is used in an indexing scheme. In the other work of Rusinol [16], the closed regions are extracted based on a connected component analysis and described by attributed string. Similar strings are grouped in a lookup table with indexing keys based on the set median strings. A string matching technique is used to compare two strings representing a closed region contour. Finally, a Hough-like voting scheme is used to verify the found matches.

Graph-based representations as in [17–20] are often used to describe the graphical symbols. Llados and Sanchez [19] presented a combination of graph matching and graph

parsing. A graph indexing mechanism is presented for symbol spotting in the drawings. In Locteau et al. [20], the geometric relations of primitives are presented by a graph and then an attributed edit distance technique is used to match a query to the regions. Qureshi et al. [17, 18] proposed a method for symbol spotting using a graph representation of graphical documents. A graph description and matching method are used to match the query to the regions. Coustaty et al. [21] use an adapted Hough transform to extract the segments of a symbol, and arrange them in a topological graph as a structural signature, and then Galois Lattice classifier is used for recognition

From a different view point, some researchers proposed techniques based on segmentation, as in the works of Tabbone et al. [22–24]. The author used a text-graphics segmentation method with user corrections to segment symbols in simple line drawings. In [23, 24], the symbols which are linked to another part of document are segmented by analyzing the junction points of the skeleton image by a loop extraction process.

2.3 Logo Description

Generally speaking, the problem of logo detection, recognition and spotting belongs to the problem of object recognition as well as symbol spotting that has been studied by many researchers in recent decades. In their works, logos are described based on different primitives and visual cues. In [4], Rusinol presented three groups of symbol description approaches. However, based on the description techniques in the logo spotting literature, we group the logo description techniques in two groups based on the different visual cues which the methods aim to encode. The first cluster named *Point-based descriptions* describes logos based on the intensity of pixels. The second cluster is *geometric descriptors*. It contains the methods which analyze the shape of the logo.

Point-based Descriptions

The main interest of a point-based descriptor is to use a set of interest points to describe the logo image. Interest points are detected and described by interest point detectors and descriptors. This kind of description can work well with complex objects such as logos. Bagdanov et al. [25] proposed a system for detection and retrieval of trademarks appearing in sports videos. Each trademark is represented by a bag of SIFT feature points. SIFT, presented by Lowe [26], is a widely used descriptor that is created based on histogram of gradient orientation of pixels around an interest point (see more information in Section 2.5). Bagdanov et al. use the matching method to match each key-point of query logo image based on the ratio of the first and second nearest neighbors. Finally, the logo is localized based on a clustering of matched feature points in the video frame.

From another point of view, there is a family of point-based descriptors in which the logo is represented by set of visual words. The bag-of-words model is translated to the visual domain by the use of local descriptors over interest points. Rusinol et al. [27, 28] proposed a method for document categorization by logo spotting. The graphical logo and the query documents are described by a set of local features and matched using a bag-of-words model and then the author also uses the ratio of the first and second nearest neighbor for matching. In order to filter the matching key-points and consider only the key-points belonging to the logo in the query document, they considered clusters of key-points. Revaud et al. [29] used a set of key-points to describe a logo image. A bag-of-words-based approach coupled with learned weights that down-weight visual words that appear across different classes is used. Then, they proposed to learn a statistical model for the distribution of incorrect detections output by an image matching algorithm.

Another family of point-based descriptors are the ones based on spatial structure within the interest points. As presented by Sahbi et al. [30], the spatial structure of interest points is computed by using shape context, and the author defined the context by the local spatial configuration of SIFT interest points. A similar measure is also proposed to match these key-point points of query image to document image.

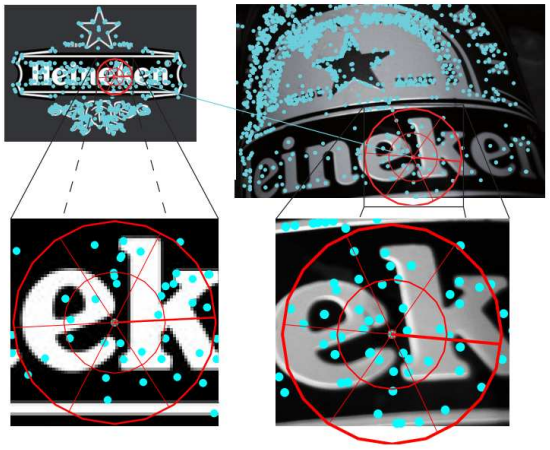


FIGURE 2.1: This figure shows an example of shape context based on the SIFT interest points and a similar configuration between two logo with the same class (extracted from [30])

Shape-based Descriptions

As the name implies, these techniques use shape information to describe logos. Logos are dissected in lower level graphical primitives and are then described by these primitives. These primitives usually include pixels, line segments, contours, connected components, skeleton.

Pham et al. [31] presented a method for logo detection exploiting contour based features. Firstly, outer contours are detected by a contour detection method and are grouped together by a line segmentation method to create strings of contours. Features of each string of contours are extracted at coarse and finer levels. Coarse-level features give graphical and domain information such as logo positions and aspect ratios. Finer-level features describe the contour regions using a gradient based representation. Finally, the authors used a learning-based method to score the strings of contours and used some rules to localize and correct the logo.

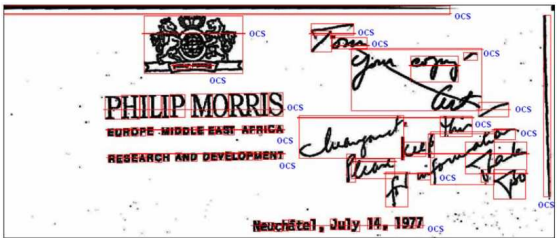


FIGURE 2.2: The outer contour strings detected for the image (extracted from [31]).

The use of connected components for logo detection is found in several works. Seiden et al. [32] focused on logo detection. In this approach, the document page was first segmented using the top-down X-Y cut algorithm [33]. A set of sixteen features of the connected components in each segment was extracted. Finally, a rule-based classification scheme was used to discriminate the segments. In another work, Zhu et al. [34] presented an approach to detect and extract logos in document images. First, a Fisher classifier at a coarse image scale on each connected component was applied to detect logo candidate regions. Then, an identified logo candidate region was successively classified at finer image scales by a cascade of simple classifiers.

Another related method is the work of Wang et al. [35]. They proposed a method based on the boundary extension of feature rectangles to detect logo candidate regions. Some rules using a simple decision tree classifier are applied on features extracted from the logo location, size, and aspect ratio.

2.4 Logo Detection, Recognition, and Spotting Approaches

Different techniques in logo recognition as well as in logo detection and spotting have been employed and developed throughout research. Each logo description technique brings about its suitable approaches. Based on the techniques used in the literature, we can classify the logo detection, recognition and spotting approaches into three clusters: key-point-based approaches and index-based approaches and learning-based approaches.

The main idea of the **key-point-based approaches** is to find pairs of key-points between sets of key-points of two images: a query image and a document image. Therefore, these approaches usually combine with point-based description techniques. For each key-point of the query image, these approaches find its best candidate match in the document image. Lowe [26] proposed an approach based on the nearest neighbor technique. The ratio of the closest neighbor and the second-closest neighbor was examined to match between the two key-points. Bagdanov et al. [25] used this approach for detection and retrieval of trademarks appearing in sports videos while Rusinol et al. [27, 28] and Revaud et al. [29] applied them to find the matches on a bag-of-words model.

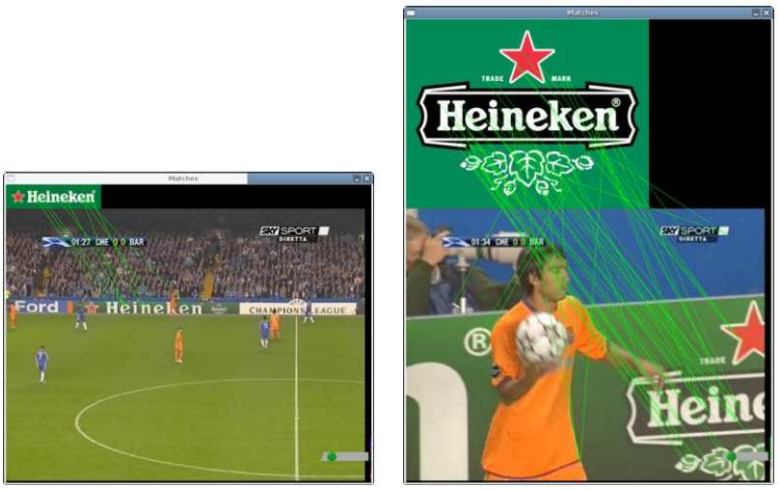


FIGURE 2.3: Two examples of the key-point matching technique using SIFT feature (extracted from [25])

Index-based approaches use a hash table to store the information in buckets or slots and access to the hashing table using the index keys. Jain and Doermann [36] used SURF feature to create a hashing table. Some positions of SURF feature with the lowest and highest distinctiveness score were used to create index keys. Each element of the hashing table contained the information of SURF feature such as X-Y coordinates, orientation and the feature vector is reduced from 64-dimension to 8-dimension of SURF feature vector using PCA. Then, the authors used a method to filter results using the orientation of local features and geometric constraints.

Learning-based approaches use machine learning techniques for logo recognition and/or spotting. Pham et al. [31] used Gentle Boost [37] combining decision trees to classify each string of contours. Francesconi et al. [38] proposed a logo recognition method by using a recursive neural network [39] based on contour-trees. First, logo images were described by a structured representation based on contour trees. Then these structured representations were classified thanks to the recursive neural network using features of nodes on contour trees such as means of the perimeter, surrounding area and

a synthetic representation of the curvature plot. A cascade of simple classifiers was used in the work of Zhu et al. [34]. The authors deployed Fisher classifier on each connected component at different scales using geometric features.

2.5 Local Feature Extraction

In Computer Vision, many algorithms rely on locating local interest points (or key-points) in each image, and calculating a feature description from the pixel region surrounding the interest point. Groups of interest points and descriptors together describe the actual objects within images. The algorithms used to find the interest points are referred to as *detectors*, and the algorithms used to describe the features are called *descriptors*.

2.5.1 Key-point Detectors

2.5.1.1 Corner detectors

Normally, a corner is defined as the intersection of two edges. However, in computer vision, a corner can also be defined as a point for which there are two dominant and different edge directions in a local neighborhood of the point as in Harris corner detector [40]. Another definition of a corner is that it is defined as a point which has different intensity with a set of pixels in a circular pattern as in FAST detector [41]. A wide range of corner detector methods have been proposed by many researchers. In this section, we discuss some popular corner detectors.

Harris corner detector, proposed by Harris and Stephens [40], is based on the second moment matrix (the auto-correlation matrix). "Corner" is defined where the intensity changes significantly in at least 2 directions. Consider the second moment matrix:

$$E(\Delta x, \Delta y) = \sum_{x,y} w(x,y)[I(x+\Delta x, y+\Delta y) - I(x,y)]^2 = \begin{bmatrix} \Delta x & \Delta y \end{bmatrix} M \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \quad (2.1)$$

M is the "second moment matrix"

$$M = \sum_{x,y} \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \quad (2.2)$$

Since M is symmetric, we can rewrite M:

$$M = R^T \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R \tag{2.3}$$

where λ_1 and λ_2 are the eigenvalues of M.

In practice, Harris proposed a measure which combines the two eigenvalues. If both λ_1 and λ_2 are small ($\lambda_1 \approx \lambda_2 \approx 0$), this point is a "flat". If one is high and one is low ($\lambda_1 \gg \lambda_2$ or $\lambda_1 \ll \lambda_2$), this point is an "edge". If both are high, this point is a "corner".

oFAST (Oriented FAST) detector, presented by Rublee et al [42], is a modified version of FAST detector [41]. FAST stands for Features from Accelerated Segment Test. FAST relies on an intensity threshold and a connected set of pixels in a circular pattern to determine a corner. FAST is known to be efficient to compute and fast to match; accuracy is also quite good. However, FAST does not produce multi-scale features. oFAST employed a scale pyramid of the image, and produces FAST features (filtered by Harris) at each level in the pyramid. For orientation invariant, they used the intensity centroid to compute the orientation of the key-point.

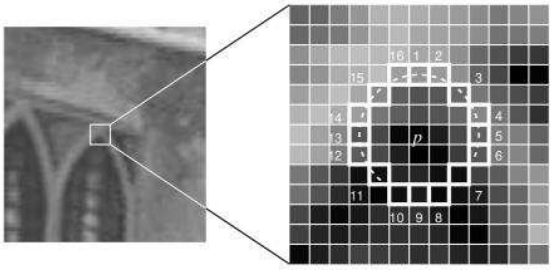


FIGURE 2.4: An example of a connected set of pixels in a circular pattern (extracted from [41]).

BRISK detector, Binary Robust Invariant Scalable Keypoints, is proposed by Leutenegger et al. [43]. To localize the key-point, it used the AGAST corner detector [44] which improves FAST detector. BRISK provides scale invariance because it detects key-points in a scale space pyramid, performing non-maximal suppression and interpolation across all scales.

2.5.1.2 Blob detectors

Difference-of-Gaussian (DoG), Lowe [45] proposed a method to efficiently detect stable key-point locations in scale space using scale-space extrema in the difference-of-Gaussian

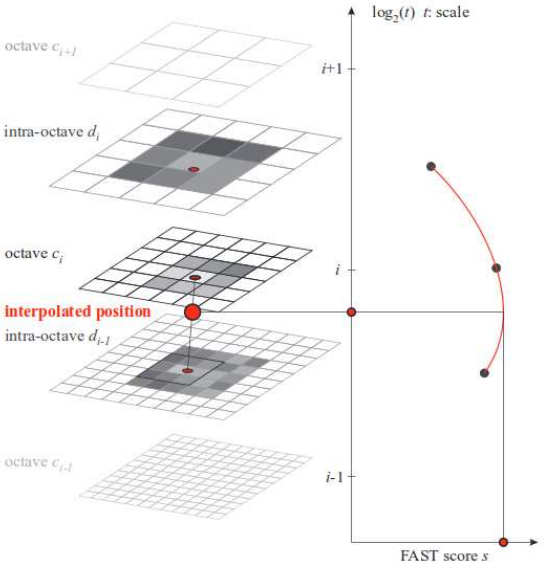


FIGURE 2.5: Scale-space interest point detection of BRISK detector (extracted from [43]).

function convolved with the image $D(x, y, \sigma)$. The Gaussian-smoothed image $L(x, y, \sigma)$ at the scale σ is computed as follows:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{2.4}$$

where $*$ is the convolution operation, $I(x, y)$ is the original image and $G(x, y, \sigma)$ is the 2D Gaussian kernel at scale σ:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.5}$$

The difference-of-Gaussian function $D(x, y, \sigma)$ between two nearly scales σ and $k\sigma$ is:

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \tag{2.6}$$

Figure 2.6(a) shows the scale-space structure of difference-of-Gaussian $D(x, y, \sigma)$.

In the next step, a point is selected if it is larger or smaller than all its neighbors. The neighbors of the point are the eight neighbors at the current scale and the nine neighbors at the scale above and below. Figure 2.6(b) shows 26 neighbors of a point to compute the maxima (or minima) of the difference-of-Gaussian image. Finally, the Taylor series expansion of scale space is used to get a more accurate location of extrema. Extrema whose intensity is less than a threshold are removed or which are like edges are also rejected.

The orientation of a key-point is computed based on the gradient magnitude and orientation of neighbors around the key-point. An orientation histogram with 36 bins covering 360 degrees of orientations of these neighbors is used to estimate the orientation of key-points. The highest peak or any peak above 80% is considered to compute the orientation. This is the reason why there are more than one key-point at the same points in some cases.

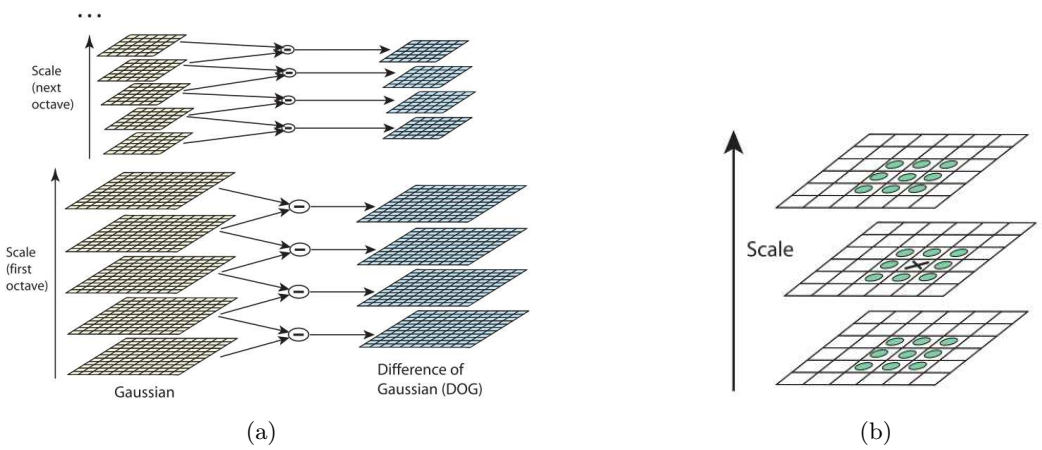


FIGURE 2.6: (a) The scale-space structure of difference-of-Gaussian and (b) 26 neighbors of a point to compute the maxima (or minima) of the difference-of-Gaussian image. (extracted from [26]).

Fast-Hessian detector, presented by Bay et al. [46], is a detector based on the Hessian matrix. Different from the Hessian-Laplace detector [47], Fast-Hessian detector relies on determinant of Hessian matrix for both scale and location. The Hessian matrix $\mathcal{H}(p,\sigma)$ in a point $p(x,y)$ in an image I at scale σ is defined as follows

$$\mathcal{H}(p,\sigma) = \begin{bmatrix} L_{xx}(p,\sigma) & L_{xy}(p,\sigma) \\ L_{xy}(p,\sigma) & L_{yy}(p,\sigma) \end{bmatrix} \tag{2.7}$$

where $L_{xx}(p,\sigma)$, $L_{yy}(p,\sigma)$ and $L_{xy}(p,\sigma)$ are the convolution of the Gaussian second order derivatives:

$$\begin{aligned} L_{xx}(p,\sigma) &= I(p) * \frac{\partial^2}{\partial x^2} g(\sigma) \\ L_{yy}(p,\sigma) &= I(p) * \frac{\partial^2}{\partial y^2} g(\sigma) \\ L_{xy}(p,\sigma) &= I(p) * \frac{\partial^2}{\partial xy} g(\sigma) \end{aligned} \tag{2.8}$$

The main idea of SURF detector is non-maximal-suppression of the determinants of the Hessian matrices. Different from SIFT which uses the approximated Laplacian of Gaussian (LoG) with DoG function, SURF uses an approximation LoG with box filter

kernel. Figure 2.7 shows examples of approximations for Gaussian second order derivatives with box filter kernel. Using convolution with box filter kernel can be computed faster thanks to integral images. Like SIFT, SURF also detects across scale-space to localize a key-point. However, SIFT downscales the image while SURF is calculated with larger and larger filters.

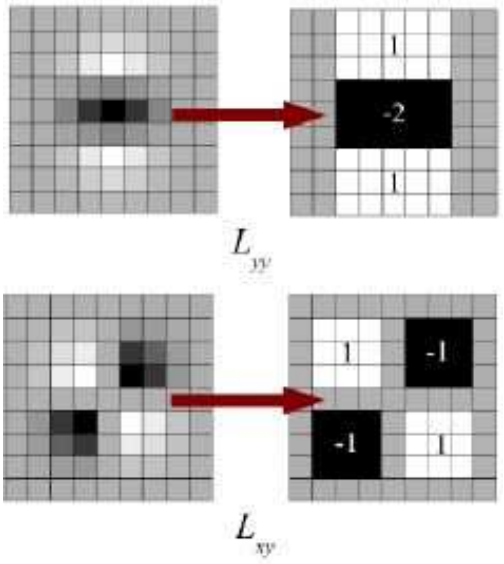


FIGURE 2.7: Examples of approximations for Gaussian second order derivatives with box filter (extracted from [46])

Maximally Stable External Regions (MSER) detector is presented by Matas et al. [48] and is based on the idea of taking stable regions through a wide range of thresholds. A binary image B_t of an image I at threshold level t is defined as follows:

$$B_t(x) = \begin{cases} 1 & \text{if } T(x) \geq t \\ 0 & \text{otherwise} \end{cases} \tag{2.9}$$

In other words, MSER detects maximally stable extremal regions which are connected components in B_t with little size change across several thresholds. The connected components are computed at each threshold level. The threshold value is increased from 0 to the maximum pixel value. During the threshold increment, *white* spots will eventually merge until the whole image is *white*. The white connected components are detected and the black connected components are obtained by inverting the intensity image. Connected components that do not grow or shrink or change as the intensity varies are considered maximally stable, and the MSER descriptor records the position of the maximal regions and the corresponding thresholds. The number of thresholds for which the connected component is stable is called the *margin*.

2.5.2 Key-point Descriptors

2.5.2.1 Binary descriptors

The main characteristic of the binary descriptor family is using the difference between image pixel point-pairs to provide binary values in a vector. Local binary descriptors are efficient to compute, efficient to store, and efficient to match using Hamming distance. Based on this characteristic, each binary descriptor presents its local sampling patterns to set the pairwise point comparisons. We start this subsection on local binary descriptors by analyzing the BRIEF descriptor and a modified version of BRIEF, since the BRIEF is well-known as a simple binary descriptor.

BRIEF [49] is a binary descriptor describing the difference of intensities between pairs of pixels u and v around the interest key-points. Given a patch p of size $S \times S$, a binary test τ on p is defined by:

$$\tau(p;u;v) = \begin{cases} 1 & \text{if } p(u) < p(v) \\ 0 & \text{if } p(u) \geq p(v) \end{cases} \tag{2.10}$$

where $p(u)$ and $p(v)$ are the pixel intensity in a smoothed version of p at point u and point v, respectively. The BRIEF feature is defined as a binary string of n binary tests on patch p, for each key-point:

$$f_n(p) = \sum_{1 \leq i \leq n} 2^{i-1} \tau(p;u_i;v_i) \tag{2.11}$$

where $\tau(p;u_i;v_i)$ is the binary test of the i-th pair on patch p. Notice that the value of each point is calculated via an integral image method to smooth a $n \times n$ region into the point value. Figure 2.8 shows five pairs sampled in a patch p.

An oriented version of BRIEF (called rBRIEF) is presented by Rublee et al. [42]. In their approach, an efficient method was used to steer BRIEF according to the orientation of key-points. And then, an improved training method was applied to find uncorrelated points in the training set with high variance, and the best 256 points were selected to define the pairwise sampling patterns used to create the binary feature vector.

BRISK descriptor [43] also used a point-pair sampling shape. Different from BRIEF, BRISK used a symmetric and circular patch with 60 total points arranged in four concentric rings as shown in Figure 2.9. In BRISK, the point-pairs are specified in two groups: long-distance and short-distance subsets. The long-distance subset is used together with the region gradients to determine angle and direction of the descriptor while

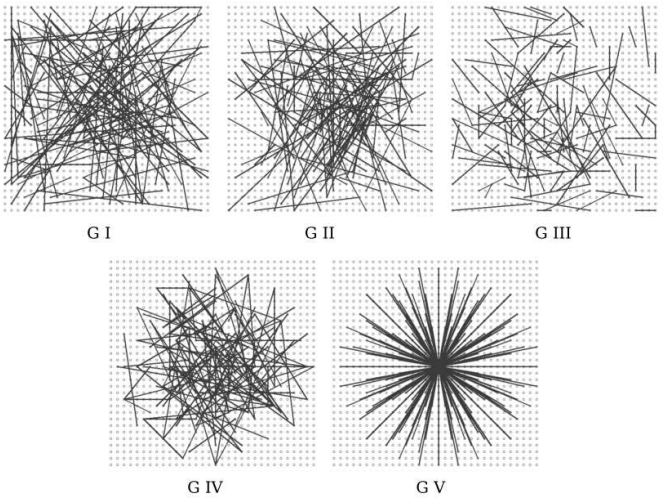


FIGURE 2.8: Five examples of the patch p of size $S \times S$ for BRIEF descriptor, each line represents a pair of pixels of p (extracted from [49])

the short-distance subset provides a 512-bit binary descriptor vector based on comparing pairs of points to form the descriptor.

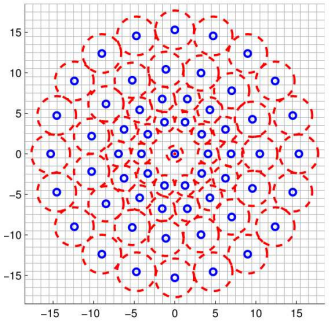


FIGURE 2.9: The BRISK sampling pattern with 60 points (extracted from [43])

FREAK descriptor [50] is also a binary descriptor. The technique adopted by FREAK is closely related to BRISK. FREAK improves the sampling pattern and method of pair selection of BRISK. FREAK uses a geometric pattern that mimics how the human retina works with 45 point pairs at locations around the key-point (see Figure 2.10). The actual pattern shape and point pairing are designed during a training phase where the best point pair patterns are learned using a method similar to ORB [42] to find point pairs with high variance.

2.5.2.2 Spectra descriptors

Different from the local binary descriptor group, the spectra descriptor group requires more intense computations and algorithms, often calculates on floating point numbers,

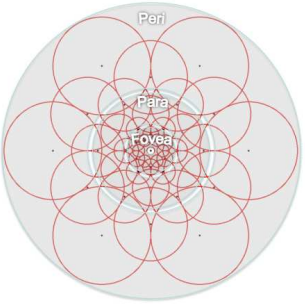


FIGURE 2.10: The FREAK sampling pattern (extracted from [50])

and may need larger memory. In this context, spectra can be histograms of light intensities, local area gradients, etc. such as histograms of local gradient orientations. This part presents two descriptors based on local gradient information and some of their modified versions.

Scale-Invariant Feature Transforms (SIFT), proposed by Lowe [26], is a feature descriptor representing the characteristics of the region around a key-point. SIFT descriptor is calculated based on the gradient distribution of the region. Firstly, the Gaussian-smoothed image $L(x, y, \sigma)$ at the scale σ where the key-point is detected is applied (see Equation 2.4 and 2.5).

A gradient magnitude $m(x, y)$ and an orientation $\theta(x, y)$ of each point $L(x, y)$ in the region is computed:

$$m(x,y) = \sqrt{[L(x+1,y) - L(x-1,y)]^2 + [L(x,y+1) - L(x,y-1)]^2} \tag{2.12}$$

$$\theta(x,y) = \tan^{-1}[L(x,y+1) - L(x,y-1)]/[L(x+1,y) - L(x-1,y)] \tag{2.13}$$

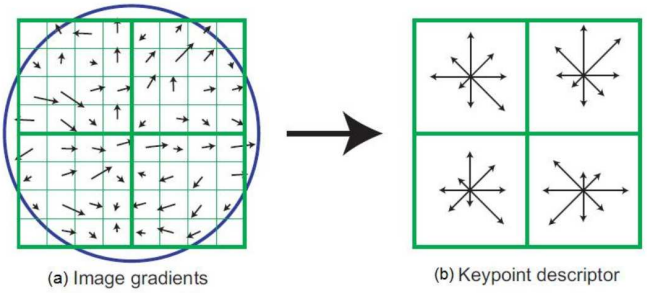


FIGURE 2.11: Example the computation of a SIFT descriptor in a region of size 8×8 (extracted from [26])

The gradient magnitude of the points around the key-point are weighted by a Gaussian window, indicated by the circle in Figure 2.11a, with a σ of 1.5 times the scale of the key-point. The feature descriptor of each key-point is concatenated from orientation

histograms in the region around the key-point such that each histogram is computed from points of a 4×4 sub-region. Each orientation histogram created on a 4×4 sub-region has 8 bins corresponding to 8 directions, as shown in Figure 2.11b. The length of each arrow (each bin) corresponds to the sum of the gradient magnitudes of the sample points within this 4×4 sub-region having a similar orientation to this arrow. An example in Figure 2.11 illustrates the computation of the SIFT descriptor concatenated from $2 \times 2 = 4$ histograms computed from a 8×8 region of sample points. In his work Lowe [26] used a region of size 16×16 for calculating the 4×4 orientation histograms; hence 128 elements of the feature descriptor.

SIFT is reported as a robust descriptor in brightness, contrast, rotation, scale and affine transforms. Therefore, in the last decade years, many works have been conducted to improve it. Ke and Sukthankar [51] have developed a descriptor similar to the SIFT descriptor called PCA-SIFT. It applies Principal Components Analysis (PCA) [52] to the normalized image gradient patch and performs better than the SIFT descriptor on artificially generated data [53]. Gradient Location and Orientation Histogram (GLOH), presented by Mikolajczyk et al. [53], is also an extension of the SIFT descriptor. They used polar coordinates and radially distributed bins to provide a 272-bin histogram. Then, PCA is used to reduce the size of the descriptor. On the other side, as the original SIFT descriptor works only on the gray channel of the image, other works have proposed descriptors which work on the color channel based on SIFT descriptor such as RGB-SIFT [54], HSV-SIFT [55], Hue-SIFT [56], OpponentDIST [54], rgSIFT [54], CSIFT [57].

Speeded Up Robust Features (SURF), introduced by Bay et al. [46], has two parts: orientation assignment and feature description. To estimate the orientation for the key-point, the Haar-wavelet response in two directions (horizontal and vertical) is applied in a circular neighborhood of radius 6s, with s the scale where the key-point is detected (see Figure 2.12). A sliding orientation window (60 degree) is used to find the dominant orientation by calculating the sum of all responses.

For feature description, SURF used a rectangular grid of 4×4 regions around the key-point to create the descriptor vector. Similar to SIFT, each region is also divided into 4×4 sub-regions. Then, the Haar-wavelet response is used to calculate each sub-region. Like SIFT, SURF uses a circularly symmetric Gaussian weighting factor for each Haar response. Finally, the descriptor vector contains four parts: $v = (\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|)$ in which each part is the sum of the wavelet response and the total descriptor vector length is 64. SURF is reported as a fast and good performance detector and descriptor [46]. A few modified versions of the SURF descriptor are discussed in [58, 59]

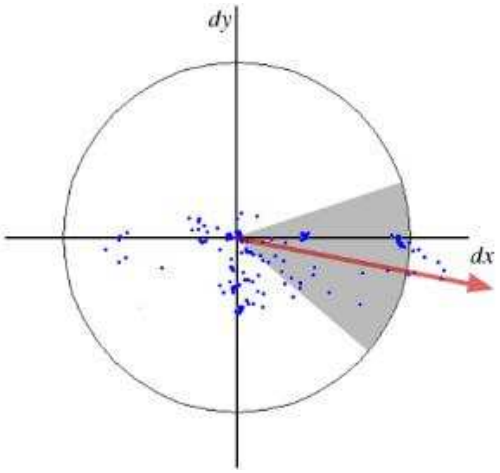


FIGURE 2.12: The orientation of key-point is calculated in a circular neighborhood using a sliding orientation window

Tables 2.1 and 2.2 give an overview of the most popular feature detectors and descriptors. Detectors are organized in two groups according to the type of feature extracted in the image: corner detectors and blob detectors. Descriptors also have two groups: binary descriptors and spectra descriptors.

2.6 Conclusions and Discussion

A logo is a complex graphic element which includes a variety of different information such as color, shape, texture, and other features. Therefore, to describe logos, we should select a robust description technique. Regarding the description part, in our view, the point-based description technique is able to carry all this information, and is suitable to deal with complex graphic elements as logos. On the other hand, shape-based description techniques usually need a segmentation step to segment candidate regions. However, segmentation techniques are still embedded with shortcomings when dealing with object overlap problems.

Once the suitable description technique has been chosen, we have to think how we should organize all the information arising from the document collection to provide an efficient search access. Indexing-based approaches with hashing techniques give some advantages to avoid traversal steps and costly balancing algorithm. However, the effect of these hashing techniques depends on hash function. A good hash function can give better results but its cost can be higher than the inner loop of the lookup algorithm for a sequential list or search tree. Learning-based approaches usually connect to segmentation problems such as overlapping and touching between text and graphics. Although key-point-based approaches are considered to be costly approaches, they can give best

TABLE 2.1: An overview of the most popular feature detectors.

Feature Detectors	Corner	Blob	Rotation invariant	Scale invariant	Notes
Harris	x	-	x	-	the intensity changes significantly in at least 2 directions
FAST	x	-	-	-	An intensity threshold and a connected set of pixels in a circular pattern to determine a corner
oFAST (ORB)	x	-	x	(x)	FAST features at each level in the pyramid. For orientation invariant and the intensity centroid computing the orientation of the key-point
BRISK	x	-	x	x	Using FAST or AGHAST based selection in scale space
DoG (SIFT)	(x)	x	x	x	Extreme interest point based on difference-of-Gaussian function in scale-space
FH (SURF)	-	x	x	x	The determinant maxima points of the Hessian matrix
MSER	-	x	x	x	Using a binary intensity thresholding loop

TABLE 2.2: An overview of the most popular feature descriptors.

Feature Descriptor	Year	Pattern shape	Pattern size	Feature size	Robustness
BRIEF	2010	Square	31x31	256	brightness, contrast
rBRIEF (ORB)	2011	Square	31x31	256	brightness, contrast, rotation, limited scale
BRISK	2011	Square	31x31	512	brightness, contrast, rotation, scale
FREAK	2012	Square	31x31	256	brightness, contrast, rotation, scale,viewpoint, blur
SIFT	2004	Square, with circular weighting	16x16	128	brightness, contrast, rotation, scale, affine transforms, noise
SURF	2006	Square, with circular weighting	16x16	64	scale, rotation, illumination, noise

results. For instance, in nearest neighbor technique, a sequential algorithm can give better results than other techniques. The performance optimization of logo spotting based on key-point matching will be discussed in Part II of this thesis.

Finally, regarding the type of the local feature extraction, we should select one of the rotation and scale invariant detectors because logos may occur on documents with different sizes or different document sizes. In addition, to deal with potential problems of scanned documents, descriptors require characteristics including brightness, contrast, rotation, scale, affine transforms, noise. At the end of Chapter 3, we will present a benchmarking of feature detectors and descriptors on isolated logo images. Then, a comparison of detectors and descriptors for logo spotting on documents will be made in Chapter 4.

Chapter 3

A Key-point Matching Approach to Logo Spotting

3.1 Introduction

In this chapter we present a framework for spotting logos in document images by using a key-point matching approach based on local feature description of logos. Each logo and document is represented by a set of key-points. The presented framework matches pairs of key-points from the logo's key-points to the document's key-points. A density-based clustering method is then used to segment and remove outliers. The spacial geometry of matches is verified by using homography with RANSAC. The effectiveness of this framework is demonstrated by experiments which compare the performances of different combinations of key-point detectors and descriptors. First, the experiments are carried out on two real datasets on isolated logo at the end this chapter. Then, the experiments based on the proposed framework will be presented for logo spotting on documents in Chapter 4.

3.2 The proposed Framework for Logo Spotting

The framework is inspired by the object recognition work presented by Lowe in [26]. In that work, the author uses key-point matching to find the best candidate match for each key-point from the images of the query objects by identifying its nearest neighbor in the set of key-points from the training images. Figure 3.1 gives an overview of our proposed framework for spotting a logo/logos in a document based on key-point matching method. Each step will be presented in detail in the following sections.

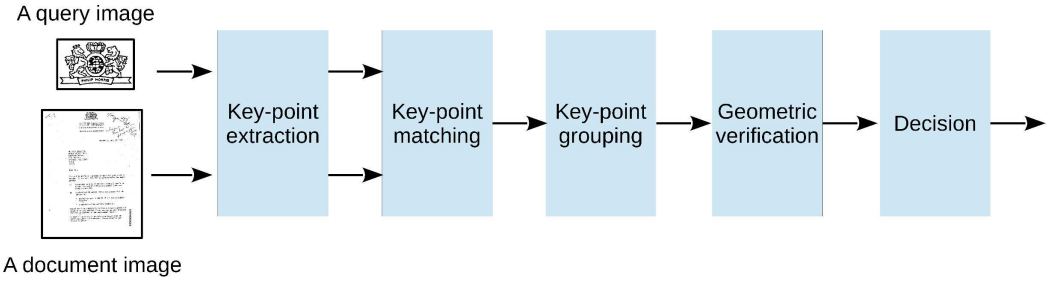


FIGURE 3.1: The proposed framework for logo spotting.

- **Key-point extraction**: This is the first step of our framework. In some cases, a preprocessing step can be added before this step to improve the performance such as noise removal, etc. In key-point extraction step, all interest points of both the logo and document image are detected by detectors. Then each point is described by a local descriptor (or more than one local descriptors). Therefore, a logo and a document are represented by sets of key-points (interest points).

- **Key-point matching**: The main idea of the key-point matching method is to find the best candidate match for each key-point from logo images in the set of key-points from the document images. The best candidate match can simply be one of its nearest key-points in a distance space (usually Euclidean distance) or can be estimated by using some rules to reduce the false matches or by combining some descriptors to increase the true matches.

- **Key-point grouping**: A density-based clustering method is applied to group the matches. This step is added to our framework to segment candidate logo regions which occur on the document. The second advantage of this step is to reject the incorrect matches which are outliers. The outliers of density-based clustering are lonely matches or small match groups which have too few matches to be a cluster. Therefore, they can not be candidate regions.

- **Geometric Verification and Localization**: The key-point grouping step groups matches into candidate regions and removes the incorrect matches as outliers. However, there are still some incorrect matches within each candidate region. To maximize the performance and localization, these incorrect matches should be removed. We propose to perform this step by using the homography with RANSAC to verify the matches of each candidate region.

- **Decision**: In this step, we use the best matches to estimate if a document contains a logo in document categorization or to rank documents in document retrieval applications. These two applications will be presented in the next chapter.

3.3 Logo and Document Representation

A given logo L_i is represented by the n_i interest points extracted by a detector. Each of these key-points is then described by a feature vector constricted by a local descriptor or more than one local descriptor. It can be denoted as:

$$L_i = \{(x_k, y_k, D_k)\} \quad \text{with } k \in \{1, .., n_i\} \tag{3.1}$$

and in the situation where uses more than one local descriptor (two descriptors):

$$L_i = \{(x_k, y_k, D'_k, D''_k)\} \quad \text{with } k \in \{1, .., n_i\} \tag{3.2}$$

where x_k, y_k are the x- and y-position of the k^{th} detected local feature key-point. D_k is a descriptor vector of the key-point with its length depending on the type of local feature descriptor (for instance, a 128-dimensional vector of SIFT descriptor and a 256-dimensional vector of BRIEF descriptor). Similarly, D'_k, D''_k are the first and second descriptors, respectively. An individual key-point q from the i^{th} logo is denoted by L_i^q.

Similarly, a document T_i is represented by a set of local feature key-points detected in the i^{th} document:

$$T_i = \{(x_k, y_k, D_k)\} \quad \text{with } k \in \{1, .., m_i\} \tag{3.3}$$

and in more than one local descriptor case (two descriptors):

$$T_i = \{(x_k, y_k, D'_k, D''_k)\} \quad \text{with } k \in \{1, .., m_i\} \tag{3.4}$$

and where each element is defined similarly as above.

3.4 Key-point Matching

3.4.1 Key-point Matching with Simple Nearest Neighbor

Key-point matching is done by matching pairs among a set of key-points representing a logo and a set of key-points detected in a document. The best match for a key-point of the logo is found by identifying its nearest neighbors in the set of key-points of the document. In this method, the nearest neighbor is defined as the key-point with minimum distance in descriptor space. A given threshold τ on the distance is used to filter the false matches.

3.4.2 Key-point Matching with 2-Nearest Neighbor Matching Rule

Key-point matching with simple nearest neighbors can work well in some cases. However, it also provides many false matches. Thus, many key-points from the logo image will not have any correct matches in the document image because they are obtained from background clutter or are not detected in the document image. A more effective method is to compare the distance of the first nearest neighbor to that of the second nearest neighbor. This allows us to obtain the best candidate matches because of the discrimination between the first nearest neighbor and the second one.

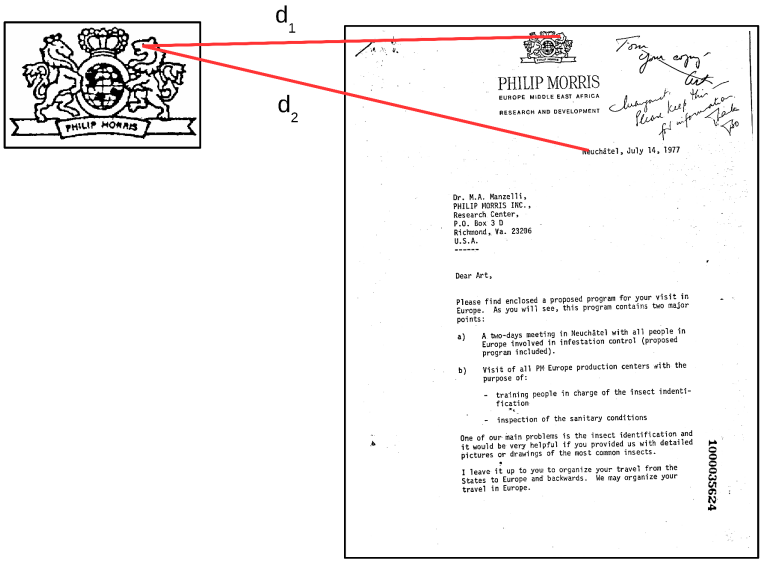


FIGURE 3.2: Each key-point of a logo, its two nearest neighbors in a document are estimated.

For each key-point q of the logo L_j, we compute the two nearest neighbors in the set of key-points of the document T_i:

$$\begin{aligned} d_1(L_j^q, T_i) &= \min_k \{dist(D_q, D_k)\} \\ d_2(L_j^q, T_i) &= \min_{k \neq d_1(L_j^q, T_i)} \{dist(D_q, D_k)\} \end{aligned} \tag{3.5}$$

where $dist(D_q, D_k)$ is a function that returns the distance between two descriptor vectors D_q and D_k. Then the ratio r of the distance to the first and second nearest neighbors is used for matching:

$$r = \frac{d_1(L_j^q, T_i)}{d_2(L_j^q, T_i)} \tag{3.6}$$

If the ratio r is greater than a given threshold ϕ then it means that the matching is not reliable, as there is a possible ambiguity between the two nearest neighbors. On the other hand, if r is lower than ϕ, then the key-point is representative enough to be considered. Figure 3.2 illustrates the two nearest matches of a key-point.

3.4.3 Key-point Matching with Post-filter by a second Descriptor

Key-point matching with the 2-nearest neighbor matching rule can give the best candidate pairs of key-points. However, in practice, there are many correct matches that are rejected by this method (Figure 3.3). This is one of the reasons for the decrease of the true matches. In addition, estimating the threshold r is difficult; that is, if r is lowered towards 0, then the accuracy of the matches increases, but the number of correct matches decreases. Conversely, if r is increased towards 1, then the accuracy decreases while the number of correct matches increases. The accuracy and number of correct matches are two influential factors on the overall result. This is demonstrated by the experiments in Section 3.7.

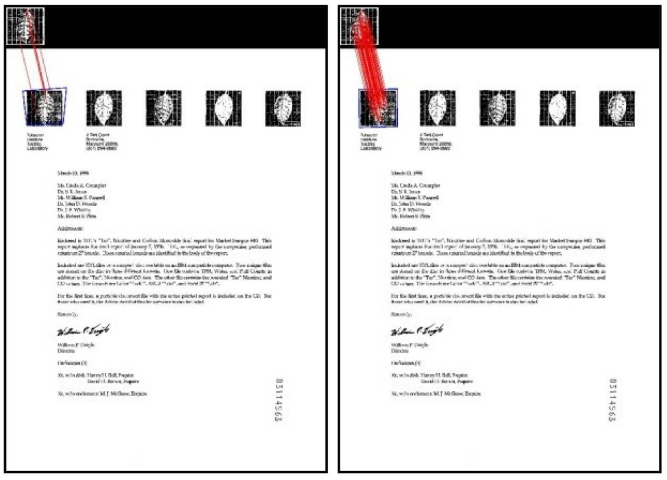
Key-points are rejected by the 2-nearest neighbor matching rule because the first and the second nearest neighbors are not far enough apart in their feature space under consideration. Thus, our hypothesis is that in other feature spaces, the rejected key-points can be matched. Therefore, to solve the shortcomings raised by this method, in this section, we propose a method to integrate a post-filter by using another descriptor in the key-point matching step. Such a method gets more pairs of correct key-points without increasing the false positives.

The proposed method has two stages. First, the key-point matching with 2-nearest neighbor matching rule presented in section 3.4.2 is applied to filter the matched key-points in the first feature space. In the second stage, the second feature space is used to post-filter the key-points that are rejected by the first stage.

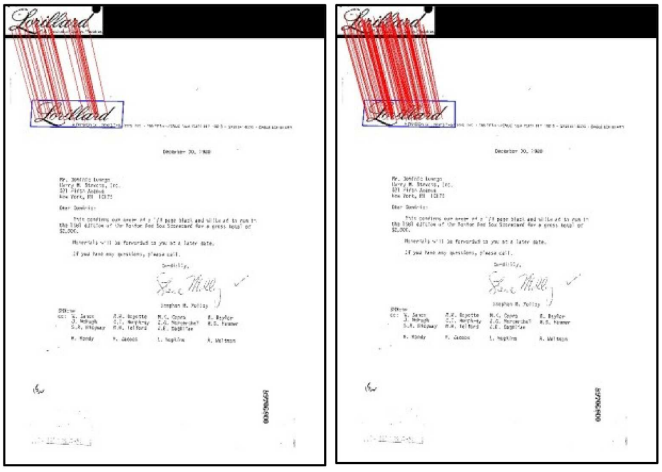
A given query L with n key-points and a document T with m key-points are described by two local descriptors as follows:

$$\begin{aligned} L &= \{(x_j, y_j, D'_j, D''_j)\} \qquad \text{with } j = 1..n \\ T &= \{(x_j, y_j, D'_j, D''_j)\} \qquad \text{with } j = 1..m \end{aligned} \tag{3.7}$$

where x_j and y_j are the x- and y-position of j^{th} key-point. D'_j and D''_j are the first and second descriptor vectors of this key-point, respectively.



(a)



(b)

FIGURE 3.3: Two examples of using post-filtering step; (left) few correct key-points are matched by the nearest neighbor matching rule based on the two nearest neighbors only; (right) many correct key-points are matched by our proposed post-filter step.

The aim of the first stage is to find the k nearest neighbors $t_i \in T$, $i = 1..k$ of query feature q, $q \in L$. Let d_i represent the distances between document features t_i and query feature q in the D' feature space with $i = 1..k$ so that $d_1 \leq d_2 \leq ... \leq d_k$. (see Figure 3.4)

$$\begin{aligned} d_i(q,T) &= \min_{j \neq d_r(q,T); r=1..i-1} \{dist^*(D'_q, D'_j)\}, i = 1..k \\ \text{with } d_1 &= \min_j \{dist^*(D'_q, D'_j)\} \end{aligned} \tag{3.8}$$

where $dist^*(D'_q, D'_j)$ returns the distance between key-point q and key-point j in the D' descriptor space. In this stage, we consider the two nearest neighbors t_1 and t_2 with their distances d_1 and d_2. If the ratio $r = \frac{d_1}{d_2}$ is lower than a given threshold ϕ, the pair

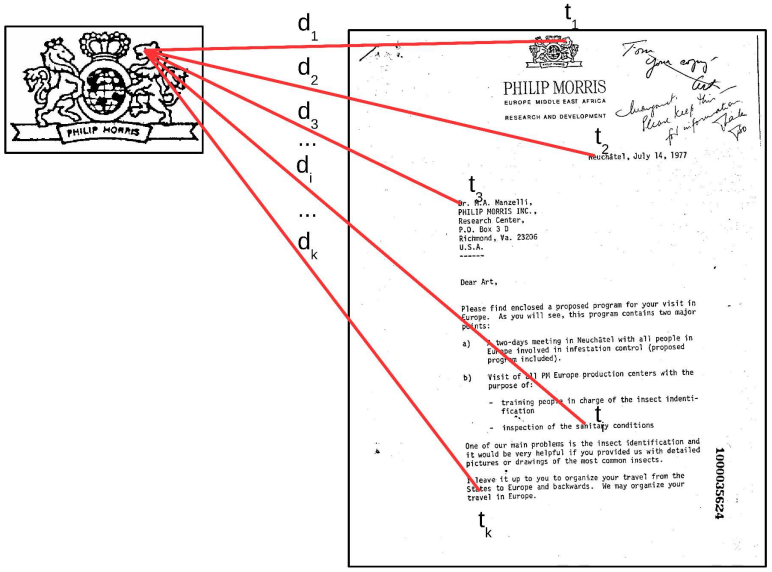


FIGURE 3.4: Each key-point of a logo, its k nearest neighbors in a document are estimated.

of key-points (q, t_1) is accepted. Otherwise, its closest key-point will be searched again within k key-points $(t_1, t_2, .., t_k)$ in the D'' feature space.

In the second stage, the minimum distance d_{min} between q and $t_{min} \in [t_1, .., t_k]$ is computed in D'' descriptor space.

$$\begin{aligned} d_{min} &= \min_{i=1..k} \{dist^{**}(D''_q, D''_{t_i})\} \\ t_{min} &= \operatorname*{argmin}_{i=1..k} \{dist^{**}(D''_q, D''_{t_i})\} \end{aligned} \tag{3.9}$$

where $dist^{**}(D''_q, D''_j)$ returns the distance between key-point q and key-point j in the D'' descriptor space. The closest key-point t_{min} is the best candidate matching key-point. However, we use a given threshold θ to filter the cases whose distances are too far. If d_{min} is lower than θ, the pair of key-points (q, t_{min}) is matched. Otherwise, this pair is rejected. The proposed key-point matching with post-filter method is summarized in Algorithm 1.

Algorithm 1 Key-point Matching with Post-filtering algorithm

Input: Query $\mathbf{L} = \{(x_j, y_j, D'_j, D''_j)\}$ with $j = 1..n$
Document $\mathbf{T} = \{(x_j, y_j, D'_j, D''_j)\}$ with $j = 1..m$
Value: k
Thresholds: ϕ, θ
Output: List of pairs of matched key-points

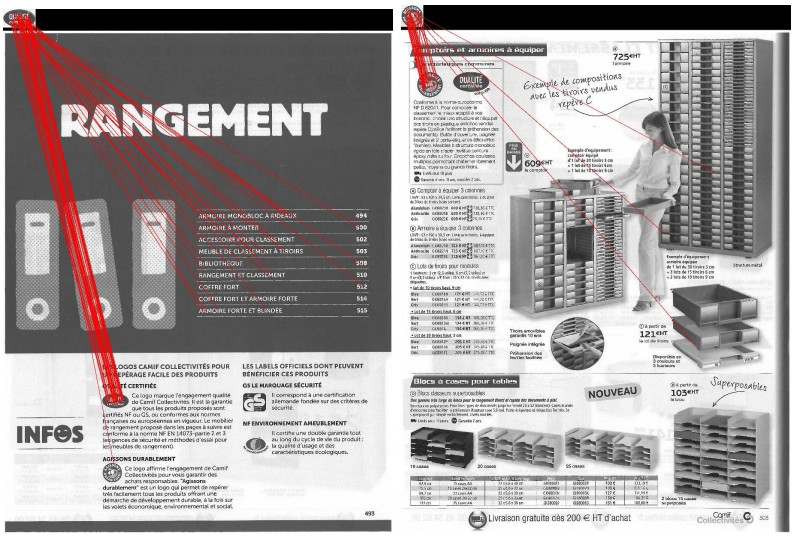
for each key-point q of query image **do**
  Find k-nearest neighbor $t_i \in \mathbf{T}$ with their distance d_i in D' descriptor space so that $d_1 \leq d_2 \leq .. \leq d_k$ with $i = 1..k$
  if $r = \frac{d_1}{d_2} \leq \phi$ **then**
    pair of key-points (q, t_1) is accepted
  else
    Find the nearest neighbor t_{min} of q within the k key-points in D'' descriptor space, with $d_{min} = \min_{i=1..k}\{dist^{**}(D''_q, D''_{t_i})\}$
    if $d_{min} \leq \theta$ **then**
      pair of key-points (q, t_{min}) is accepted
    else
      no key-point matches with q, and q is rejected
    end if
  end if
end for

3.5 Grouping the Matching Key-points

3.5.1 Motivation

After the matching stage, normally, there are a geometric verification step and an accumulating histogram H for counting the number of matching key-points corresponding to each logo and deciding which is the logo in the query document. However, not all the matching key-points are located inside the logo region. Some of them are located in other regions of the document image. An example in Figure 3.5 shows that many matches are located outside of logo region. In another case, as illustrated in Figure 3.6, there is a high concentration of matches on the correct logo while the concentration decreases when considering the incorrect logo. These lead to incorrect spotting. Therefore, we consider these key-points as noises and we do not take them into account in our final decision. For example in Figure 3.5(b), there is a high density of matches on regions where logos occur on a document.

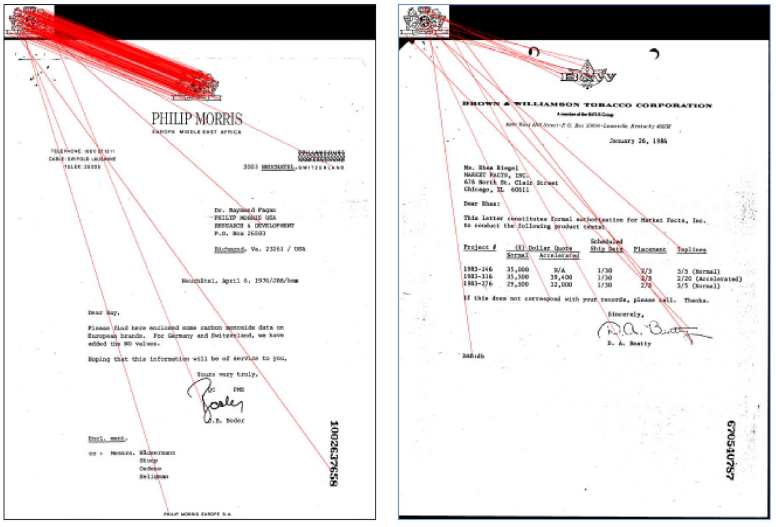
We therefore decided to add an intermediate segmentation step, where the spatial density of the key-points in the query document is taken into account, in order to filter key-points and enhance our decision. In this segmentation step, we use a density-based clustering method because it does not require that the number of clusters be specified as opposed to k-means and it can find arbitrarily shaped clusters. In the next subsection, we will



(a) (b)

FIGURE 3.5: There are many matched key-points outside logo regions. These key-points are considered as noise and they should be removed.

present the Density-based spatial clustering of applications with noise (DBSCAN) – a well-known density-based clustering method – which is used in our experiments.



(a) (b)

FIGURE 3.6: The lines show the matching between key-points of a logo (top left corner) and a document (bottom) (a) high density of matches on the correct logo and (b) low density of matches on the incorrect logo.

3.5.2 Density-based Clustering Algorithm

Density-based spatial clustering of applications with noise (DBSCAN), proposed by Ester et al. [60], is a density-based clustering algorithm. The ϵ-neighborhood of a point

$\mathbf{x}$ is composed of points within a radius ϵ from $\mathbf{x}$, as follows:

$$N_\epsilon(x) = \{y | \delta(x, y) \leq \epsilon\} \tag{3.10}$$

where $\delta(x, y)$ is the distance between points x and y. The Euclidean distance is usually used, i.e., $\delta(x, y) = \|x - y\|_2$. However, other distance metrics can also be used.

ϵ-neighborhood of a point $\mathbf{x}$ has "high density" if it contains at least $MinPts$ of points and "low density" otherwise. $MinPts$ is a user-defined density threshold. Figure 3.7 displays an illustration of ϵ-neighborhood, high density and low density of points.

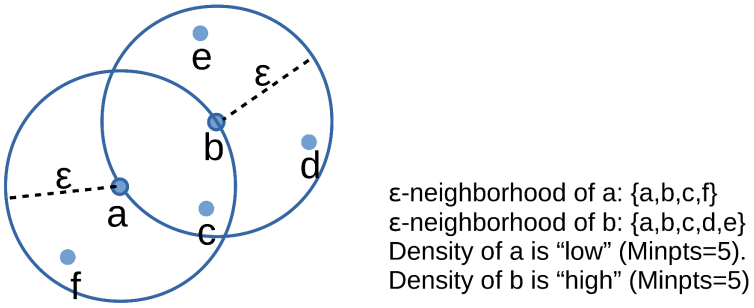


FIGURE 3.7: Examples of ϵ-neighborhood, high density and low density of points.

A point $\mathbf{p}$ is a *core point* if it has "high density", meaning that there are at least $MinPts$ points in its ϵ-neighborhood. In other words, $\mathbf{p}$ is a core point if $|N_\epsilon(\mathbf{p})| \geq MinPts$. A *border point* is a point which has fewer than $MinPts$ point in its ϵ-neighborhood but belongs to the ϵ-neighborhood of one or many core points. Finally, a point that is not a core point nor a border point is defined as a *noise point*.

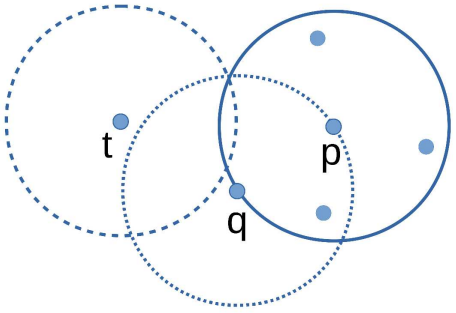


FIGURE 3.8: Examples of a core point, border point and noise point.

An example for core point, border point and noise point is displayed in Figure 3.8. Using $MinPts = 5$, here $\mathbf{p}$ is a core point since $|N_\epsilon(\mathbf{p})| = 5$. $\mathbf{q}$ is a border point since $|N_\epsilon(\mathbf{q})| = 3$, but it is reachable from $\mathbf{p}$. Finally, $\mathbf{t}$ is a noise point.

A point **q** is *directly density reachable* from another point **p**, if **p** is a core point and **q** is in the ϵ-neighborhood of **p**, i.e. $q \in N_\epsilon(p)$. A point **q** is *density reachable* from a point **p**, if there exists a string of points, $\mathbf{q} = \mathbf{q_0}, \mathbf{q_1}, ..., \mathbf{q_k} = \mathbf{p}$, such that $\mathbf{q_i}$ is directly density reachable from $\mathbf{q_{i-1}}$. In other words, there is set of core points leading from **p** to **q**. Note that density reachability is an asymmetric or directed relationship. Finally, define any two points **q** and **p** to be density connected if there exists a core point **z**, such that both **q** and **p** are density reachable from **z**. We can now define a density-based cluster as a maximal set of density connected points.

Algorithm 2 DBSCAN algorithm

```
procedure DBSCAN(D, ε, MinPts)
C ← 0
for each unvisited p ∈ D do
  mark p as visited
  N_ε(p) ← neighborhood(p,ε)
  if |N_ε(p)| < MinPts then
    mark p as NOISE
  else
    C ← next cluster
    expandCluster(p, N_ε(p), C, ε, MinPts)
  end if
end for
end procedure

procedure expandCluster(p, N_ε(p), C, ε, MinPts)
C ← p
for each point p' ∈ N_ε(p) do
  if p' is not visited then
    mark p' as visited
    N_ε(p') ← neighborhood(p',ε)
    if |N_ε(p')| ≥ MinPts then
      N_ε(p) ← N_ε(p) ∪ N_ε(p')
    end if
  end if
  if p' is not yet member of any cluster then
    C ← p'
  end if
end for
end procedure

function neighborhood(p,ε)
return  all ε-neighborhood points of p
end function
```

Algorithm 2 shows the pseudo-code for the DBSCAN algorithm. Initially, all the points are marked as unvisited. First, the ϵ-neighborhood of each point $x \in D$ is computed. If it does not contain at least $MinPts$ points, the point is labeled as noise. Otherwise,

a cluster C is started. Next, a procedure to expand the cluster C is used to find all other points density connected to $\mathbf{x}$. For each density connected point p', if p' is a core point, all points within ϵ-neighborhood of the density connected point are added into the cluster C. Otherwise, if p' is not yet a member of any cluster, p' is added into the cluster C.

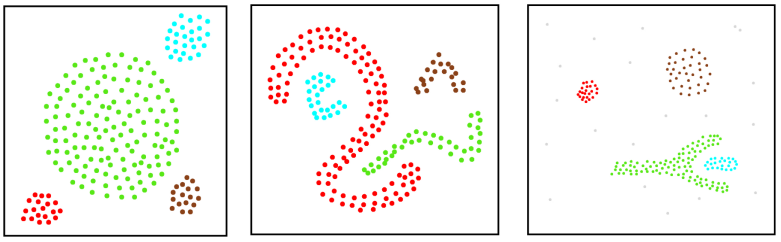
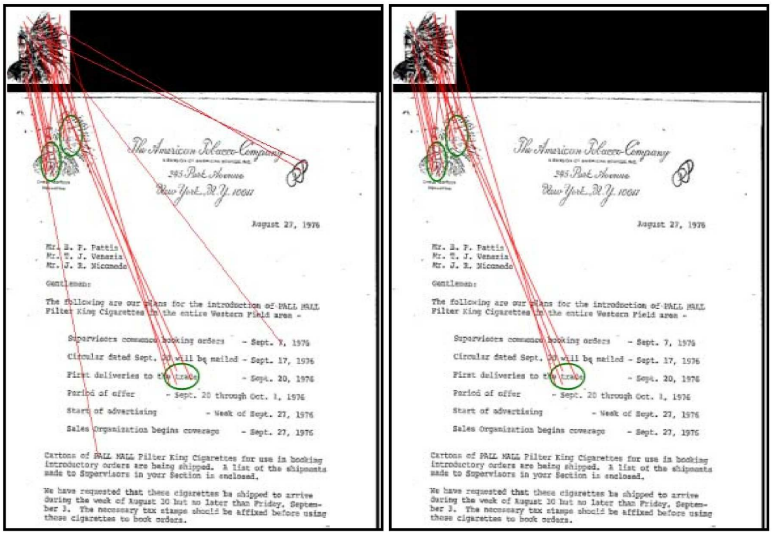


FIGURE 3.9: Example of clusterings discovered by DBSCAN (extracted from [60]).

3.6 Geometric Verification and Localization

Grouping the matched key-points helps us to reduce the number of incorrect matched key-points which are as outliers and are considered as noise, for example, Figure 3.10 shows that some incorrect matches are removed. However, there are till some incorrect matches which occur within the key-point groups, for example in Figure 3.11. This can be a cause of imprecise localization or incorrect spotting. To maximize our performance of logo spotting, these key-points need to be removed before the localization step starts.



(a) (b)

FIGURE 3.10: Before (a) and after (b) grouping and segmentation by DBSCAN.

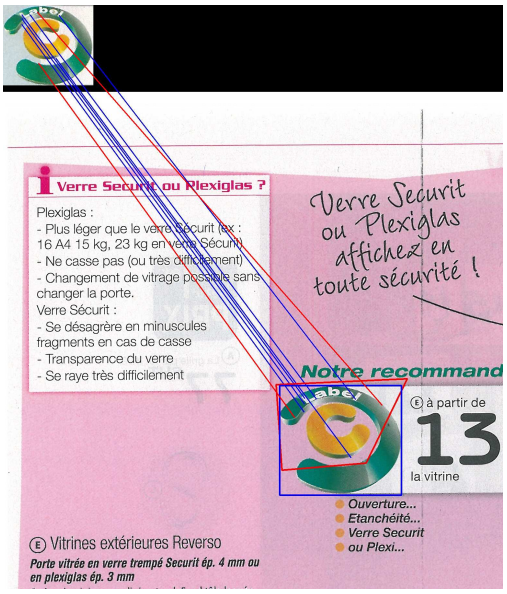


FIGURE 3.11: Example for geometric verification of matching, there are some incorrect matches of pairs of key-points (red lines).

3.6.1 Homography using RANSAC

An homography is an invertible transformation from points in R^2 to points in R^2 that map lines to lines [61]. Homography is used in many applications relying on geometry. In the field of computer vision, homography can be employed to compute matches between images [61]. In a 2D plane, let us consider a set S of source points in the original plane $s_i(x'_i, y'_i)$, and a set T of target points $t_i(x_i, y_i)$ in the target plane. Homography is a perspective transformation H between the source and the target planes:

$$s_i \begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} \sim H \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} \tag{3.11}$$

where H is the 3×3 homography matrix:

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{3.12}$$

and the back-projection error (for projecting the target points in the original plane) is:

$$\sum_i \left(x'_i - \frac{h_{11}x_i + h_{12}y_i + h_{13}}{h_{31}x_i + h_{32}y_i + h_{33}} \right)^2 + \left(y'_i - \frac{h_{21}x_i + h_{22}y_i + h_{23}}{h_{31}x_i + h_{32}y_i + h_{33}} \right)^2 \tag{3.13}$$

We need to estimate the homography matrix H so that the back-projection error is minimized. Among many algorithms to estimate the homography H, the RANdom SAmple Consensus (RANSAC) algorithm, presented by Fischler and Bolles in [62], is a very robust algorithm for estimating H. It is robust to the presence of outliers. The main idea is to identify the outliers as data samples with greatest residuals with respect to the fitted model. The steps of the general RANSAC algorithm are as follows [61]:

1. Randomly select a subset s of n points of S and estimate the homography model from s and the corresponding subset t of matched key-points in T.
2. Determine the set of inliers and outliers based on a distance threshold θ to the model.
3. If the number of inliers is greater than some threshold Θ, re-estimate the model and terminate.
4. If the number of inliers is less than Θ, then select a new subset s and repeat.
5. After repeating N times, the largest consensus set of inliers is selected, and the model is re-estimated using this set.

In our application, we compute a homography matrix H from the matched key-points in the logo to its corresponding candidate logo region in the document. Therefore, the logo is considered as a source plane while the target plane is the candidate logo region. The parameters of this algorithm are discussed in [61]. We set the values of these parameters using experiments.

3.6.2 Filter by Homography and Localization

We propose an algorithm to filter incorrect matched key-points and to localize the logo based on homography using RANSAC. First, we try to find a transformation between the pairs of matched key-points in each cluster to integrate spatial relationships between the key-points in the logo and those in the clusters. Then we compute the transformation again after determining all matched key-points in a bounding box covering the logo region candidate. This step aims to solve cases where a cluster region may not cover the logo region, but just a part of the logo region. An example, Figure 3.10, of a cluster does not cover a logo where there are two clusters in the logo region. In addition, to reduce

the false positive, if detected regions are not convex polygons, we consider them to be negative results. This can be explained by that if we use four corners of logo image (a convex polygon in 2D plan) to find a bounding box, having four corners, of logo region on a document by a transformation, the bounding box has to be a convex polygon too.

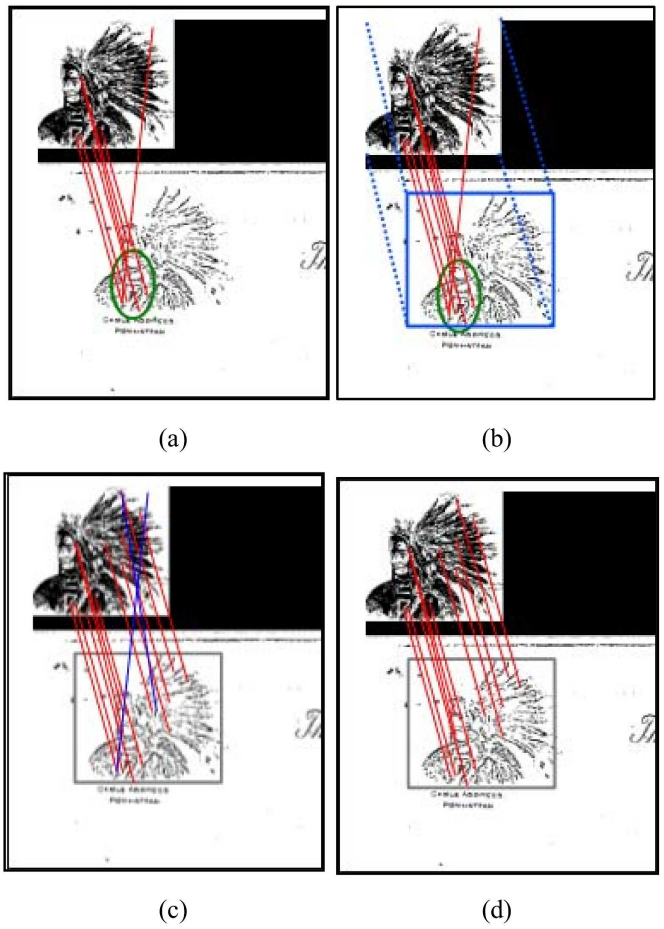


FIGURE 3.12: The filter with transformation in our approach. Blue lines are the pairs of matched key-points which are rejected.

Our algorithm is as follows:

For each logo region candidate:

1. Find a transformation H between the pairs of the matched key-points in the logo region candidate. Let $s_i(x_{i1}, x_{i2})$ be the coordinates of the matched key-points in the logo gallery, and let $t_i(y_{i1}, y_{i2})$ be the coordinates of the matched key-points in the query document image (Figure 3.12a).

2. Determine a bounding box which may contain a logo in the query document, thanks to the transformation H and the four corners of the minimal bounding box of the logo in the logo gallery (Figure 3.12b).

3. Re-estimate the transformation H using all the pairs of matched key-points in the bounding box (Figure 3.12c).

4. Filter the incorrectly matched key-points: if $\|t_i - H(s_i)\| \geq \theta$ then reject this pair (Figure 3.12c)

5. Finally, the position of the logo is estimated based on the four corners and the transformation H. In addition, the convex polygon checker is used to verify the localization.

3.7 Benchmarking Feature Detectors and Descriptors

As discussed above, each detector and descriptor has advantages and disadvantages in different contexts. To find the best combination in order to use it in our key-point matching framework, we present – in this section – several experiments that we have conducted to compare robust combinations of detectors and descriptors. They are tested on larger isolated logo image datasets.

3.7.1 Database

We experiment with two sets of isolated logo images from two databases. The first dataset comprises 113 isolated logo images which are extracted from our magazine database with 6 classes of logos. The second one is a set of 432 isolated logo images with 35 classes extracted from all 412 documents containing logos of Tobacco-800. The description of the two database is presented in Appendix A. Figure 3.13 shows the example isolated logo images from the two datasets.

3.7.2 Experimental Setup

We consider a notion of matching based on combinations of detectors and descriptors of isolated logo images. Therefore, each logo image is used as a query to match with each logo image of the rest of dataset as a scene. The pairs of matched key-points are computed based on the 2-NN matching rule and then the geometric verification step based on homography with RANSAC is applied to remove the outlier pairs of matched key-points.

From computer vision reviews, we consider the following seven robust combinations between detectors and descriptors for comparison which are associated with the best performance for computer vision applications in [26, 42, 43, 46, 49, 50] respectively:

(a)

(b)

FIGURE 3.13: (a) Isolated logo images extracted from our magazine database and (b) Tobacco-800 database.

DoG+SIFT, FH+SURF, BRISK, FH+BRIEF, BRISK+FREAK, ORB (oFAST+rBRIEF) and MSER +SIFT:

- **DoG+SIFT**: This combination initially relies on the DoG detector to detect image key-points. Each key-point is described by a 128 dimensional feature vector of SIFT [26].
- **FH+SURF**: A 64-dimensional orientation histogram of SURF descriptor is calculated from the distribution of 4 bins of Harr-wavelet responses in 4×4 windows around the FH key-points [46].
- **BRISK**: BRISK presents both a detector and a descriptor. BRISK descriptor provides a binary string of length 512 by concatenating the results of simple brightness comparison tests [43].
- **FH+BRIEF**: Following [49], the image keypoints first are detected by the FH approach and the Brief descriptor is computed as the binary comparison between two values of pairs of FH keypoints. The size of this vector descriptor is fixed to 256 in our experiments.

- **BRISK+FREAK**: This combination is presented in [50] using the multi-scale AGAST detector introduced by BRISK. FREAK also provides a bit-string descriptor and we use the fixed length 256 of the vector descriptor.
- **ORB (oFAST+rBRIEF)**: ORB is built based on FAST key-point detector and BRIEF descriptor. Each key-point is described as a binary string by simple binary test between pixels in a patch [42]. The size of this vector descriptor is fixed to 256 in our experiments.
- **MSER+SIFT**: Using an algorithm which is similar watershed algorithm [63], MSER provides the features which are actually shapes rather than points or corners. MSER performs well when combined with SIFT descriptor [48].

3.7.3 Evaluation Protocol

To evaluate the combinations between detectors and descriptors on the isolated logo images, we define a correct match and an incorrect match as follows:

- *A correct match* is the pair of matched key-points between two logo images from the same class (Figure 3.14(a)).
- *An incorrect match* is the pair of matched key-points between two logo images from different classes (Figure 3.14(b)).

Three popular evaluation measures, *precision*, *recall* and *accuracy* are used to evaluate our experiments. We use a given threshold ω to determine whether the query and the scene have the same logo class or not. If the ratio r between the number of matched key-points and the number of the query's detected key-points is greater than ω, it means that they are predicted to have the same logo class. Otherwise, they are predicted to have different logo classes. Therefore, true positive (TP), false positive (FP), true negative (TN) and false negative (FN) are defined as:

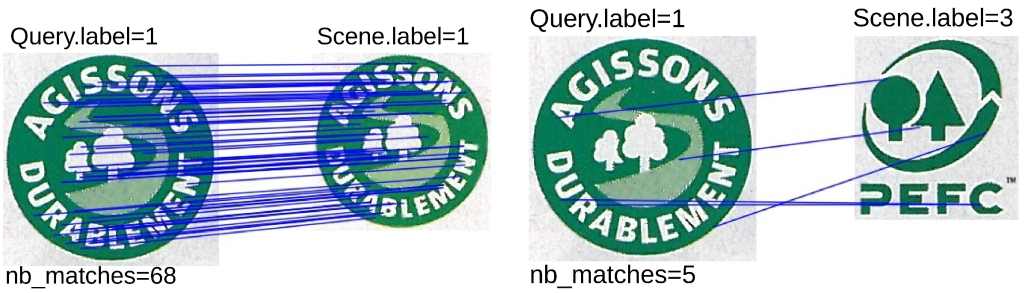
	query.label=scene.label	query.label≠scene.label
$r \geq \omega$	True positive	False positive
$r < \omega$	False negative	True negative

and then the *precision*, *recall* and *accuracy* are:

$$Precision = \frac{TP}{TP + FP} \tag{3.14}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.15}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.16}$$



(a) (b)

FIGURE 3.14: Examples of correct matches and incorrect matches: The number of correct matches are 68 (a) and the number of incorrect matches are 5 (b).

In addition, to compare the combinations of the detectors and descriptors, the Receiver Operating Characteristic (ROC) [64], or ROC curve, is also used. ROC is a graphical plot that illustrates the performance of a binary classifier system as its discrimination threshold is varied. ROC curves are very useful for visualizing and evaluating classifiers. They provide a richer measure of classification performance than scalar measures such as accuracy, error rate, or error cost, and have advantages over other evaluation measures, such as precision-recall graphs and lift curves [65].

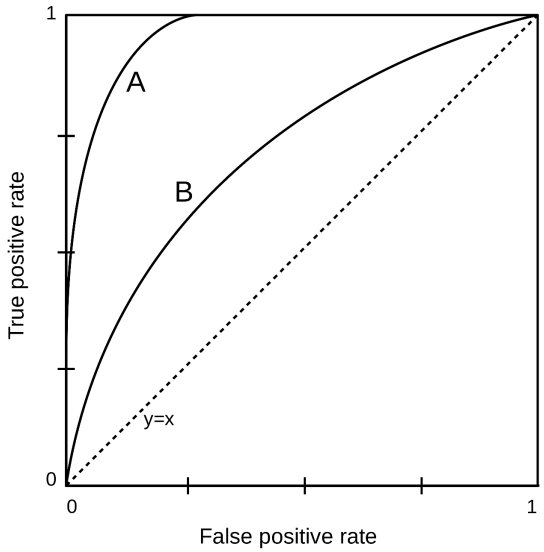


FIGURE 3.15: Example for ROC graph. Curve A is better than curve B

In ROC, the *true positive rate* (TPR), or *sensitivity* or *recall*, is plotted as a function of the *false positive rate* (FPR), or $1 - specificity$, at various threshold settings. TPR and FPR are described as:

$$TPR = \frac{TP}{TP + FN} \tag{3.17}$$

$$FPR = \frac{FP}{FP + TN} \tag{3.18}$$

On ROC graph, TPR is plotted on the y-axis and FPR is potted on the x-axis (see Figure 3.15). A perfect classifier will score at the top-left corner (point (0,1)). A worst case classifier will score at the bottom-right corner (point (1,0)). The line (y=x or TPR=FPR) represents the strategy of randomly guessing the class. Informally, one point on the ROC graph is better than another if it is located more top-left (TPR is higher, FPR is lower, or both cases). The area under the ROC curve is used to compare the performance of the classifiers. In Figure 3.15, curve A is better than curve B because it has a larger area under the curve.

3.7.4 Results

Correlation between the correct matches rate and the number of correct matches

In the key-point matching with the 2-nearest neighbor matching rule (see Section 3.4.2), estimating the threshold ϕ is difficult. In this experiment, we study the trend of the correct matches rate and the number of correct matched key-points when varying of the threshold ϕ from 0 to 1. Here, the correct match rate is computed as:

$$\text{Correct match rate} = \frac{\#\text{ of correct matches}}{\#\text{ of correct matches} + \#\text{ of incorrect matches}} \tag{3.19}$$

Figure 3.16 gives an overview of the variation of the correct match rate, the number of correct matches and the accuracy. Figure 3.16 shows that if ϕ is lowered towards 0, then the correct match rate increases, but the number of correct matches decreases. Otherwise, if ϕ is increased towards 1, then the correct match rate decreases while the number of correct matches increases. The correct match rate and the number of correct matches are the two influential factors on the overall result. Here, the accuracy curve in the graphs is computed by Equation 3.16 at the threshold $\omega = 0.05$.

Comparison of the performances of the combinations

In Figure 3.17, ROC curves are plotted based on the combinations. The points on the ROC plot represent sensitivity/specificity pairs corresponding to varying of the threshold ϕ from 0 to 1. As can be seen in the figure, the DoG+SIFT combination gives the best

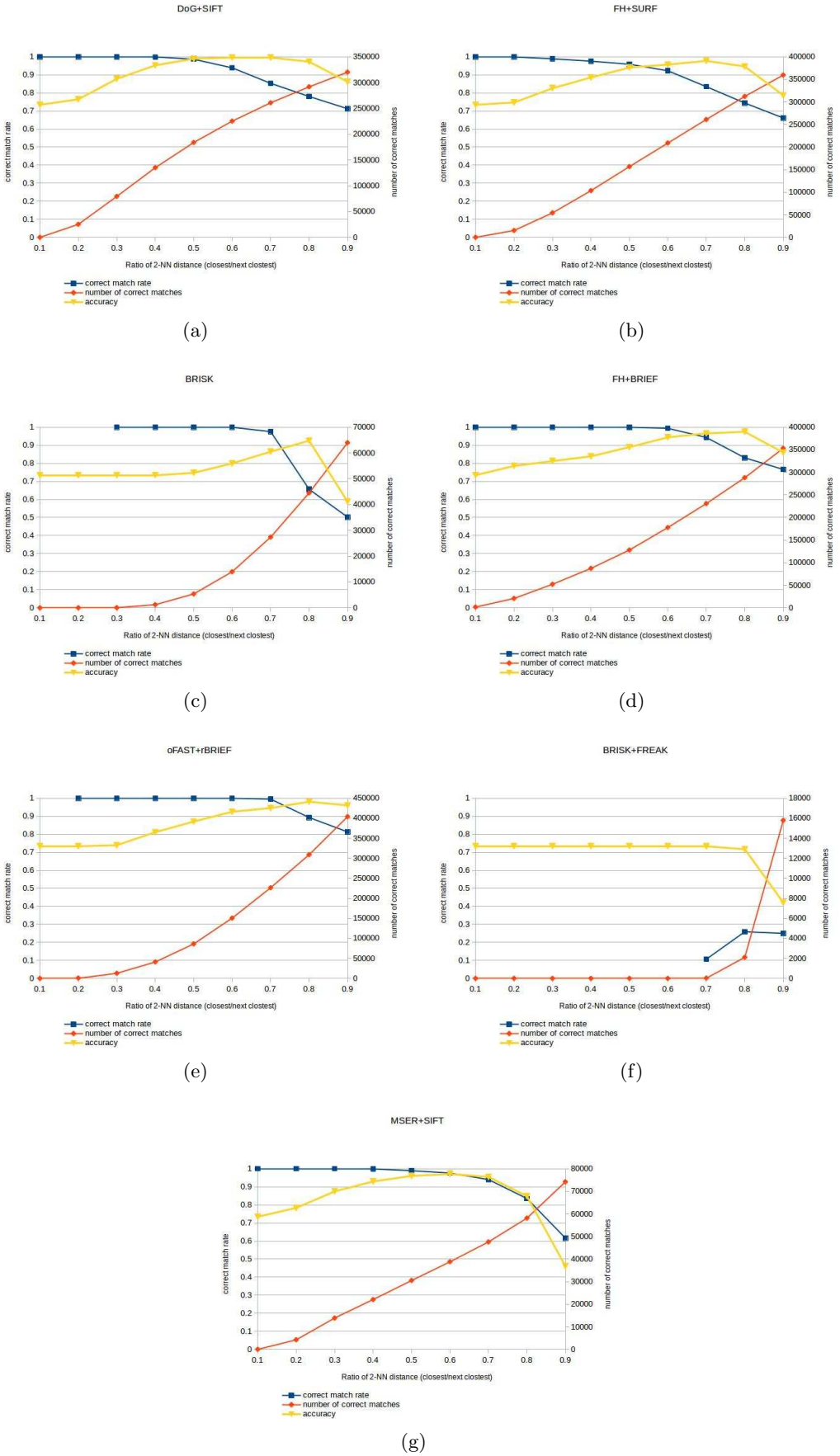


FIGURE 3.16: Correct match rate, the number of correct matches and accuracy as function of the threshold ϕ on 113 isolated logo images extracted from our magazine database.

performance while the BRISK+FREAK combination gives the worst one, and the second worse is the BRISK. The others have similar performances.

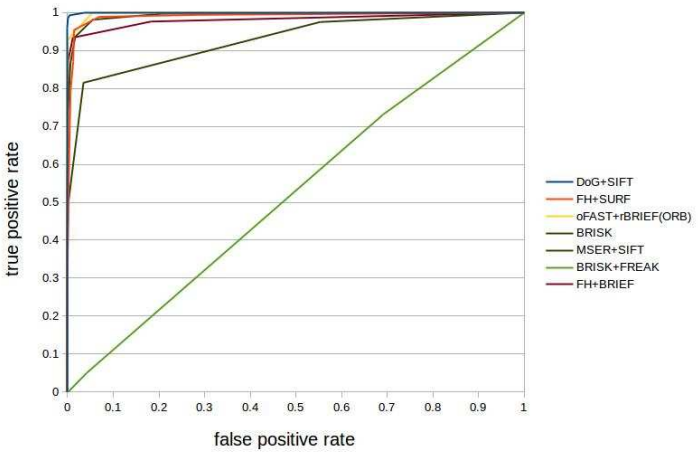


FIGURE 3.17: Experiment 1: Comparison between the combinations on ROC graph on 113 isolated logo images extracted from our magazine database. The best performance is the DoG+SIFT combinations.

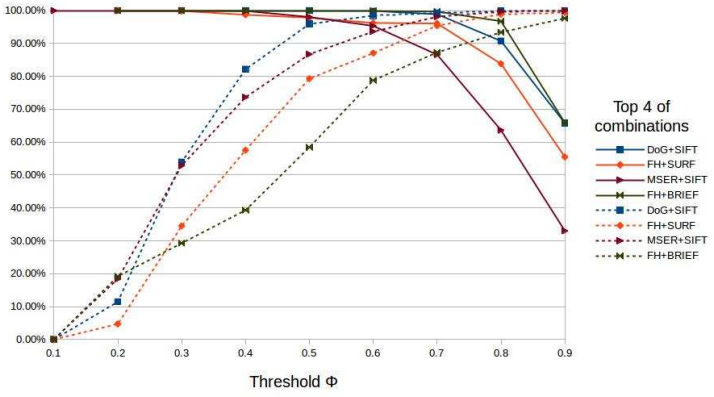


FIGURE 3.18: Top 4 of combinations of experiment 1 based on Precision (continuous lines) and Recall (dashed lines) as a function of the threshold ϕ. The DoG+SIFT combination is the best where the recall of about 95% can be obtained at precision of around 95% with values of the threshold ϕ between 0.5-0.7.

Concerning another evaluation protocol, in Figure 3.18, we examine this experiment on the Precision and Recall graph where precision and recall are a function of the threshold ϕ from 0.1 to 0.9. The graph shows the top four combinations for recall. The DoG+SIFT combination is also the best where the recall of about 95% can be obtained at precision of around 95% with values of the threshold ϕ between 0.5-0.7.

Figure 3.19 and Figure 3.20 show the comparison on the second dataset (Tobacco800 dataset). As we can see, the combination DoG+SIFT gives the best results in both the evaluation protocols.

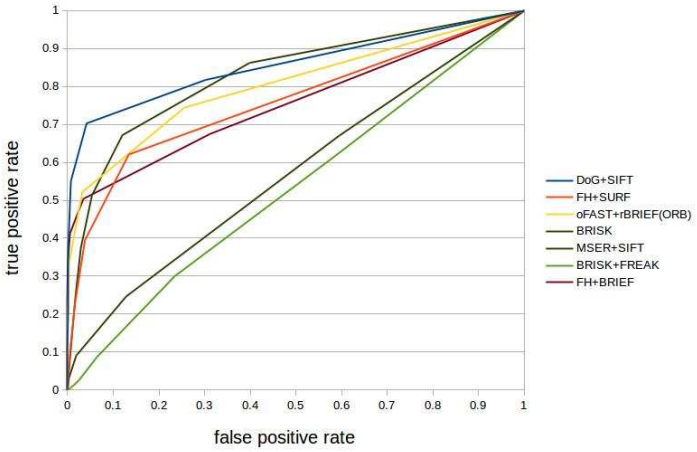


FIGURE 3.19: Experiment 2: Comparison between the combinations on ROC graph with Tobacco800 dataset. The best performance is also the DoG+SIFT combinations.

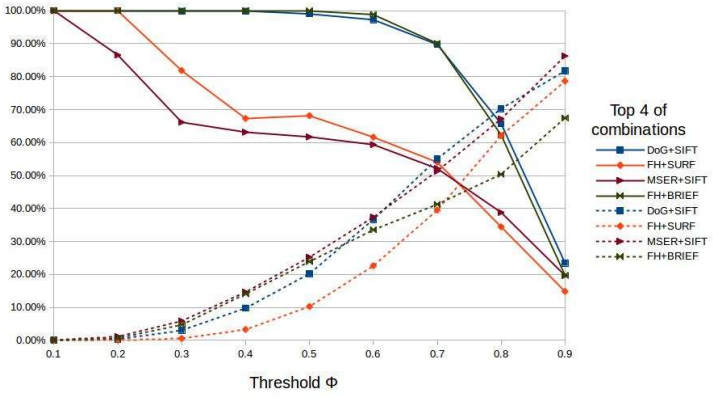


FIGURE 3.20: Top 4 of combinations of experiment 2 based on Precision (continuous lines) and Recall (dashed lines) as a function of the threshold ϕ on isolated logos extracted from Tobacco-800 database.

3.8 Discussion

In this work, we proposed a framework for spotting logos in document images using the key-point matching method. First, interest points of both a logo and a document are extracted and described as key-points by local detectors and descriptors. The matching key-point is an important next step to compute the best match candidate for each key-point from a query image. We also proposed a method to integrate a post-filter based on another descriptor into key-point matching step. Such a method gets more pairs of correct key-points without increasing the number of the false positives. Then a density-based clustering method is used to segment the matched key-points into clusters and to reduce the incorrect matches as outliers.

In this chapter, we have examined the correlation between the correct matches rate and the number of correct matches when varying the threshold ϕ. The benchmarking test is also constructed to compare the performances of the robust combinations between detectors and descriptors introduced in the Computer Vision field. The two sets of isolated logo images extracted from the large real databases are tested. The results show that the DoG+SIFT combination gives the best performance.

The proposed framework will be incorporated into the applications. In the next chapter, the first section will present logo spotting on the document collected from the advertisements. Then the document categorization and retrieval applications will be presented in the following sections.

Chapter 4

Applications: Document Categorization and Retrieval based on Logo Spotting

4.1 Review of Logo Spotting-based Applications

The last decades have seen an explosion in the amount of digitized document libraries. In order to properly index these documents, it is necessary to categorize or retrieve them. In the earlier times, several document classification systems were investigated, based on OCR and analysis of text by natural language processing. However, OCR systems reach good performances only with typewritten and printed documents, and the natural language processing depends greatly on the context. On the other hand, graphical objects in the document images contain much important information. In particular, logos (as well as stamps) are commonly used in documents, especially in business and administrative documents. It allows us to determine the source of the documents quickly and accurately, without any textual transcription and at a low cost.

In recent years, there has been a number of approaches proposed for logo detection [31, 32, 34, 35, 66], logo recognition [67–69] and logo spotting [27, 34]. In the earlier works, Seiden et al. [32] used the top-down X-Y cut algorithm to segment an image and then they detected logo and non-logo image regions using a classifier based on a set of sixteen features, mostly derived from statistics about connected components of black pixels. The system presented by Wang et al. [35] first uses an extension process of a rectangle and performs initial segmentation of logos. Then some rules based on geometric features such as logo positions, size and aspect ratio are applied to discriminate each image segment. Pham [66] presented a simple logo detection method. They assumed

that the spatial density of foreground pixels in a logo region is greater than that in non-logo regions. After binarizing a document image by global thresholding, the spatial density within each fixed size window is computed and the region with the highest density is hypothesized as a logo region. Doermann et al. [67], in one of the earliest works on logo recognition, presented an approach to logo recognition based on combination of text, shape and global and local affine invariants. Zhu et al. [34] presented an approach to logo detection and extraction in document images. They used a multi-scale boosting strategy. At a coarse image scale, a Fisher classifier provides an initial classification. Then, each logo candidate region is further classified at a finer image scale by a cascade of simple classifiers. Rusinol and Llados [27] proposed a method for document categorization by logo spotting. The graphical logo and the query documents are described by a set of local features and matched using a bag-of- words model. In order to filter the matching key-points and consider only the key-points belonging to the logo in the query document, they consider clusters of key-points.

In the field of logo retrieval, Jain and Doermann [36] presented an approach for logo retrieval in document images. In their work, the highest and lowest distinctiveness of the SURF features are used for scoring and sorting the key-points; then a filter method based on the properties of the features orientation and their geometric characteristics were used to verify the key-points. Meanwhile, Rusinol et al. [70] introduced a method for organizing and indexing logos based on a description using a variant of the shape context descriptor. A locality-sensitive hashing data structure was used to index these descriptors. In the earlier work, Zhu et al. [71] built a retrieval system using local shape context descriptors, measures of shape dissimilarity, and shape matching algorithms.

In this Chapter, systems for document categorization and retrieval based on logo spotting are presented. In the next Section 4.2, we test our framework – which has been introduced in Chapter 3 – on a set of real document images and logo images. An evaluation protocol for spotting is also discussed. The comparisons of performances of experiments on key-point matching methods and combinations of detectors and descriptors are presented in the last part of this chapter. Section 4.3 describes a proposed system for document categorization based on logo spotting while another system for document retrieval is presented in Section 4.4. Both systems are built based on our proposed framework and they are also tested on a large collection of real world documents using a well-known benchmark database of logos. This benchmark shows that our approach achieves better performances compared to state-of-the-art approaches. Finally, the discussion and conclusions are presented in Section 4.5.

4.2 Logo Spotting on Document Images

4.2.1 An Evaluation Protocol for Spotting

The measure of performance of logo spotting in particular or symbol spotting algorithms in general is quite different from the measure of Graphic Recognition techniques, since both recognition and localization have to be evaluated together. According to the evaluation protocol for the spotting system in [72], we notice the evaluation at logo level. In this case, a binary concept of retrieval whether a logo is found or not is considered. Given a logo L_i and its region P_{GT_i} represented by a polygon from the ground-truth, it will be considered as recognized if:

$$Area(P_{GT_i} \oplus P_{RS}) \geq \varepsilon * Area(P_{GT_i}) \tag{4.1}$$

where the function $Area()$ returns the area of a polygon. P_{RT} is the region represented by a polygon from the output of the system. $P_{GT_i} \oplus P_{RS}$ is the region(s) represented by a polygon (or polygons) which is the intersection of P_{GT_i} and P_{RS}. Figure 4.1 illustrates the overlapping between the results and the ground-truth.

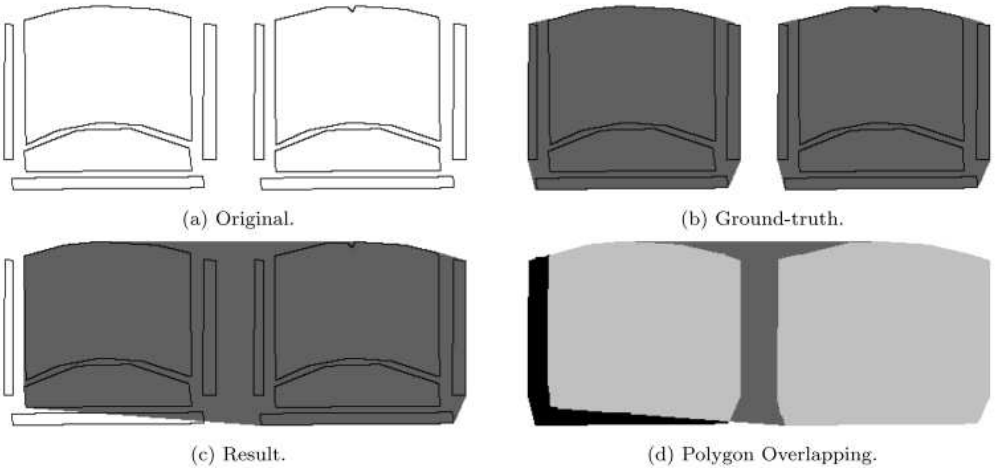


FIGURE 4.1: An example of ground-truth symbol and a result from a spotting system (extracted from [72]): (a) original image, (b) its ground-truth P_{GT} and (c) the result of spotting system P_{RS}. In the overlapping (d), $P_{GT_i} \oplus P_{RS}$ is labeled by light gray.

Equation 4.1 means that if the area of ground-truth can be overlapped with at least a certain percentage of the result area, this logo is considered as recognized. In our experiments, ε is 0.75 which means that a logo is recognized as *true positive* if it overlaps with at least a 75% its ground-truth area. Otherwise, it is considered as *false negative*. On the other hand, if the result area does not overlap with any recognized logo, it is considered as *false positive*.

4.2.2 Experimental Setup and Database

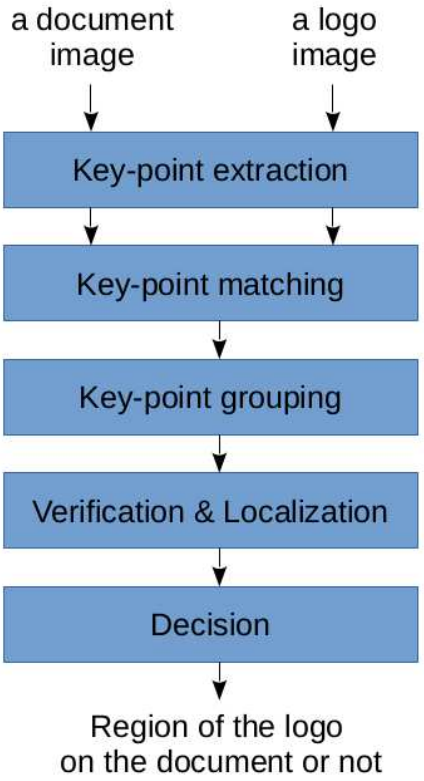


FIGURE 4.2: Overview of logo spotting system

The outline of the system is shown in Figure 4.2. It is constructed based on the proposed framework presented in Chapter 3. The input of the system is a logo image and a document image. The output of the system is a detected region likely to contain logos, or "no detection". First, sets of key-points representing both the logo image and the document image are extracted. The key-point matching methods proposed in Chapter 3 are used to find pairs of key-points between the key-point set of the query logo and the key-point set of the document. The key-point grouping step is applied to segment and group the matches. Then, the geometric verification and localization step is also used to verify the matches and localize the logo on the document. Finally, a given threshold ω based on the percentage of matches is also used to estimate if the document contains the logo or not. If the percentage of the matches is greater than ω, the document contains the logo. Otherwise, the document does not contain this logo. The evaluation protocol for spotting presented in section 4.2.1 is applied to evaluate the performance in our experiments.

In our experiments, each query image is matched with each document image in succession. In our evaluation, we ignore the document from which the query logo images was

extracted.

Six robust combinations of detectors and descriptors presented in Chapter 3 are also used in our experiments.

To provide a realistic evaluation of the performance of our system, we have collected a database and used it in our experiments. It has 100 real document images scanned from commercial advertisements. 113 logo images extracted from these documents are used as the query images. The description of this database is presented in Appendix A.

4.2.3 Experiments

Our experiments aim to compare the performance of the key-point matching methods presented in chapter 3 for logo spotting. These methods respectively correspond to key-point matching with nearest neighbor simple method (Method 1), key-point matching with 2-nearest neighbor matching rule method (Method 2) and key-point matching with post-filter by the second descriptor (Method 3). We consider the results of Method 1 as the baseline results and compare them to Methods 2 and 3.

Performance evaluation of logo spotting using key-point matching with nearest neighbor simple method

In the first experiment we use Method 1 to make the comparison between different combinations of detectors and descriptors. This method relies on the key-point nearest neighbor simple method (Method 1) because it is a simple and basic method for key-point matching. In addition, the results of this experiment will be considered as the baseline to compare to the other methods. The given threshold τ on the distance used to filter the false matches is set so that the half of matches are kept as the best matches. For density-based clustering, two parameters are considered the distance ϵ and the minimum number of points required $MinPts$. Both of them are estimated based on the height and the width of logo, and the minimum number of points for verification by homography ($\epsilon = max(height, with)/2$ and $MinPts = 4$). For the decision step, the threshold ω is set at the value of 0.02, meaning that the document contains the logo if the percentage of matches is at least 2%. Otherwise, it is considered that the document does not contain this logo. The performance is evaluated using four popular measures: *precision*, *recall*, *accuracy*, and *F1 score*. Table 4.1 shows the results of the experiment. Concerning the detector, we can ascertain some differences between the use of blob detectors (DoG, FH, MSER) and other detectors (BRISK, FAST). On the other hand, concerning the descriptor, the binary descriptors that are provided by the

difference of intensities between pairs of pixels function are less effective than others. These results are used as the baseline results to compare with other methods.

TABLE 4.1: Comparison of performance of logo spotting using key-point matching with nearest neighbor simple method on our magazine database

Detector & Descriptor	Precision	Recall	Accuracy	F1 score
DoG+SIFT	94.28%	98.40%	98.15%	96.29%
FH+SURF	97.92%	99.30%	99.29%	98.61%
MSER+SIFT	86.83%	95.20%	95.52%	90.83%
BRISK	73.05%	96.84%	92.51%	83.28%
FH+BRIEF	90.75%	87.62%	94.36%	89.16%
ORB	62.00%	99.46%	90.20%	76.38%

Comparison of performance of the combinations on key-point matching with the 2-nearest neighbor matching rule

Table 4.2 shows the results of the combinations using Method 2. We set the value of the threshold $\phi = 0.8$ for the matching method. The rest of the framework and the parameters are kept the same as in the first experiment. Four popular measures: *precision*, *recall*, *accuracy* and *F1 score* are also used to compare the performance. Compared with the results of the first experiment (the baseline results in Table 4.1), there are slight decreases in *precision* in some of the combinations; however, there are remarkable increases in *recall* of all the combinations. This proves that the quality of the matches is improved thanks to the key-point matching with the 2-nearest neighbor matching rule. Overall, we can see that the results are better in most of the combinations. Specifically, the blob detectors and the gradient orientation descriptors (DoG+SIFT and FH+SURF) give the best results.

Comparison of performance of the combinations using key-point matching with post-filtering by the second descriptor

In this part, we aim to compare the performance of the system using the key-point matching with post-filter by the second descriptor (Method 3). The main reason for this is to improve the number of the correct matches without increasing the incorrect matches, or allowing a small, acceptable, increase. In this experiment, BRIEF, a binary descriptor, is used as the second descriptor and the threshold θ is set the value of 64. We will give more details about the motivation of using BRIEF descriptor in Section 4.4. We keep all the rest of the framework and the parameters of the first experiment. To allow comparisons with the other methods, we use the *true positive rate* (TPR)

TABLE 4.2: Comparison of performance of combinations of detectors and descriptors using the 2-nearest neighbor matching rule on our magazine database

Detector & Descriptor	Precision	Recall	Accuracy	F1 score
DoG+SIFT	99.12%	99.96%	99.76%	99.54%
FH+SURF	98.23%	100%	99.54%	99.11%
MSER+SIFT	68.02%	100%	91.83%	80.97%
BRISK	56.08%	100%	88.78%	71.86%
FH+BRIEF	86.67%	100%	96.60%	92.86%
ORB	54.01%	100%	88.25%	70.14%

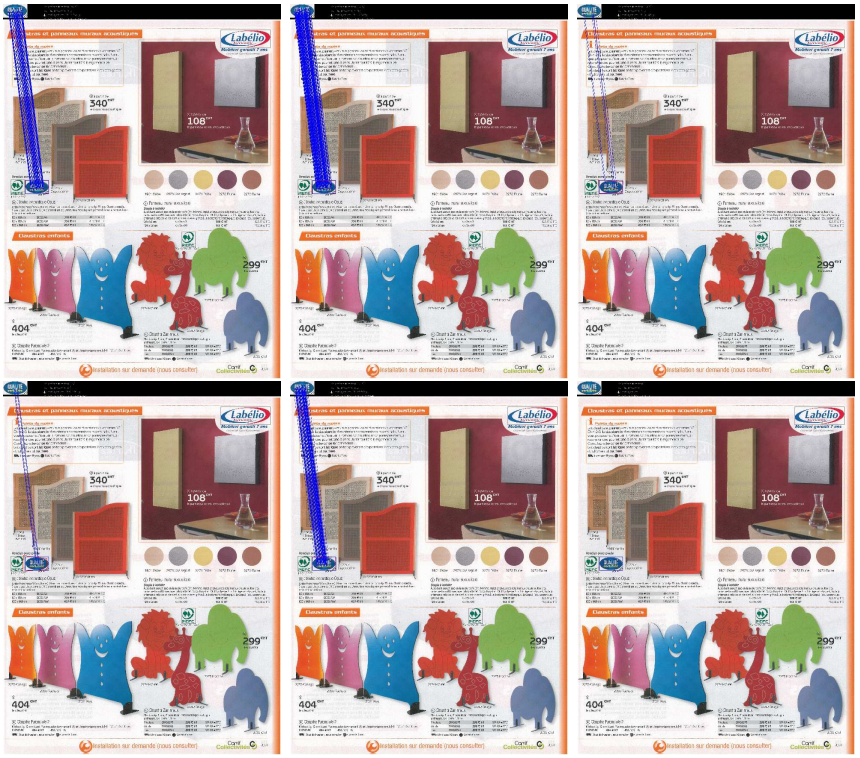


FIGURE 4.3: An example of correct matching using different combinations: row 1: DoG+SIFT, FH+SURF, MSER+SIFT; row 2: BRISK, FH+BRIEF and ORB on our magazine database. There is an incorrect matching using ORB (the last image) while DoG+SIFT, FH+SURF and FH+BRIEF give the good results with many correct matches.

(Equation 3.17), the *false positive rate* (FPR) (Equation 3.18) and the average time of each matching without key-point extraction time.

Comparing between Method 2 and Method 3, the misclassification leads the overall TPR shown in Table 4.3 to be inconsiderably lower when using the Method 3 than when using Method 2. However, when we test the documents that do not contain the same class of the query logo (or any logo) and should be classified in the true negative class, Method 3 performs better than Method 2 as shown by the FPR in Table 4.3.

TABLE 4.3: Comparison of performance of two best combinations of detectors and descriptors using Method 1 and Method 2 on our magazine database

Detector & Descriptor	**TPR**	**FPR**	**Avg. time (s)**
Method 2 and $\omega = 0.025$			
DoG+SIFT	100%	0.33%	1.20
FH+SURF	100%	0.98%	0.91
Method 3 with post-filter by BRIEF and $\omega = 0.025$			
DoG+SIFT	99.77%	0.08%	2.25
FH+SURF	99.65%	0.12%	1.89

FIGURE 4.4: An example of correct matching using SIFT+BRIEF (left) and incorrect matching using SURF+BRIEF on our magazine database.

4.3 Logo Spotting for Document Categorization

Logo spotting is of great interest to the field of document image analysis and recognition. It consists of matching a set of query document images (documents to be categorized) with a set of known logos (logo gallery). The matching pairs of key-points between each logo in the gallery and the query document images are estimated. Then, the query document is assigned to the category corresponding to the logo having the maximum proportion of matching key-points. In this section, we present the system of document categorization using logo spotting. According to the results of our experiments in the previous chapter, our system uses the DoG+SIFT descriptor and the key-point matching with the 2-nearest neighbor matching rule method.

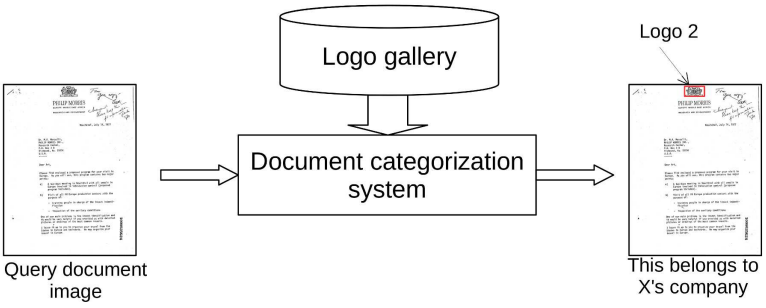


FIGURE 4.5: Logo spotting applied to document categorization applications.

4.3.1 The Proposed System for Document Categorization

Figure 4.6 illustrates the outline of our system based on the framework presented in Chapter 3. After removing noise from the document image by preprocessing based on morphological operators, the interest points of the query document image and logo images are extracted and described using DoG+SIFT descriptor. Then, the key-points of the query document image are matched towards the key-points of all the logo images in the gallery. The logo in the query document is segmented by clustering the matching key-points using a density-based clustering algorithm, and the logo category is determined thanks to an accumulating histogram.

Feature Extraction

SIFT descriptor is widely used for describing interest key-points, because it is invariant towards scaling, rotation and partially invariant to affine transform. The interest key-points are detected by DoG (Difference of Gaussian) filter at different scales, then they are described by the SIFT feature vector. Each SIFT feature vector is characterized by a 4x4 matrix of orientations of the intensity gradient in the sixteen 4x4 windows around the considered key-point. The orientation being quantized over eight values, the

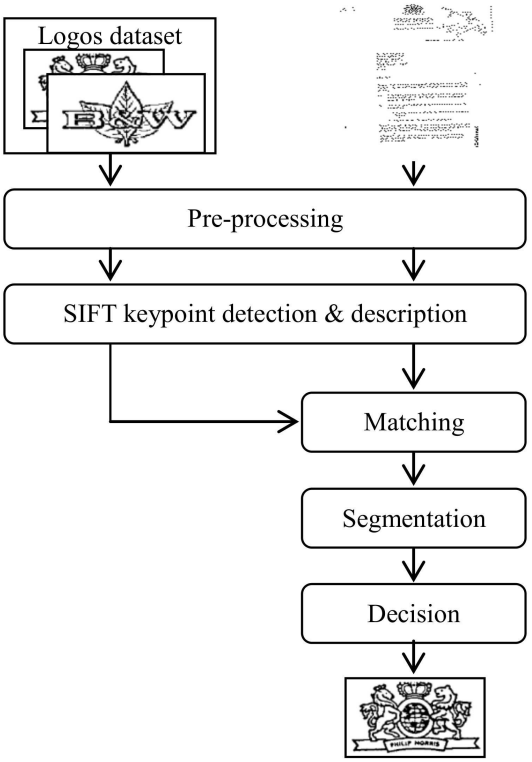


FIGURE 4.6: The outline of document categorization approach.

resulting SIFT feature vector is 128-dimensional, $S_i = (s_{i1}, .., s_{i128})$ (Figure 2.11). In addition to these 128 features, the x- and y-position of the key-points are also used for logo spotting.

Matching

Matching is performed by considering, for each key-point in each logo image, its two nearest neighbors within a given query document image in the SIFT feature space. Equation 3.5 can be rewritten as follows:

$$\begin{aligned} d_1 &= \min_k(\|S_q - S_k\|) \qquad \text{and } k^* = argmin_k(\|S_q - S_k\|) \\ d_2 &= min_{k \neq k^*}(\|S_q - S_k\|) \end{aligned} \tag{4.2}$$

where $(\|S_q - S_k\|)$ is the Euclidean distance between two key-point SIFT descriptors q and k. The ratio r of d_1 over d_2 is then used for matching. If r is greater than a given threshold ϕ, then it means that the matching is not reliable, as there is a possible ambiguity between the two nearest neighbors. On the other hand, if r is lower than ϕ, then the key-point is representative enough to be considered. In practice, we set the value of the threshold ϕ to 0.6, based on experiments.

Segmentation: Grouping the matched key-points

As discussed above in Section 3.5.1, not all the matching key-points are located inside the logo region. Some of them are located in other regions of the document image. We consider these key-points as noise, and we do not want them to be taken into account in our final decision. In addition, there is a high concentration of matched key-points on the correct logo. DBSCAN [60] define a cluster based on the notion of density reach-ability. Basically, a point belongs to a group if it is within a certain distance ϵ from any point of this group. A group forms a cluster if it has more than $MinPts$ points, otherwise it is noise. DBSCAN can determine the number of clusters automatically (as opposed to the basic version of k-means for instance), and it can find arbitrarily shaped clusters (while the clusters provided by k-means are circular). Here, we use the Euclidean distance to compare the positions of the key-points $(x, y) \in R^2$ and cluster them using DBSCAN. In practice, the values of ϵ and $MinPts$ affect the clustering result and may be set by experiments. We therefore obtain clusters of matched key-points as an output of the segmentation stage.

Geometric verification based on Homography using RANSAC

We apply the algorithm proposed in Section 3.6.2 on each the clusters (determined by DBSCAN). The contribution of this section is to add to our approach an additional step in which the pairs of incorrectly matched key-points are filtered based on homography using RANSAC. This will reduce the incorrect matches and provide better results in the spotting system. This will be proved in the experiments.

Decision based on the number of matched key-points

For decision, we use an accumulating histogram to count the number of matches in the biggest cluster (determined by using DBSCAN). The matching logo is finally determined by searching the maximum m of the accumulating histogram H after normalizing each cell with the number of key-points in the corresponding logo. If m is equal or greater than a threshold ω, it means that the document image contains this logo; otherwise, it does not contain any logo from the gallery (no logo at all).

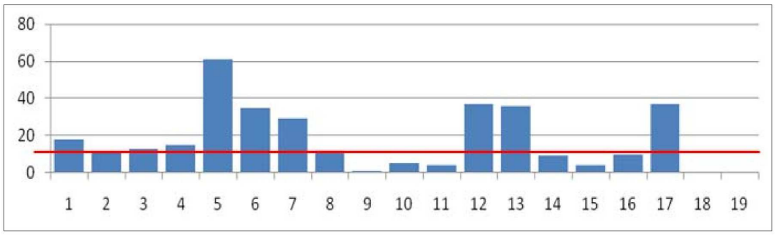


FIGURE 4.7: The accumulating histogram of the number of matches. Red lines show the threshold ω. In this case, our algorithm decides that the document contains logo number 5.

4.3.2 Experiments

Comparison with state-of-the-art methods in logo detection

In the first experiment, we compare the precision and accuracy measures of our approach with different state-of-the-art methods for logo detection on the Tobacco-800 database. We consider all of the documents containing a gallery logo as positive and those not containing logos from the gallery as negative. We compare our approach with those of Zhu and Doerman [34], Li el al. [69], and Pham et al. [31]. However, our algorithm relies on matching logos in the query document and in the gallery, so we can only use the subset of logos from the Tobacco-800 database which have at least two occurrences (which represents 19 logos, 378 query documents containing logos and all the 878 document images without any logo), while other authors in Table 4.4 use the whole Tobacco-800 database (which represents 412 query documents containing 35 different logos and all the 878 document images without any logo). In addition, we also evaluate the performance of our approach with and without the density-based clustering process. The comparison results presented in Table 4.4 show that our approach with the density-based clustering process is more accurate than previous methods, while precision remains very satisfying. However, our method is conceived for logo recognition more than logo detection, so we use a gallery of logos for matching, which is not the case of other methods. Therefore, we note that a fair comparison cannot be made because each of the methods in the table uses different number of documents for testing.

TABLE 4.4: Comparison of our approach with state-of-the-art methods for logo detection using Tobacco-800 database. Because of the requirements of our method, we use only a subset of this database (34 documents are used for constituting our logo gallery)

Approaches	Accuracy	Precision
G. Zhu and D. Doerman [34]	84.2%	73.5%
Zhe Li et al. [69]	86.5%	99.4%
Pham et al. [31]	91%	85%
Our appoach without DBSCAN	82.22%	80.72%
Our appoach with DBSCAN	**94.04%**	**91.11%**

**Note that a fair comparison cannot be made.*

Logo spotting for document categorization

In the second experiment, we consider the 15 logos from the Tobacco-800 database contained in at least 3 document images, coming up in a total with 374 document images containing logos and all the 878 document images without any logo.

TABLE 4.5: Confusion matrix corresponding to document categorization on Tobacco-800 database (the ground-truth is in the rows).

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	**2**														
2		**59**													
3			**49**	1							1				
4			1	**43**											
5	1				**17**										
6						**67**			1				1	2	
7							**17**								
8								**8**							
9			1						**5**		1				
10										**32**					
11											**5**				
12												**9**			
13													**4**		
14														**5**	
15															**3**

As a result of the first experiment, Table 4.5 shows the confusion matrix between the 15 logo classes. We first have to consider that the false negative rate (cases where a known logo is missed) is 10.42% and the false positive rate (cases where a document with no logo is matched to a gallery logo) is 3.75%. Second, we consider the recognition of each class. The average classification rate is 86.90%. However, it should be noticed that some classes such as the fifth and ninth classes achieve the lowest accuracies of 61% and 56%, respectively (see Table 4.5). This may be explained by the fact that, in both cases, the size of the logo image in the gallery is small; and therefore, only a small number of key-points can be considered for matching and classification.

Comparison between the methods with and without re-filtering with homography

According to the first experiment, we consider the same sub-dataset. We compare the recognition rates we obtain with the approach once with, and once without, the re-filtering step. The results shows an improvement, as the accuracy reaches 88.77%, instead of 86.90% (see Table 4.6).

TABLE 4.6: Performance evaluation and comparison for logo recognition on Tobacco-800 database.

Approach	**Accuracy**
Our approach without re-filtering	86.90%
Our appoach with re-filtering	88.77%

The next experiment is dedicated to logo detection. We compare the precision and accuracy measures of the approaches with and without the re-filtering step, and other

state-of-the-art methods. Please note that this comparison has to be considered carefully, as in our approach we know the logo gallery, which is not generally the case for pure logo detection approaches. Table 4.7 shows an increase in Accuracy and Precision thanks to the filter by homography using RANSAC. However, the recall drops slightly of 0.52% because this filter also reduces the number of the matched key-points in each cluster for the decision step. As a result, a few document images containing logos are incorrectly classified as containing no logos.

TABLE 4.7: Performance evaluation and comparison for logo detection on Tobacco-800 database.

Approaches	Accuracy	Precision	Recall
G. Zhu and D. Doerman [34]	84.2%	73.5%	-
Z. Li et al. [69]	86.5%	99.4%	-
T-A. Pham et al [31]	91%	85%	-
Our method without re-filter	94.04%	91.11%	88.94%
Our method with re-filter	**95.86%**	**97.67%**	**88.42%**

4.4 Logo Spotting for Document Retrieval

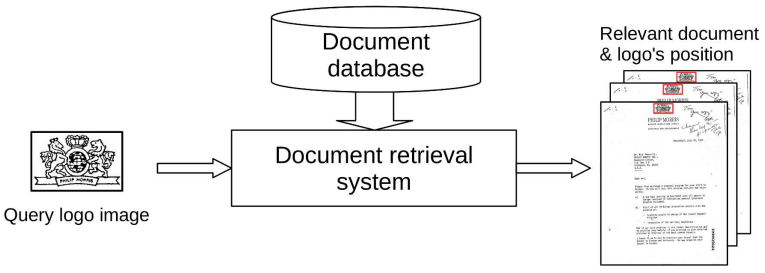


FIGURE 4.8: Logo spotting applied to document retrieval applications.

4.4.1 Using BRIEF Descriptor for Post-filtering

Although key-point matching with the 2-nearest neighbor matching rule can give the best candidate pairs of key-points, there are many correct matches that are rejected by this method. According to the key-point matching methods presented in chapter 3, we use the key-point matching with post-filtering by BRIEF descriptor in this system.

Key-points are rejected by the 2-nearest neighbor matching rule when the first and second nearest neighbor are not far enough in SIFT feature space to be concerned. Thus, our hypothesis is that in other feature spaces, the rejected key-points can be matched. BRIEF is fast and achieves approximately similar recognition performance compared to other key-point descriptors. It uses the intensity of pairs of pixels directly for description, while SIFT descriptor is computed based on the gradient orientations in sixteen 4x4 windows around a key-point. In addition, using the hamming distance in matching with BRIEF descriptor produces the number of positions at which the corresponding bits are different. This means that we estimate the different intensities of corresponding pixels between two patches of pixels around key-points. These differences allow a second phase of filtering the matches based on additional criteria other than the SIFT- based nearest neighbor matching rule. This in the end improves the matching performance. In our proposed method, BRIEF descriptor is employed to post-filtering the key-points rejected by the nearest neighbor matching rule with the hope of increasing the number of matching key-points without decreasing their accuracy. Figure 4.9 shows an example of correct spotting thanks to the post-filtering by BRIEF descriptor. In the next section, we will explain the method with SIFT and BRIEF descriptor in detail.

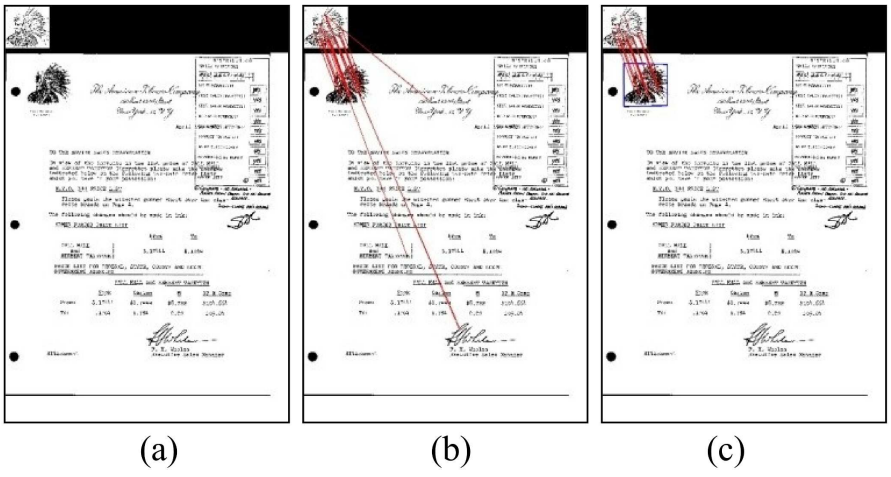


FIGURE 4.9: An example of correct spotting thanks to the post-filtering by BRIEF descriptor. (a) No key-point is matched because all the key-points are rejected by the 2-nearest neighbor matching rule with SIFT. (b) Many key-points are matched after post-filtering based on BRIEF descriptor and then (c) the spotting is correct of the case (b).

4.4.2 Outline of the Proposed System

Our approach is based on matching pairs of key-points between the query logo image and each document in the database. The key-point matching step has a two-stage filtering with SIFT descriptor and BRIEF descriptor. After that, the matched key-points are

grouped by a density-clustering method and are filtered by our geometric verification method. Finally, each document is ranked by the proportion of matched key-points.

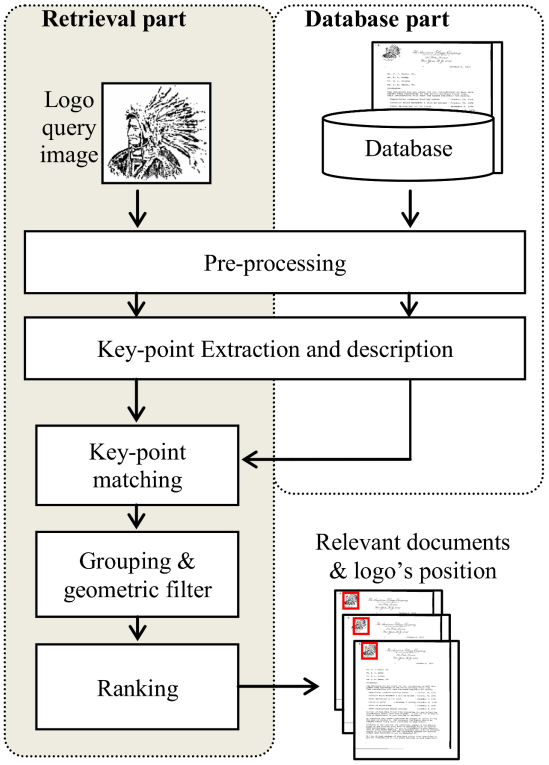


FIGURE 4.10: Outline of proposed document retrieval system based on logo spotting.

Figure 4.10 shows the outline of our proposed system. The first step is preprocessing using morphological operators to reduce noise. The interest points of the query logo image and the document image are extracted and described by SIFT and BRIEF descriptors. In the third step - key-point matching step, key- points are matched in the SIFT feature space using the nearest neighbor matching rule and then filtered using BRIEF descriptor. Like our previous system in document categorization, segmentation by density-clustering and post-filtering by homography are used in the fourth step. Finally, each document image is ranked based on the number of matched key-points after normalization.

Feature Extraction

In our approach, the local features are used to describe logo query images and document images in order to match the query and all documents in database. The interest key-points are detected by DoG (Difference of Gaussians) filter at different scales, and then they are described by the two descriptors SIFT and BRIEF.

A given query $\mathbf{Q}$ with n key-points and a document $\mathbf{T}$ with m key-points are described as follows:

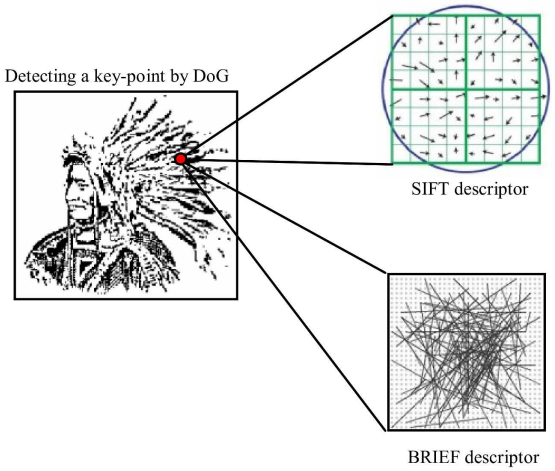


FIGURE 4.11: Each key-point is described by SIFT descriptor and BRIEF descriptor.

$$\begin{aligned} \mathbf{Q} &= (x_j, y_j, S_j, B_j) \qquad \text{with } j = 1..n \\ \mathbf{T} &= (x_j, y_j, S_j, B_j) \qquad \text{with } j = 1..m \end{aligned} \tag{4.3}$$

where x_k and y_k are the x- and y-position of k^{th} key-point. S_k and B_k are the SIFT and BRIEF descriptor of this key-point, respectively.

Key-point Matching with post-filtering by BRIEF descriptor

As presented in Chapter 3, in this section, we present the key-point matching with post-filtering method by SIFT and BRIEF as a second descriptor. Our matching key-point step has two stages. Firstly, the 2-nearest neighbor matching rule is applied to find the matches in SIFT descriptor space. Then, BRIEF descriptor is used to post-filter the matched key-points that are rejected by the 2-nearest neighbor matching rule. The matching between each key-point $q = (x_q, y_q, S_q, B_q)$ of $\mathbf{Q}$ and all key-point of $\mathbf{T}$ are computed as in the two following stages:

In the first stage, find the k nearest neighbors $t_i \in \mathbf{T}$, $i = 1..k$ of query feature q (k=5 in our experiments). Let d_i represent the distances between document features t_i and query feature q in the SIFT feature space with $i = 1..k$ so that $d_1 \leq d_2 \leq ... \leq d_k$. We consider the two nearest neighbors t_1 and t_2 with their distances d_1 and d_2:

$$\begin{aligned} d_1 &= \min_{j=1..m} \{dist_E(S_q, S_{t_j})\} \\ d_2 &= \min_{j=1..m; t_j \neq t_1} \{dist_E(S_q, S_{t_j})\} \\ &\quad \text{with } t_1 = argmin_{j=1..m}\{dist_E(S_q, S_{t_j})\} \end{aligned} \tag{4.4}$$

where $dist_E(S_q, S_{t_j})$ is the Euclidean distance between the two key-points q and t_j in the SIFT descriptor space. If the ratio of d_1 and d_2 is lower than a threshold ϕ (in

practice we use $\phi = 0.6$), then the pair of key-points (q, t_1) is representative enough to be considered; and this pair is pushed into a good match list. Otherwise, this pair is moved to the second stage.

In the second stage, we try to find the best key-point within the k nearest neighbors $(t_1, t_2, .., t_k)$ based on BRIEF descriptor. The hamming distances $h_1, h_2, .., h_k$ of the pairs $(q, t_1), (q, t_2), .., (q, t_k)$ are computed in BRIEF descriptor space. We consider h_min and t_min which are denoted by:

$$\begin{aligned} d_{min} &= \min_{i=1..k} \{dist(B_q, B_{t_i})\} \\ t_{min} &= \operatorname*{argmin}_{i=1..k} \{dist(B_q, B_{t_i})\} \end{aligned} \tag{4.5}$$

where $dist(B_q, B_j)$ is the hamming distance between the two key-points q and t_i in the BRIEF descriptor space. Then, a threshold θ is also used to reject the key-points which are too far from the query key-point. In practice, the threshold $\theta = 64$ is chosen, meaning that if the difference between two descriptors of key-point in BRIEF descriptor space is greater than 25%, the key-point is rejected.

Grouping and Geometric Filter

As discussed above, there is a high concentration of matched key-points on the correct logo region and the concentration decreases when considering an incorrect logo, or when the document does not contain any logo. We consider key-points located outside of the logo region as noise, and we want to reject them. For this purpose, we use a density-based clustering method to group key-points into clusters as logo candidate regions and to reject noisy key-points. DBSCAN, a density-clustering method, is used. It can determine the number of clusters automatically (as opposed to the basic version of k- means for instance), and it can find arbitrarily shaped clusters (while the clusters provided by k-means are circular). In this experiment, we reused the values of the DBSCAN parameters: ϵ=60 and MinPts=5.

Even after the candidate regions are estimated by DBSCAN, not all the matched pairs of key-points in each candidate region are correct, because the density-based clustering method does not integrate enough information concerning the spatial distribution of the matched key-points. Incorrect matches (inconsistent with the spatial distribution of the key-points in the logo) need to be rejected before the next step. According to the two-step method which is proposed based on homography using RANSAC, a transformation between the pairs of matched key-points in each candidate region is computed to integrate spatial relationships between the key-points in the query image

and those in the candidate region. To solve the cases where a cluster region may not cover the logo region but just a part of the logo region, the transformation is computed again after we have determined all matched key-points in a bounding box covering the logo region candidate.

Ranking

For ranking, we use two measures based on counting the number of the matched key-points in candidate regions after and before having been filtered by the geometric filter step. These measures are normalized by the number of key-points in the query image, as follows:

$$m_1 = \frac{N_1}{\#\text{ key-points of query}} \tag{4.6}$$

$$m_2 = \frac{N_2}{\#\text{ key-points of query}} \tag{4.7}$$

where N_1 is the highest number of the matched key-points in all candidate regions after the geometric filter step and N_2 is the highest number of the matched key-points in all candidate regions before the geometric filter step.

The retrieved documents are ranked by the value of m_1 and then by m_2. The second distance m_2 aims at solving the cases where the first distance of the two documents is equal and/or all key-points are rejected by the geometric filter.

4.4.3 Evaluation Metrics

Average precision and R-precision are the two most commonly used measures to evaluate performance in information retrieval. Average Precision (AP) is a value combining precision and recall while R-Precision (RP) is the precision at R-th position in the ranking table for a query that has R relevant documents. AP can be calculated by:

$$AP = \frac{\sum_{k=1}^{n}\{P(k) \times rel(k)\}}{\#\text{ relevant documents}} \tag{4.8}$$

where n is the number of retrieved documents, $P(k)$ is the precision at rank k, $rel(k)$ is 1 if the result at rank k is a relevant document, and otherwise 0.

The overall system performance across all queries is measured using Mean Average Precision (MAP) and Mean R-Precision (MRP).

$$MAP = \frac{\sum_{q=1}^{Q} AP(q)}{Q} \tag{4.9}$$

$$MRP = \frac{\sum_{q=1}^{Q} RP(q)}{Q} \tag{4.10}$$

where Q is the number of queries, $AP(q)$ is the average precision of query q, $RP(q)$ is the R-precision of query q.

To compare with state-of-the-art methods, we also use mean average precision per class (MAPc). MAPc is the mean average precision across all classes of queries while MAP is mean average precision across all queries.

4.4.4 Experiments

In the first experiment, we would like to compare the accuracy and the number of matched key-points between the method with the nearest neighbor matching rule only and the method combined with BRIEF descriptor. We use 432 queries and 412 documents containing logo(s) of Tobacco-800 database for testing the accuracy of matched key-points and the number of correctly matched key-points with ϕ from 0.1 to 0.9. The correctly matched key-points are estimated by counting the matched key-points within a bounding box thanks to the ground-truth. The accuracy is the ratio of the number of correctly matched key-points to the number of the matched key-points. The result shows that the number of the correctly matched key-points increases but the accuracy decreases when ϕ approaches to 1. Otherwise, the number of the correctly matched key-points decreases but the accuracy increases when ϕ approaches to 0. As seen in Figure 4.12, at the same accuracy level, there is an increase in the number of correctly matched key-points of the method combined with BRIEF descriptor compared to the method using the nearest neighbor matching rule only.

In the second experiment, we compare the performance of different methods in combination with different key-point matching rules (the rest of the framework is the same). The first method is a simple nearest neighbor matching using the nearest neighbor key-point as the best matched key-point. The second method is similar to our previous method in [8] using the nearest neighbor matching rule with the two nearest neighbors to filter the matched key-points. The third method, is the proposed method, which combines the nearest neighbor matching rule with the two nearest neighbors and BRIEF descriptor to filter the matched key-points as presented in section 4.4.1. We also test on 412 documents containing logo(s) and on all 1290 documents in database. We observe that the

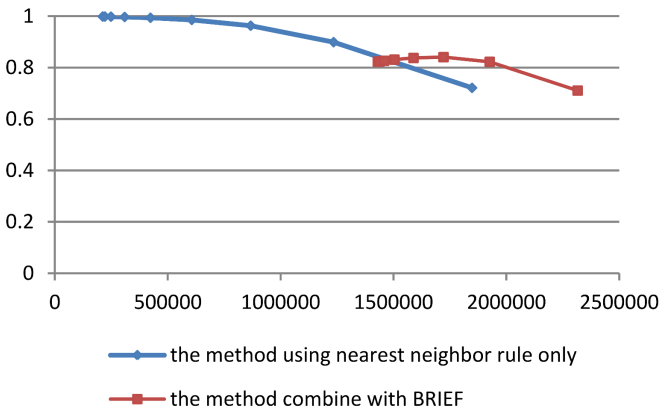


FIGURE 4.12: Comparison of the two methods on the number (x-axis) and the accuracy (y-axis) of the matched key-points. There is an increase in the number of matched key-points of the method combined with BRIEF at the same accuracy level.

number of correct matched key-points of the method combined with BRIEF descriptor increases 1.49 times more than that by the method without BRIEF descriptor filter.

Figure 4.13 shows some examples of the results. Table 4.8 summarizes the results of this experiment. As we can see, the result of the third method we propose in this section is better than the other two we tested.

TABLE 4.8: Comparison of performance of different key-point matching methods using Tobacco-800 database on document retrieval: Method 1 - simple nearest neighbor; Method 2 - the two nearest neighbor rule; Method 3 - Two nearest neighbor rule with post-filtering by BRIEF.

Methods	412 documents containing logo(s)		All 1290 documents	
	MAP	MRP	MAP	MRP
Method 1	85.21%	82.74%	81.62%	80.05%
Method 2	90.14%	87.88%	88.31%	85.87%
Method 3	**93.13%**	**90.61%**	**91.15%**	**88.78%**

In the final experiment, we compare MAP, MRP and MAPc of our approach with different results reported from state-of-the-art methods using all 1290 documents in the database. Table 4.9 shows that our approach with a post-filtering based on BRIEF descriptor is higher than those by other methods. The results in the table for state-of-the-art methods are taken from their respective papers [36, 70, 71].

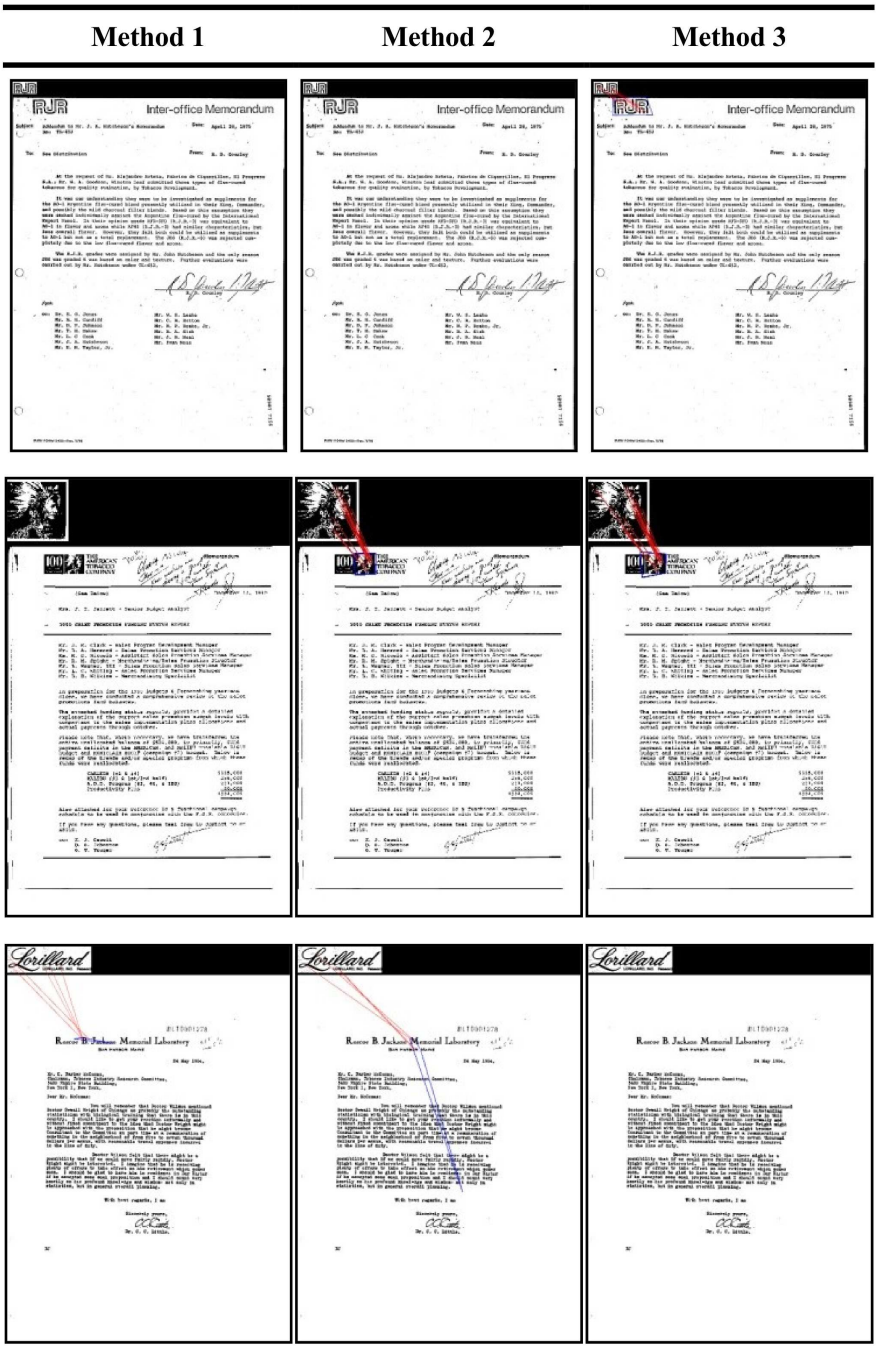


FIGURE 4.13: Examples of the result comparison between methods.
Row 1: a correct matching of method 3;
Row 2: an incorrect matching of method 1;
Row 3: incorrect matchings of both method 1 and 2.

TABLE 4.9: Comparison of performance between our method and state-of-the-art methods using Tobacco-800 database on document retrieval

Methods	MAP	MRP	MAPc
Zhu et al [71]	82.6%	78.5%	-
Rusinol et al [70]	82.6%	-	-
Jain et al [36]	88%	-	87%
Our method without post-filtering	88.31%	85.87%	91.97%
Our method with post-filtering	**91.15%**	**88.78%**	**94.14%**

**The results in the table for state-of-the-art methods are taken from their respective papers.*

4.5 Discussion

In this chapter we have presented an approach for logo spotting using key-point matching and tested it on real documents and logo images. We used local features to describe the query document images and the logos in the gallery. Interest points are extracted and described using a local detector and descriptor. Pairs of key-points are determined by key-point matching methods in each logo. To filter the matches outside the logo region and to segment the logo region, we apply segmentation using density-based clustering of the logo key-points. Geometric verification and localization are done using the proposed two-stage method based on homography with RANSAC. The evaluation protocol for logo spotting was also discussed in this chapter. To compare to the state-of-the-art methods, we built two systems for document categorization and retrieval. These systems were tested on the well-known benchmark database of real world documents containing logos. The results showed that our approach achieves better performance than the state-of-the-art methods.

The first conclusion can be drawn from the comparison between detectors and descriptors. According to the experiments, the combinations of blob detectors and spectra descriptions may be the best choice for logo spotting on the document images. All our experiments indicate that DoG+SIFT feature is a good representation for many types of logo.

Secondly, we propose the use of the density-based clustering to group the matches into the logo spotting framework. This can not only segment the candidate logo region but also reject the incorrect matches which are outliers. Besides filtering the incorrect matches, the proposed algorithm for geometric verification with two steps based on homography with RANSAC aims at solving cases where a cluster region may not cover the logo region, but just a part of the logo region (for an example see Figure 3.10).

Moreover, we believe that this can apply into other frameworks in the graphical spotting field such as symbol spotting.

In addition, overall, the use of key-point matching with post-filtering by BRIEF as a second descriptor gives better results than the others. However, it should be preferred in applications where correct identification of the incoming documents is of more importance than the false alarms (documents which do not contain any logo). On the other hand, if we expect a faster method or intend to minimize the false alarms which may include misclassified documents, the key-point matching with 2-nearest neighbor matching rule should be considered.

Finally, the computation cost problem should be considered in key-point matching with local features. The computation time tends to be high when the dimensions of feature vectors increase. The SIFT descriptor vector has 128 dimensions whereas the length of SURF descriptor vector is 64. The high dimensional feature space can help to be discriminant enough between key-points, but the computation time increases correspondingly. In the next part of this thesis, to optimize the logo spotting system, text/graphic separation as well as the dimension reduction are studied.

Part II

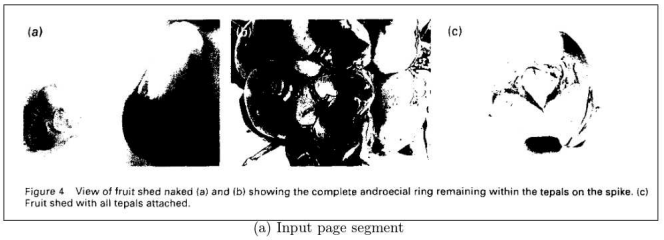
Logo Spotting Optimization based on Text/Graphics Separation

Chapter 5

Text/Graphics Separation

5.1 Introduction

Separating text and non-text in a document is an important layout analysis step in the Document Image Analysis and Recognition (DIAR) field. Such a step improves the accuracy rate as well as the running time within the OCR process or in some other DIAR tasks. Figure 5.1 shows an example of an incorrect OCR result of a segmented zone containing both text and graphics. Separating text and non-text is also useful for information spotting tasks in documents such as symbol spotting, word spotting, logo spotting, and so on, because it is easier and faster to perform spotting if the text and non-text are well segmented. Non-text in documents can be one of the following categories: halftone, drawing, math, logo, table, chart, separator, etc.



(a) Input page segment

tv _ . - l 4
la) ` `)åo«. , ` (C)
Â» ` fr ⁄ .1 Â·r , ` WM. åo«ý
7 if Â¢åo ÂŸ, ` ` _` { Ã©>Ã©_ j l; I X I I
K;~Â· - _ V . Â· ` _ ~ M" JF Q} { fÂ'¸; Ai ' V _ I \' g,
åo _ it ~ _y, i * t
f jgj . __ it ` l åo«ý f
1< .Â· åo«ý Â·. V Â·- åo«Â¢ åo«. åo«. åo«ý F {U
Â·~ . 7 ~ Ã© . i⁄ ` I / _åo«ý * åo«. K Â»
Figure 4 View of fruit shed naked la) and lb) showing the complete androecial ring remaining within the tepals on the spike. (c) Fruit shed with all tepals attached.

(b) OCR result

FIGURE 5.1: An incorrect OCR result of a segmented zone containing both text and graphics (extracted from [1])

In the key-point-based approaches, the interest points are detected and typically described by high dimensional feature vectors. For each key-point of the query image, these approaches find its best candidate match in the document image. However, it is often too costly to deal with high dimensional feature and large number of key-points in searching space. In this chapter, we deal with the problem of text and non-text separation because, for us, it is a preprocessing step and our final goal is making logo spotting more efficient and more accurate. The remainder of this chapter is structured as follows: an overview of the text/graphics separation approaches is presented in the next section. Section 5.3 focuses on connected component features. In Section 5.4, we detail the proposed method for separating text and non-text on documents using connected component features, discuss the evaluation protocol for our method on connected component-level, and pixel-level and describe the experiments. Finally, Section 5.5 is for the discussion and conclusion.

5.2 Related Work in Text/Graphics Separation

Text and non-text segmentation approaches can generally be classified into three groups: (i) region (or block or zone) based segmentation [73, 74], (ii) pixel based segmentation [75, 76], and (iii) connected component based segmentation [1, 77, 78].

In region-based segmentation approaches, a page segmentation step on the document image is first carried out to get document zones, and then a learning and classification step is applied to classify those zones [79]. Methods of this type of approach rely greatly on the accuracy of the segmentation step, which can be challenging in less-structured documents of complex layout. In pixel-based approaches, the classification is applied on each pixel in a document. Methods which follow this approach tend to be sensitive to noise and are time consuming. The connected component-based approaches, on the other hand, are independent of block (zone) segmentation step, and are more robust thanks to considering the connected component level to discriminate between text and non-text.

The review by Okun et al. [74] details the approaches of page segmentation and zone classification, and reviews the main approaches used for document segmentation and region classification in the 1990s. Run Length Smearing Algorithm (RLSA), presented by Wong et al. [73], is one of the earliest approaches using region-based segmentation. RLSA analyses the spaces between black pixels in order to merge characters into blocks and then uses some basic features to classify each block. Lin et al. [80] proposed a region-based approach. They used different texture-based GLCM (Grey Level Co-occurrence Matrix) features to divide a document into blocks of graphics, text and space zones, and

used K-means for clustering the blocks into zones. They then used pre-learned heuristic rules for zone classification.

In connected component based segmentation, there are some notable approaches. Fletcher and Kasturi [77] proposed a method based on Hough transform to group connected components into a text string and then classify them by using an analysis method. Tombre et al. [78] developed a consolidation of the method proposed by Fletcher and Kasturi, with a number of improvements in the analysis method.

Bukhari et al. [1] have presented a method which is more related to the method that we propose here. They used shape and context information of each connected component as a feature vector and then discriminate them into text and non-text classes by a self-tunable multi-layer perceptron (MLP) classifier. Each connected component and its surrounding context area are rescaled to 40×40 pixel window size for a 3204-dimension vector including four other features based on the connected component's size. The other work of Bukhari [3] presents improvements to the text/image segmentation algorithm described by Bloomberg [81]. A hole-filling morphological operation and reconstruction of broken drawing lines are added into Bloomberg's algorithm. These improvements aim to solve cases where non-text components do not contain any solid bunch of pixels.

5.3 Connected Component Features

5.3.1 Connected component detection

Connected component (CC) labeling operation works by scanning a binarized image pixel-by-pixel from top to bottom and left to right in order to identify connected pixel regions [82]. The operation scans the image by moving along a row until it comes to a foreground pixel p. When this is true, the four neighbors of p which have already been met during scanning are examined. The four neighbors of p are the neighbors to the left of p, above it, top-left neighbor and top-right neighbor in a 3×3 window size. Based on this information, the labeling of p occurs as follows:

- If all four neighbors are background pixels, set a new label to p, else
- If only one neighbor is a foreground pixel, assign its label to p, else
- If more than one of the neighbors are foreground pixels, assign one of the labels to p and make a note of the equivalences.

After completing the scan following the algorithm above, the equivalent label pairs are sorted into equivalence classes and a unique label is assigned to each class. Then, a

second scan is made through the image, during which each label is replaced by the label assigned to its equivalence classes.

5.3.2 Feature Extraction

Connected component features

Connected components are classified through their characteristics represented by features. Therefore, it is important to find features which have the ability to well discriminate the different visual characteristics of connected components. In this subsection, we present connected component's features clustered into three groups: simple geometric features, moment features, and stroke width features.

Simple geometrical features can be very effective in many applications [82]. Examples of these features are the area, perimeter (number of pixels in the boundary of the connected component), aspect ratio, compactness ($perimeter^2/area$), etc. Among these features, we specially consider two features: *elongation* and *solidity*. Elongation is the height to width ratio of a rotated minimal bounding box of the connected component (see Figure 5.2). It represents the square of connected component which has differences between texts and lines. Solidity is the number of black pixels divided by the area of the bounding box. It is selected for the reason that tables, borders, and many graphical drawings have many differences in solidity compared to texts.

$$Elongation = \frac{min(height, width)}{max(height, width)} \tag{5.1}$$

$$Solidity = \frac{\#\_black\_pixels\_of\_component}{area\_of\_bounding\_box} \tag{5.2}$$

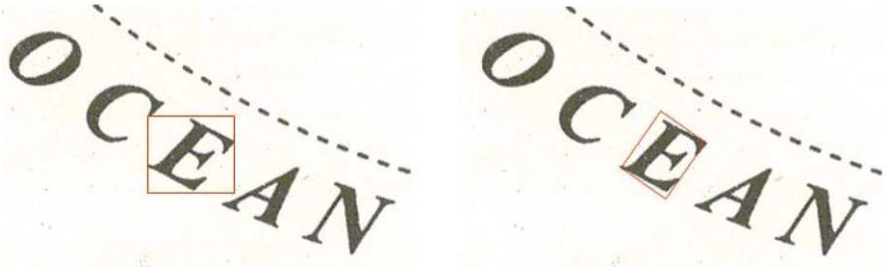


FIGURE 5.2: An axis-aligned bounding box (left) and a rotated minimal bounding box (right) of a connected component. (extracted from [2])

Moment features are classified into the shape-based description. Hu moments is a representative of this type of feature. Hu [83] defined seven moment invariants from geometric

moments. These moments are invariant to translation, rotation and scaling. We first start with the definition of a 2-dimensional $(p+q)^{th}$ order general geometric moment of an $M \times N$ image I.

$$m_{pq} = \sum_x \sum_y x^p y^q I(x,y) \tag{5.3}$$

However, the moments in Equation 5.3 may be not invariant to translation, rotation and scaling. Central moments provide the invariant features which are defined as follows:

$$\mu_{pq} = \sum_x \sum_y (x-\bar{x})^p (y-\bar{y})^q I(x,y) \tag{5.4}$$

where $(\bar{x}, \bar{y})$ is the centroid of the image I.

$$\bar{x} = \frac{m_{10}}{m_{00}}, \quad \bar{y} = \frac{m_{01}}{m_{00}} \tag{5.5}$$

Using the centroid of image has made the central moment invariant to image translation. Scale invariant can be achieved by normalization. The normalized central moments are defined as follows:

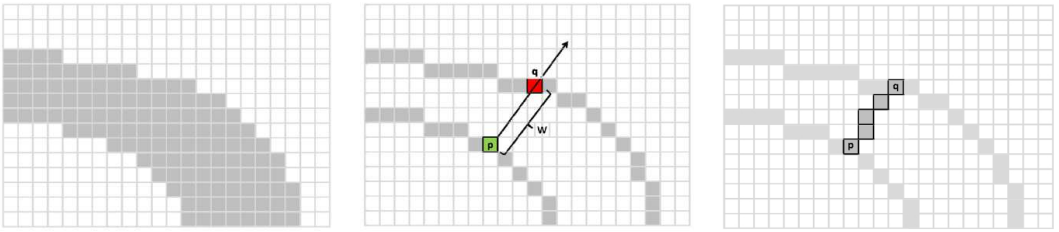
$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{(1+\frac{p+q}{2})}} \tag{5.6}$$

Finally, the seven Hu moments are calculated as follows:

$$\begin{aligned}
H_1 = {} & \eta_{20} + \eta_{02} \\
H_2 = {} & (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
H_3 = {} & (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
H_4 = {} & (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\
H_5 = {} & (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\
& + (3\eta_{21} - \eta_{03})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
H_6 = {} & (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
& + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \\
H_7 = {} & (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\
& - (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]
\end{aligned} \tag{5.7}$$

Stroke width features are often used in text detection or separation because text has an almost similar stroke width while non-text does not. Many researchers used these features in their methods as in [84–86] and [87]. Yin et al. [87] proposed a method to calculate the stroke width based on computing the distance from an edge pixel to its opposite edge pixel. First, edge pixels were detected using Canny edge detector [88]. For each edge pixel p, its opposite edge pixel q is found based on following the ray $r = p + nd_p$ where d_p is the gradient direction of p. At the pixel q with the gradient direction d_q, if $(d_q = -d_p \pm \pi/6)$, then each element along the segment $[p, q]$ is set by the Euclidean

distance $\|p - q\|$ unless it already has a lower value. Otherwise, the ray is rejected if q is not found or $(d_q \neq -d_p \pm \pi/6)$ (see [87] for more information). Figure 5.3 shows the process of stroke width computation.



(a) (b) (c)

FIGURE 5.3: (a) a stroke, (b) a edge pixel p and its opposite edge pixel q, (c) each pixel along the ray (p, q) (extracted from [87]).

Surrounding Context Features

Usually the text components are aligned horizontally in the document images which results in structured surrounding area for a text component as compared to the non-structured surrounding area for non-text components. Therefore, the surrounding context of a connected component can play an important role in classifying the text and the non-text components. In [1], Bukhari et al. used the connected components within a rectangle $5 \times l$ by $2 \times h$ where l and h are the length and height of the target component. Then, each connected component and its surrounding context area is rescaled to a 40×40 window to create its surrounding feature. However, rescaling a large connected component into a small window size reduces its shape information. Thus, it will become a black window if the connected component is large and solid enough. Hence, Bukhari's method works only for separating text from halftone, one specific type of non-text. To solve the shortcomings raised by this, we extract context features without rescaling of the defined context. Those features include shape information, size information, and stroke width information. Our method is able to segment text versus all types on non-text.

Normalization using Log-normal distribution

Normalizing features is a necessary step. The size information and the position information of a connected component are normalized by considering all connected components of a document. Normalization overcomes the problem of documents of different resolutions. Log-normal distribution is used to normalize the features. Log-normal distribution is a probability distribution of a random variable whose logarithm is normally distributed. It is defined as:

$$F(x; \sigma; \mu) = \frac{1}{x\sigma\sqrt{2\pi}} \exp - \frac{(\ln x - \mu)^2}{2\sigma^2} \tag{5.8}$$

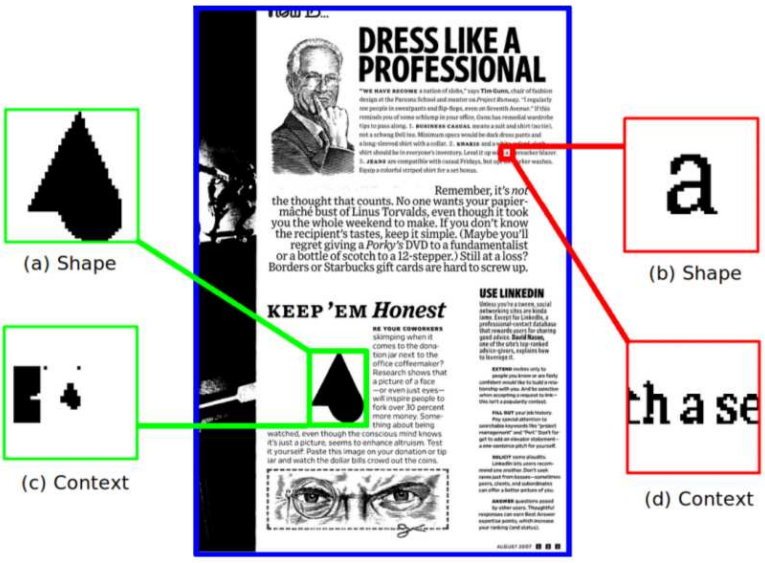


FIGURE 5.4: Text and non-text connected components shape and context features (extracted from [1]).

where x is variable; σ and μ are the mean and standard deviation of the variable's natural logarithm for all connected components of the document respectively.

5.4 Our Proposed Method

We propose a learning-based approach for text and non-text separation in document images. The training features are extracted at the level of connected components, a mid-level between the slow noise-sensitive pixel level, and the segmentation-dependent zone level. Given all types, shapes and sizes of connected components, we extract a powerful set of features based on size, shape, stroke width and position of each connected component. Adaboosting with Decision trees is used for labeling connected components. Finally, the classification of connected components into text and non-text is corrected based on classification probabilities and size as well as stroke width analysis of the nearest neighbors of a connected component.

Figure 5.5 shows a block-diagram of our proposed method for segmenting text and non-text in document images. A learning-based method is applied. Therefore, our method has two phases: training phase and testing phase. We describe each of the blocks in detail in the following subsections.

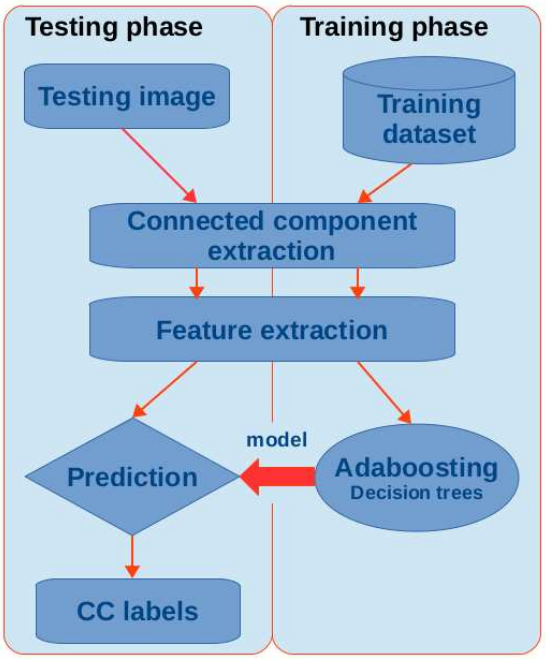


FIGURE 5.5: Block diagram of our proposed method.

5.4.1 Preprocessing

Our method is a connected component-based method. Therefore, pre-processing is an important step to binarize images, extract connected components and remove noise. First of all, Otsu's binarization method [89] is applied to binarize documents and then all connected components are extracted by the connected component labeling method. However, not all connected components are suitable for the learning phase and/or the testing phase such as noise, stains resulting from the scanning process. Based on the characteristics of connected components such as size, shape and position, we apply some rules to remove such small noisy components and stain connected components which usually have long shapes and appear in the boundary of a document. In practice, connected components with size lower 10; or with elongation lower 0.1 and with position near page border are removed.

5.4.2 Feature Extraction

Many features can be extracted from connected components. However, if a selected feature is not good, it does not benefit the classification. As shape and context are very important features with which humans recognize or segment an image, we extract features from size information, shape information, stroke width, and position of connected components. Bonakdar [90] has computed a set of such features for connected

components. We use the stroke width features and the subset of features presented in [90]. Our selected set of features is:

- *Elongation* and *Solidity*.
- $Height$, $Width$ and $X-Ycoordinates$ of a connected component The size of most text connected components in a document does not vary much. In addition, their position is also useful for classification, because unlike text, most of noise components, borders and frames are located near the boundaries of a document. X-Y coordinate is the center of a connected component. $Height$ and $Width$ denote the size of the bounding box of a connected component.
- $Hu\ moments$ [83] are a set of good features used to describe the shape of connected components because they are invariant to the image scale, rotation, and reflection. In our experiments, the first four of the seven moments are selected.
- The *stroke width* of connected components is a good feature for discriminating text from non-text. This is because text characters, lines, etc. have nearly constant stroke width while non-text do not [87]. The stroke width of an edge point of a connected component is computed based on the algorithm presented in [87]. The mean (mSW) and coefficient of variation ($Coefvariation SW$) of the stroke width at all edge points are also considered as features in our method.

For normalization, Log-normal distribution is used to normalize the elongation, solidity and height of the connected components, while the height and the width are normalized with respect to the size of the document from which they are extracted.

We simultaneously consider the context of connected components. According to [1], the surrounding context of a connected component provides important information for separating text and non-text. The ratio of height, width and stroke width between a connected component and its k-nearest neighbors are extracted as features (k=10 in our experiments). Table 5.1 shows the selected features. The last three features are computed from the k-nearest neighbor connected components as the surrounding context. The $mean\_of\_mSW$, $mean\_of\_height$ and $mean\_of\_width$ are the mean of mSW, $height$ and $width$ features of k-nearest neighbors.

5.4.3 Learning by Adaboosting Decision Trees

Boosting, presented in [91], is a supervised and a powerful learning tool. The main idea of this learning technique is to combine the performance of many "weak" classifiers

TABLE 5.1: The selected feature set for learning in our proposed method.

Features	Formulas
Log-normal distribution of 1/elongation	Eq 5.8 with $x = 1/elongation$
Log-normal distribution of 1/solidity	Eq 5.8 with $x = 1/solidity$
Log-normal distribution of height	Eq 5.8 with $x = height$
Normalized x's center	$x/doc.width$
Normalized y's center	$y/doc.height$
Logarithm Normalized of height	$\log(doc.height/height)$
Logarithm Normalized of width	$\log(doc.width/width)$
Logarithm of 1/elongation	$\log(1/elongation)$
Logarithm of 1/solidity	$\log(1/solidity)$
Logarithm of Hu's moment 1	$\log(Hu1 + 1)$
Logarithm of Hu's moment 2	$\log(Hu2 + 1)$
Logarithm of Hu's moment 3	$\log(Hu3 + 1)$
Logarithm of Hu's moment 4	$\log(Hu4 + 1)$
Coefficient of variation of SW	$CoefvariationSW$
The ratio of mSW	$mSW/mean\_of\_mSW$
The ratio of height	$height/mean\_of\_height$
The ratio of width	$width/mean\_of\_width$

(such as naive Bayes or Decision trees) to improve their performance. Different variants of boosting are known as Discrete Adaboost, Real AdaBoost, LogitBoost, and Gentle AdaBoost all of which have very similar overall structure [92]. To classify connected components, we use Discrete Adaboost with Decision trees because Decision trees is a simple learning method for our set of features, and it provides fast and good results. A two-class Discrete Adaboost model is trained as follows:

1. Give N samples (x_i, y_i) with $x_i \in \Re^K$, $y_i \in \{-1, +1\}$
2. Initialize a weight w_i of each sample.
3. For each weak classifier $T_m, m = 1, .., t$
   - Fit the classifier $T_m(x) \in \{-1, +1\}$, using weights w_i on the training data.
   - Compute error and scaling factor.
   - Update all weights w_i based on error and scaling factor so that the weights are increased for training samples that have been misclassified and vice versa.
4. The final classifier is the sign of the weighted sum over the individual weak classifiers.

Figure 5.6 shows the work-flow of Adaboosting with Decision trees algorithm.

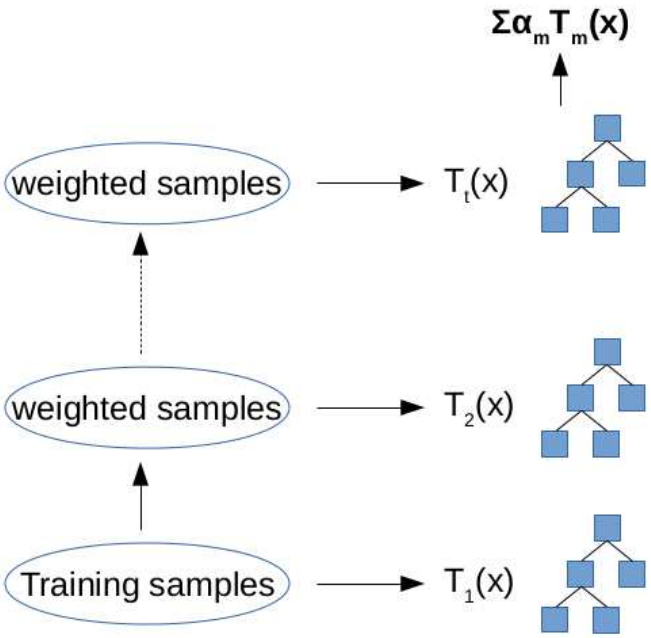


FIGURE 5.6: Adaboosting with Decision trees

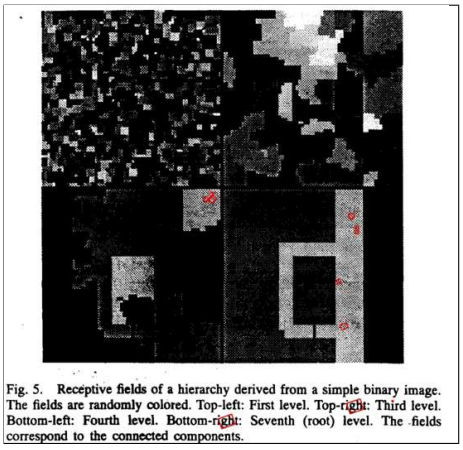
5.4.4 Post-processing

The post-processing step is necessary to correct some connected components which are labeled incorrectly by the classifier. For example, some text connected components like lines such as "l", "I", "-", "—", "_" and some connected components coming from more than one character which are assigned to incorrect class labels. Also, the broken parts of non-text connected components also have to be fixed.

To refine the class label of each connected component, we use the average text and non-text probabilities of nearest neighbors and the analysis of size and stroke width to update the classified labels of connected components. That is, if a connected component classified as non-text appears within high text probabilities of nearest neighbors and if it has similar size as well as stroke width, it will be reclassified as text connected component and vice versa. Figure 5.7 shows an example classified results before and after using the post-processing step.

5.4.5 Performance Evaluation Protocol

In this section, we present two methods for evaluation, one at the connected component-level and another at the pixel-level. The evaluation at connected component-level assumes that the importance of all connected components is the same, while the evaluation



(a) (b)

FIGURE 5.7: A document region (a) before and (b) after the post-processing step. Connected components shown in red colored boxes are incorrectly labeled by our method.

at pixel-level uses weights for small and large connected components. The weight represented here as the area of connected components. *Precision*, and *Recall* are used to evaluate the performance of our segmentation method:

$$Precision = \frac{tp}{tp + fp} \tag{5.9}$$

$$Recall = \frac{tp}{tp + fn} \tag{5.10}$$

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn} \tag{5.11}$$

Evaluation based on the connected component-level: each connected component has either a text or a non-text label based on ground-truth regions and a label predicted by our method. In that case, tp, tn, fp and fn are the numbers of connected components which are true positives, true negatives, false positives, and false negatives, respectively.

Evaluation based the on pixel-level: each pixel contains either a text or non-text label based on ground-truth regions or predicted connected components. In that case, tp, tn, fp, and fn are the number of pixels which are true positives, true negatives, false positives, and false negatives, respectively. Moreover, as we would like to compare our method to that of [1], we also use the same notation that they used. In [1], the metrics for performance evaluation are defined as *Text classified as text*, *Non-text classified as non-text* and *Segmentation accuracy*. In fact, those are derived from *Recall*.

- *Text classified as text*: the ratio of intersection of text pixels in both segmented and ground truth image to the total number of text pixels in ground truth image. It is the *Recall* of text class.

- *Non-text classified as non-text*: the ratio of intersection of non-text pixels in both segmented and ground truth image to the total number of non-text pixels in ground truth image. It is the *Recall* of non-text class.
- *Segmentation accuracy*: average ratio of text classified as text accuracy and non-text classified as non-text accuracy.

5.4.6 Experimental Results and Discussion

We experiment with two datasets. The first dataset is a subset of the standard UW-III dataset. We select 250 images from 1600 images which contain non-text regions, page-header regions, text-body regions, page-footer regions, etc. The second dataset is ICDAR-2009 page segmentation competition dataset for layout analysis of contemporary colored documents. The description of these databases is presented in Appendix A.

In the first experiment, to compare with the two popular classifiers Support Vector Machine (SVM) [93] and Multi-Layer Perceptron MLP [94], we use a set of connected components selected randomly from ICDAR-2009 dataset with 10000 connected components (5000 text and 5000 non-text). We use 5-fold cross-validation method. Table 5.2 shows that Adaboosting Decision Trees performs better and faster in our proposed method.

TABLE 5.2: Comparison of performance between Adaboosting Decision Trees and MLP, SVM on ICDAR-2009 dataset (10000 connected components).

Methods	Mean squared error	Build model time (s)
LibSVM	0.3963	15.04
MLP	0.3283	23.44
Adaboosting Decision Trees	0.2184	11.04

In the second experiment, we test our method on the subset of UW-III with 250 documents which contain halftones, drawings, and tables. We use 5-fold cross-validation method meaning that 200 documents are selected for training randomly and the remaining 50 documents for testing. For training, we keep all non-text connected components and select at random twice as many text connected components as non-text connected components (because the number of text connected components is much higher than non-text components, which causes imbalanced training). For testing, we keep all the text and non-text connected components.

Table 5.3 shows the average results of multiple runs of our method, every time randomly selecting the set of documents for training and testing. The Recall of non-text is about 82.83%, meaning that nearly 18% non-text connected components are classified as text.

However, most of them are actually text connected components but are located in the non-text zones of ground-truth such as text in the drawing, tables, charts, etc. Figure 5.8 shows three examples of texts contained in images, in tables and in charts where the actual text is labeled as non-text in the ground-truth, but they are predicted as text in our method.

TABLE 5.3: Performance evaluation results of our method on subset of UW-III dataset (250 documents)

Evaluation method	Text		Non-text	
	Precision	Recall	Precision	Recall
CC-level	95.76%	99.25%	96.58%	82.83%
Pixel-level	94.69%	99.26%	99.72%	98.07%

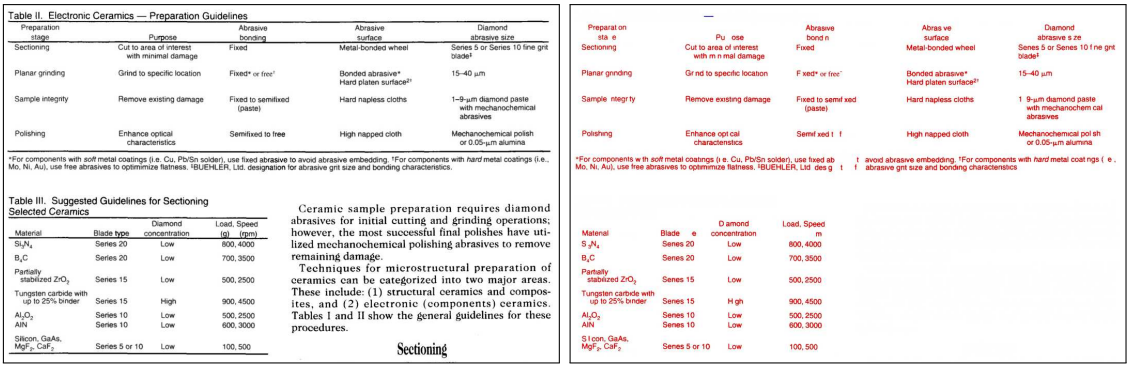
In the third experiment, to compare our method with the methods in [1, 3], 136 documents containing halftones are selected from UW-III dataset in order to try to ensure that it covers 95 documents selected by [1, 3]. The same scheme as in the second experiment is used. Table 5.4 shows that our method is significantly better and on a larger set of documents.

TABLE 5.4: Comparison of performance between our method and the methods in [1, 3] on UW-III dataset (Note that our subset is a superset of the documents in [1, 3]: 136 compared to 95 documents)

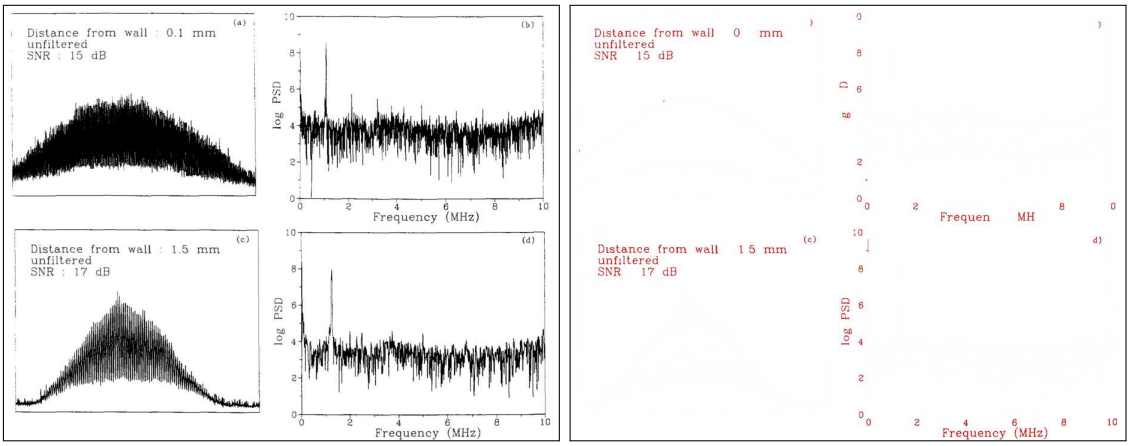
	Method in [1]	Method in [3]	Our method
non-text classified as non-text	98.91%	98.41%	99.49%
text classified as text	95.93%	99.42%	99.21%
segmentation accuracy	97.42%	98.92%	99.35%

Finally, we test our method on the challenging ICDAR2009 dataset. The same scheme for the training phase and testing phase as in the second experiment is used. The average results of multiple runs are shown in Table 5.5. There is a big difference of the non-text's recall between the two evaluation methods because ICDAR-2009 dataset comes from magazines. Therefore, there are many broken parts of natural images which are like text. In addition, the results also show that the surrounding context features improved the performance of separation, specially for non-text connected components. In this experiment, we also compare the performance of our system to the three well-known state-of-the-art systems (ABBYY FineReader® Engine 8.1, OCRopus 0.3.1 and Tesseract [95]) on F-measure. Table 5.6 shows that our method achieves the best result on text at 98.98% and a good result at 64.99% on non-text. The performance can be improved by adopting sophisticated preprocessing techniques for grouping the broken connected components in non-text regions before applying our method.

(a)



(b)



(c)

FIGURE 5.8: Three examples of (a) text contained in images, (b) in tables, and (c) in charts. In each example, left shows original images (a part of a document) and right shows the connected components that are non-text in ground-truth but our method classifies them as text.

TABLE 5.5: The results of performance evaluation of our method on ICDAR2009 dataset (55 documents).

Evaluation method	Text		Non-text	
	Precision	Recall	Precision	Recall
Our method without surrounding context features				
CC-level	97.82%	97.96%	63.05%	57.62%
Pixel-level	97.47%	88.67%	89.22%	97.38%
Our method with surrounding context features				
CC-level	98.89%	99.09%	66.72%	63.35%
Pixel-level	98.37%	92.46%	92.37%	98.41%

TABLE 5.6: Comparison of performance using F-measure between our method and Tesseract, FineReader, OCRopus on ICDAR-2009 dataset (55 documents).

	Text	Non-text
Tesseract [95]	92.50%	**74.23%**
FineReader	93.09%	71.75%
OCRopus	84.18%	51.08%
Our method	**98.98%**	64.99%

5.5 Discussion and Conclusions

This chapter has presented a method for segmenting the text and non-text in document images. The method is based on a set of powerful connected component features. These features utilize size, shape, stroke width, and position information of connected components.

Surrounding context features help to improve the performance in two stages: feature extraction and post-processing. Adaboosting with decision trees trained on those features to obtain a model for labeling connected components. Our results show that the method is simple and is efficiently able to discriminate text from non-text, including the text that appears within graphical zones.

Concerning the performance of the method, in our context, we do not take into account the very precise discrimination because our goal is just to get rid of as much text as possible and keep all the non-text. For this chapter, we tried to optimize the performance for both text and non-text. However, in the next chapter we will tune the method to have high recall for graphics (even if this means lower precision) and add it as a preprocessing step into our framework of logo spotting. In addition, in the next chapter an approximate nearest neighbor algorithm is also studied to speed up our system.

Chapter 6

Performance Optimization of Logo Spotting

Searching for the most similar matches to high dimensional vectors is the most computationally expensive part of many computer vision algorithms. Nearest neighbor searching is not an exception. To deal with this problem, many techniques are proposed such as dimensionality reduction techniques or approximate nearest neighbor algorithms. In addition, preprocessing techniques are also applied to reduce the size of the matching space.

In this chapter, we focus on optimizing our logo spotting framework which has been proposed in Chapter 3. First, in Section 6.1, the dimensionality reduction techniques and approximate nearest neighbor algorithms are used to deal with logo spotting on documents which are captured from a camera. In Section 6.2, we apply the text/graphics separation method as a preprocessing technique to reduce the number of key-points in the matching space. Then, an approximate nearest neighbor algorithm is used to optimize the document retrieval system.

6.1 Logo Spotting in Camera-Acquired Documents

In this section, the logo spotting on document images will also be deployed in a camera-based application. We use the framework for logo spotting on document images proposed in Chapter 4. According to our previous experiments on the result and the computation time, the framework using the 2-nearest neighbor matching rule is chosen. However, to meet the real time requirement of a camera-based application, we need some nearest

neighbor matching algorithms considered to be sufficiently efficient. Moreover, reducing the high dimension of descriptor vectors is also considered.

6.1.1 Reducing the Dimension of Description Vector

Many methods for reducing the dimensionality of feature vectors have been investigated by many researchers. These methods deal with different problems, but they have a similar goal. Principal Components Analysis (PCA) [52] is a classical method that provides a sequence of best linear approximations to a given high-dimensional observation. Multidimensional Scaling (MDS) [96] is closely related to PCA. Factor analysis [97, 98] and Independent Component Analysis (ICA) [99] are also linear approximations and use factor analysis modeling the correlation structure.

Principal Components Analysis (PCA) is a very popular technique for dimensionality reduction. Of the two approaches performing dimensionality reduction: feature extraction and feature selection, PCA is a feature extraction approach. Applying PCA on a set of features will create a set of new features. In other words, given a set of data of n dimensions, PCA aims to find a linear subspace of dimension d lower than n so that the data points lie mainly on this linear subspace. We can study an example rather than reviewing the PCA's mathematical theories. The example in 2D is shown in Figure 6.1. At first sight, the points are set in a random order concerning Feature 1 axis or Feature 2 axis. However, we can observe a linear pattern (indicated by the black line). This means that it is possible to approximate the set of points to a single line. Therefore, it can reduce the dimensionality of the given points from 2D to 1D. Moreover, we could also see that the points vary the most along the black line, more than they vary along the Feature 1 axis or Feature 2 axis. This means that if we know the position of a point along the black line we have more information about the point than if we only knew where it was on Feature 1 axis or Feature 2 axis. Finding the direction along which the data varies the most can be calculated using PCA. In fact, PCA provides vectors called *eigenvectors* which are the *principal components* of the data set. The size of each eigenvector is encoded in the corresponding eigenvalue and indicates how much the data vary along the principal component. In this example, the result of running PCA consists of 2 eigenvectors as Figure 6.1(b).

6.1.2 Nearest Neighbor Searching using Approximate Approaches

The nearest neighbor problem is defined as follows: given a set of n points $P = p_1, .., p_n$ in a vector space X and a query point $q \in X$, build a data structure which returns the points in P which are the closest to the query q. This problem is of major importance in several

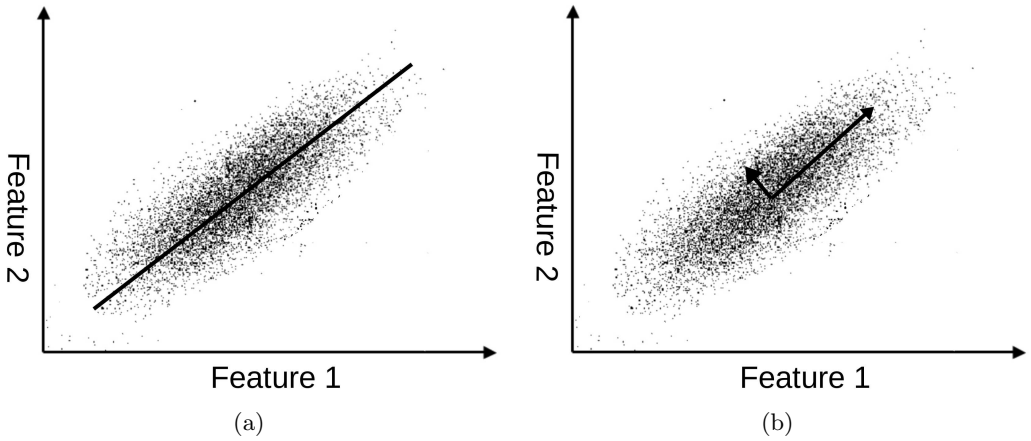


FIGURE 6.1: An example in 2D: (a) it is possible to approximate the set of points to a single line; (b) the result of running PCA consist of 2 eigenvectors.

areas; such as data compression, databases and data mining, information retrieval, image and video databases, machine learning, pattern recognition, statistics and data analysis. A simple linear searching method is the most efficient for nearest neighbor searching; however, it is often too costly to deal with high dimensional spaces. This brings interest to many researchers to provide algorithms that perform approximate nearest neighbor searching [100–106], in which the results are sometimes the best neighbors instead of the closest neighbors. In other words, such approximate algorithms can be orders of magnitude faster than exact search, while still providing near-optimal accuracy. In the literature, the multiple randomized kd-tree and the hierarchical k-means tree are two algorithms which give the best performance [107, 108].

The randomized kd-tree algorithm

The classical kd-tree algorithm, presented by Freidman et al. [100], is the most widely used algorithm for nearest neighbor searching. The general idea of kd-trees is that the data is splited into two parts based on a hyperplane orthogonal to a chosen dimension at median in the dimension with the greatest variance in the data. This algorithm is reported with N build time and $log(N)$ search time for N points, and efficient in low dimensions. However, the performance degrades quickly in high dimensions. An improved version of kd-tree algorithm using multiple randomized kd-trees is presented by Silpa-Anan and Hartley [103]. The randomized trees are built by choosing the split dimension randomly from the first D dimensions on which data have the greatest variance. Meanwhile, the original version of kd-tree algorithm splits the data in half at each level of the tree on the dimension where the data show greatest variance.

The hierarchical k-means tree algorithm

The algorithm is proposed by Muja and Lowe [107, 108] based on clustering the data points. Different from the kd-tree algorithm or the randomized kd-tree algorithm to

partition the data based on one dimension at a time, the hierarchical k-means tree is built by splitting the data points at each level into K distinct regions using k-means clustering. Muja and Lowe also provide an algorithm to select and determine the best algorithm and parameter settings automatically for any given dataset. A cost is calculated based on a combination of the search time, tree build time, and tree memory overhead with weights. This algorithm is also implemented in the Fast Library for Approximated Nearest Neighbor (FLANN) [109] which is used for our experiments in the next section.

6.1.3 Experiments

Since SIFT and SURF descriptors give the best results in logo spotting, and ORB descriptor is known as a fast descriptor for camera-based applications [110], these descriptors will be compared with each other in our experiments. We use the framework for logo spotting on document images proposed in Section 4.2 for logo detection. Therefore, a given set of logo images concerns as a gallery and a captured image concerns as a query document. However, to satisfy the real time requirement of a camera-based application, the dimensionality reduction method and the approximate nearest neighbor (ANN) algorithms are deployed into the key-point extraction and key-point matching steps, respectively.

For the high dimension feature SIFT, the length of feature vectors is reduced to 64. FLANN library [109] with randomized kd-tree algorithm, hierarchical k-means tree algorithm packaged as part of the OpenCV is then used to find matches. For ORB descriptor, we use LSH algorithm [111] in FLANN to deal with binary descriptor as ORB.

The first experiment scenario is to compare the linear searching to ANN searching algorithms, such as randomized kd-tree algorithm, hierarchical k-means tree algorithm. We create a video clip with 150 frames with complex background as shown in Figure 6.2. We use a gallery with 20 logo images which has total 3080 key-points. The number of extracted key-points of each frame is 1543 on average, and it takes 228 ms for key-point extraction on average.

Table 6.1 shows the comparison of the number of missing frames and the average detected time per frame. By comparison, the hierarchical k-means tree algorithm appears to be not suitable in this case because, in my view, the number of searched key-points is not big while the time to built the tree is longer. Other screen-shots are shown in Figure 6.3, and Figure 6.4 shows the cases where a part of the logos is covered.

FIGURE 6.2: A frame of the video clip with complex background.

TABLE 6.1: Comparison of performance of methods using SIFT, SURF, and ORB descriptors on the video clip with 150 frames with complex background.

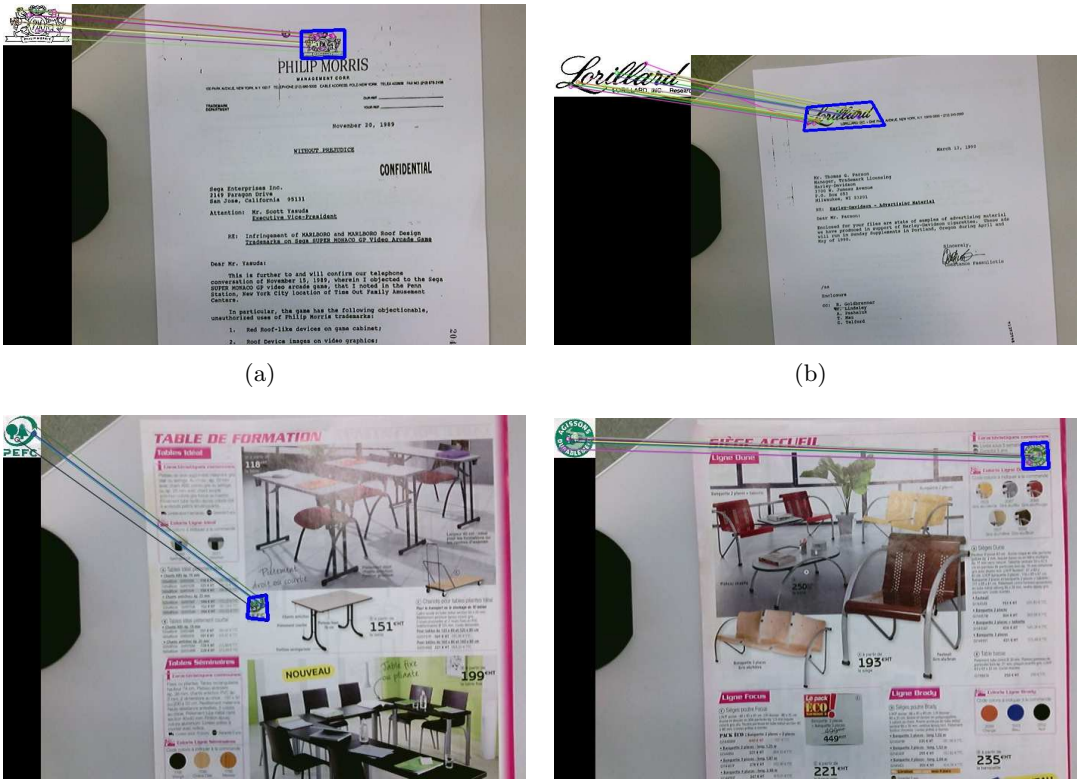
Dimensionality reduction	**Algorithm**	**# Missing frames**	**Time/frame (ms)**
No reduction (128-dimension)	linear	2	680
	kd-tree	2	499
	k-mean tree	2	2150
Reduction (64-dimension)	linear	2	516
	kd-tree	2	402
	k-mean tree	3	1338

In the second experiment, we compare the performance of methods using SIFT, SURF, and ORB descriptors. To diversify the scenario of experiments, we create 4 video clips with 200 frames per clip with different scales and levels of brightness. For Video 1 and Video 2, we set the camera height value at 10 cm. The distance in Video 3 is set 20 cm. For Video 4, we use the same query document of Video 3 but the distance is 30 cm. As we can see in Table 6.2, the method using SURF descriptor can work well in the cases where the distance from the camera to the document is not too big. Meanwhile, the method using SIFT descriptor gives the best result in all cases. Some screen-shots are illustrated in Figure 6.5.

6.2 Logo Spotting in Text/Graphics separated Documents

6.2.1 Introduction

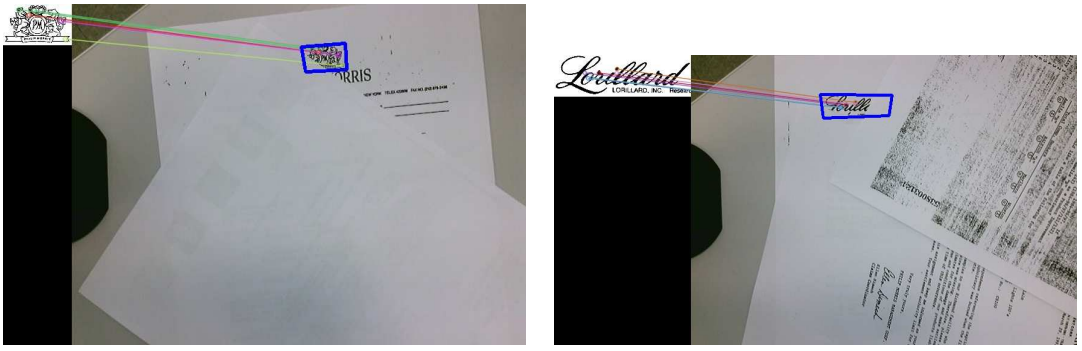
In this section, Tobacco-800 database will be used to estimate the performance of logo retrieval on text/graphics separated documents. Each document will be divided into a



(a) (b)

(c) (d)

FIGURE 6.3: Screen-shots of experiments on Tobacco-800 database and our magazine database.

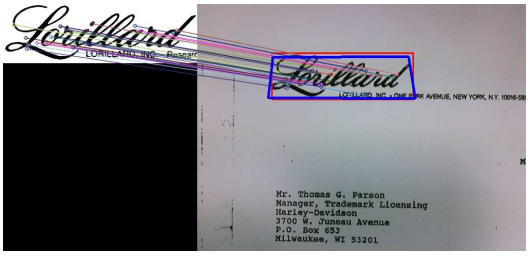


(a) (b)

FIGURE 6.4: Some cases where a part of logo are covered.

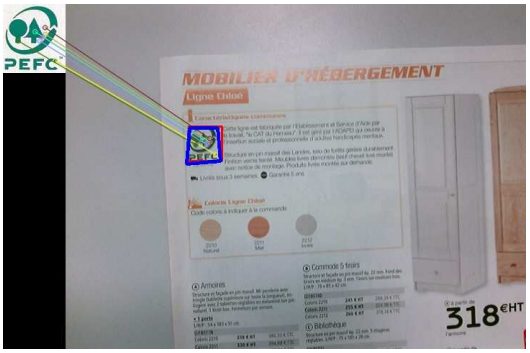
TABLE 6.2: Comparison of performance of methods using SIFT, SURF, and ORB descriptors on four video clips.

Video clips	Descriptors	# Missing frames	Time per frame (ms)
Video 1	SIFT	0	294
	SURF	0	270
	ORB	101	285
Video 2	SIFT	0	399
	SURF	0	441
	ORB	154	345
Video 3	SIFT	0	353
	SURF	13	484
	ORB	200	-
Video 4	SIFT	4	321
	SURF	193	397
	ORB	200	-



(a) A screen-shot of video 1

(b) A screen-shot of video 2



(c) A screen-shot of video 3

(d) A screen-shot of video 4

FIGURE 6.5: Some screen-shots extracted from Video clips with different scales and levels of brightness of four video clips.

text layer and a non-text layer using a text/graphics separation algorithm. The set of non-text layers is then used as a dataset of logo retrieval system proposed in Chapter 4. Unfortunately, the ground-truth of Tobacco-800 database is not available for text/-graphics separation. Therefore, to compare the performance of algorithms, we propose an evaluation protocol based on keeping the logo area regions and reducing the non-logo area regions in Section 6.2.2. We also compare our text/graphics separation method presented in Chapter 5 with a block based segmentation algorithm (Run-length Smoothing Algorithm [73, 112]) and a pixel based segmentation algorithm (Bloomberg's algorithm [81]).

Run-length Smoothing Algorithm

Run-length smoothing algorithm (RLSA) has been used in [73, 112] to segment graphics and text on a document based on detecting long vertical and horizontal pixel lines. The method developed for analyzing/segmenting documents has two steps. First, the document is divided into blocks (regions). Each block should contain only one type of data, such as text, graphics, halftone, etc. Second, each block is classified using some of its basic features.

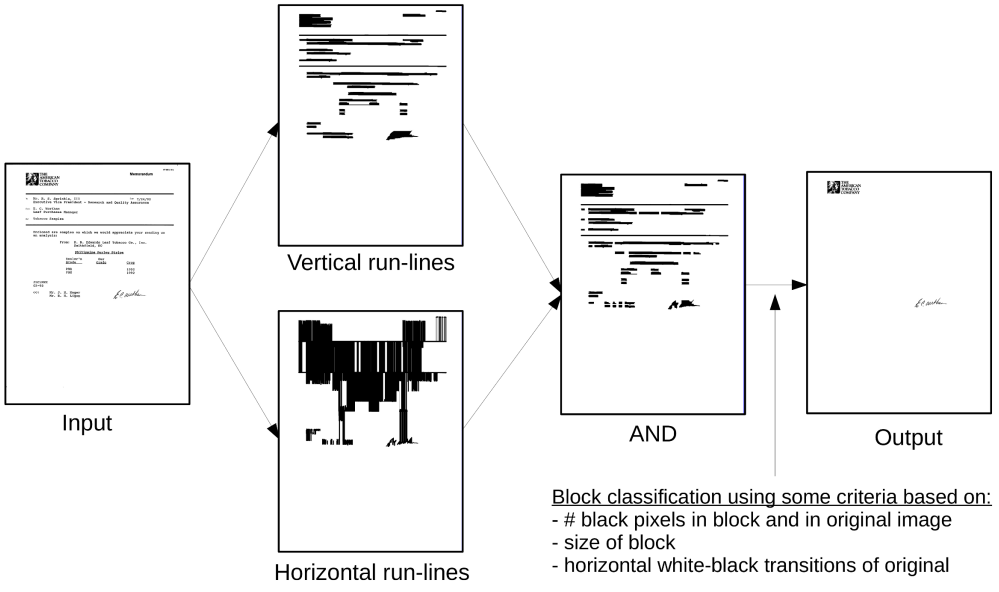


FIGURE 6.6: The outline of the RLSA which the output is graphic layer.

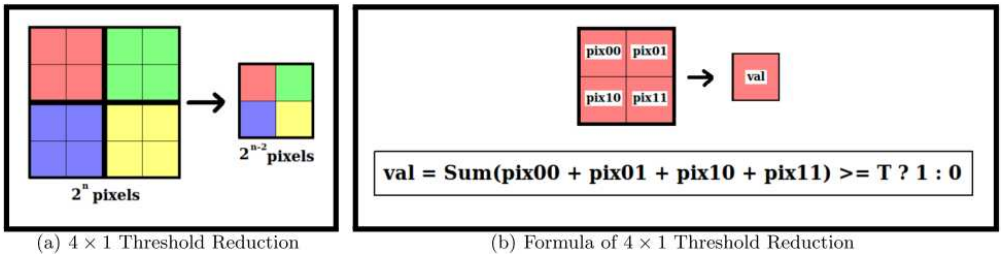
The main idea of RLSA to segment the blocks is connecting near-by pixels together thanks to a given threshold C. The RLSA is applied on two directions row-by-row as well as column-by-column with two different values of C for row and column to create two bit-maps. The two bit-maps are then combined by using the logical AND operation. Additional horizontal smoothing using the RLSA produces the final segmentation result.

In the classification step, each block is labeled by using some criteria based on its height, eccentricity, ratio of the number of block pixels to the area of the bounding box, and the mean horizontal length of the foreground pixel runs of the original data. Each block is classified as texts, horizontal or vertical solid lines, graphics and halftone images. Figure 6.6 shows the outline of the RLSA from which the output is the graphic layer. However, choosing the value for the series parameters of RLSA is difficult and depends on the type of documents in each database.

Bloomberg's algorithm

The main technique used in this algorithm is multi-resolution. The authors defined the key concept of "threshold reduction" for re-sampling the image to lower resolution image. The threshold reduction is defined as follows:

Threshold reduction: Given a threshold $T \in [1, 2, 3, 4]$ and a binary image where each foreground pixel and background pixel are represented by "1" and "0", respectively. Considering a 2×2 pixel window, if the sum of the values of four pixels is greater or equal to T, the 2×2 window is replaced by a foreground pixel in the sub-sampled image. Otherwise, it is replaced by a background pixel.



(a) 4×1 Threshold Reduction (b) Formula of 4×1 Threshold Reduction

FIGURE 6.7: Definition of multi-resolution morphology based threshold reduction operation: (a) each 2×2 window is replaced by one pixel (4×1 Reduction). (b) the value of pixel is "1" if the sum of the values of four pixels is greater than or equal to the threshold T, otherwise "0" (extracted from [3]).

The algorithm is as following:

1. The input image is processed twice using the threshold reduction operation with T=1.
2. Two threshold reductions with T=4 and T=3 are respectively applied and then followed by a morphological opening by using a 5×5 structuring element.
3. The sub-sampled image is expanded twice using 1×4 expansion operation. The result image at this step is referred to as the seed image.

4. The halftone mask image is created by comparing and selecting only fully or partially overlapping components between the seed image and the image of step (1). Then, a morphological dilation with 3×3 structuring element is deployed.

5. The halftone mask image is then expanded twice using 1×4 expansion operation to become equal to the dimension of the input image.

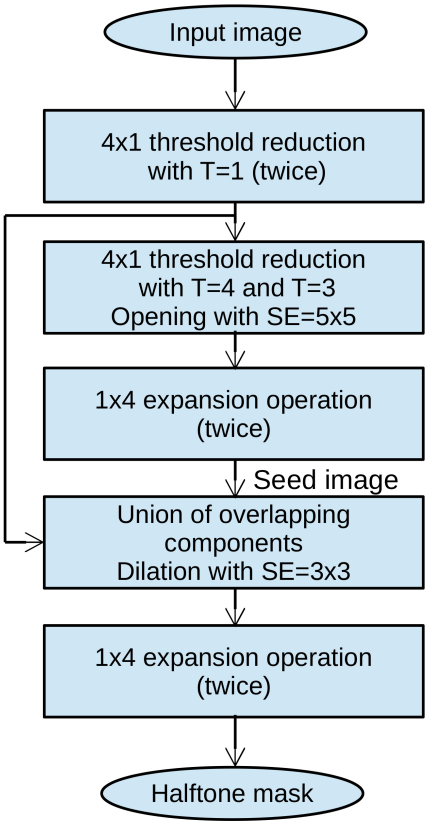


FIGURE 6.8: Data flow diagrams of Bloomberg's text/image segmentation algorithm.

6.2.2 Evaluation Protocol

As discussed above, Tobacco-800 database was not created for text/graphics separation. Its ground-truth provides only the logo regions and the signature regions [34, 113]. In this context, we consider only two classes of regions: logo and non-logo regions. Therefore, a text/graphics separation algorithm will be chosen so that it can reject the non-logo regions but it keeps the logo regions. We define the first factor based on the remaining area of logo region after the segmentation as follows:

$$\epsilon = \frac{Area(logo\_region\_after\_segmentation)}{Area(original\_logo\_region)} \tag{6.1}$$

where the function $Area()$ returns the area of a region, $logo\_region\_after\_segmentation$ is the region of logo on the segmented document, and $original\_logo\_region$ is the region of logo on the original document. The logo region is estimated based on the ground-truth. The important thing here is that the value of ϵ is chosen so that it has realistic sense. In our case, we consider a correct case if $\epsilon \geq 0.75$. Otherwise, it is a missing case. Figure 6.9 shows examples of a correct case and a missing case.

In addition, we also define the second factor to compute the reduced area of all documents after the segmentation.

$$reduction = 1 - \frac{\sum Area(document)}{\sum Area(original\_document)} \tag{6.2}$$

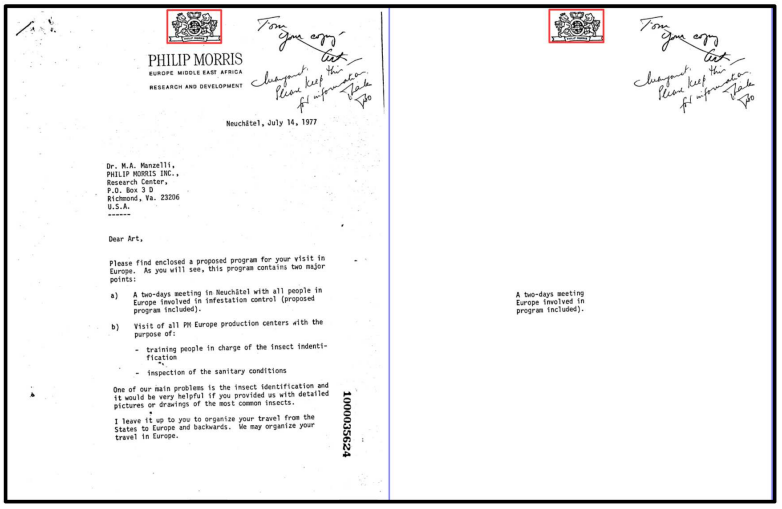
where $\sum Area(document)$, $\sum Area(original\_document)$ are the sum of region area (except logo regions) of all segmented documents and original documents, respectively. In our context, a separation algorithm is better if it has more correct cases and a lower $reduction$ factor.

6.2.3 Experiments

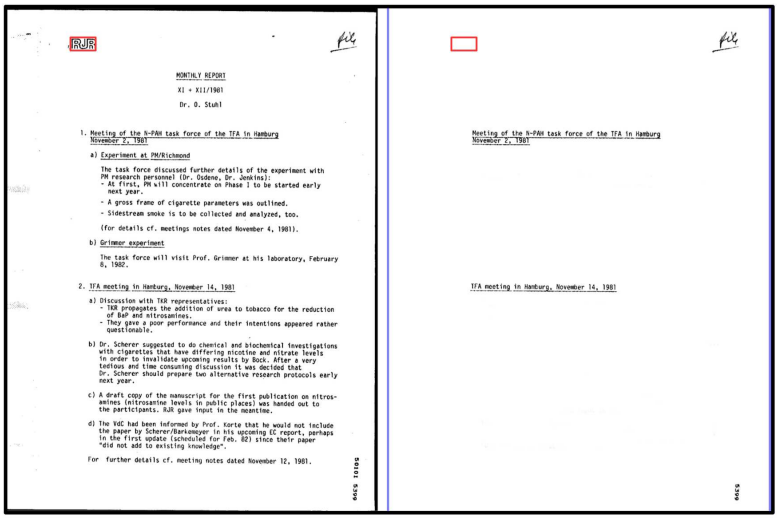
Comparison between text/graphics separation algorithms

In the first experiment, we compare RLSA (Method 1), Bloomberg's algorithm (Method 2) to our proposed method in Chapter 5 (Method 3). Because our method is a learning-based method, it needs a set of documents for training. To avoid a bias, we use the set of 136 documents of UW-III dataset as in experiments of Chapter 5. We keep all non-text connected components and select randomly the number of text connected components to be twice as many as that of non-text connected components (because the number of text connected components is much higher than non-text components, which causes imbalanced training). For testing, we use all 1290 documents of Tobacco-800 database.

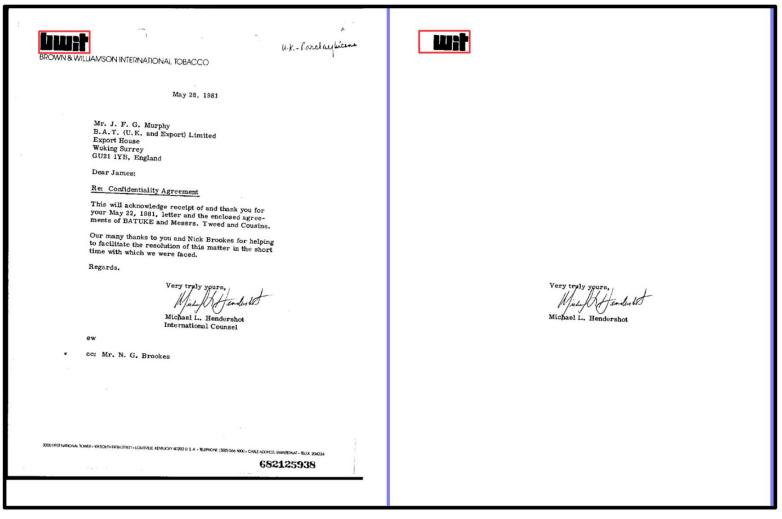
Table 6.3 shows the results. Method 2 has lower performance in terms of missing cases because Bloomberg's algorithm is proposed for text and halftones separation. Our method (Method 3) and RLSA (Method 1) give the best results with only 10 missing cases over 412 documents containing logos and Method 1 can reduce more region areas than Method 3. However, we strongly prefer to use Method 3 because Method 1 needs a series of fixed parameters which depend on each dataset while Method 3 can work automatically. In addition, using a different database for training and testing



(a) a correct case



(b) a missing case



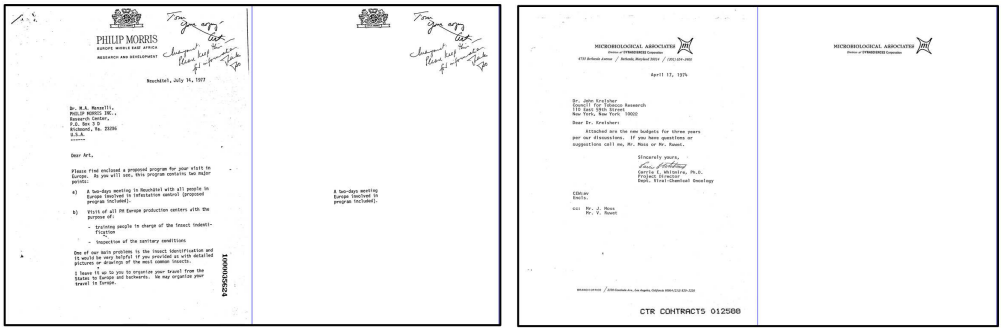
(c) another missing case

FIGURE 6.9: Examples of (a) a correct case, (b) a missing case, and (c) another missing case because the remain area of logo is lower than 75% compare to the original logo.

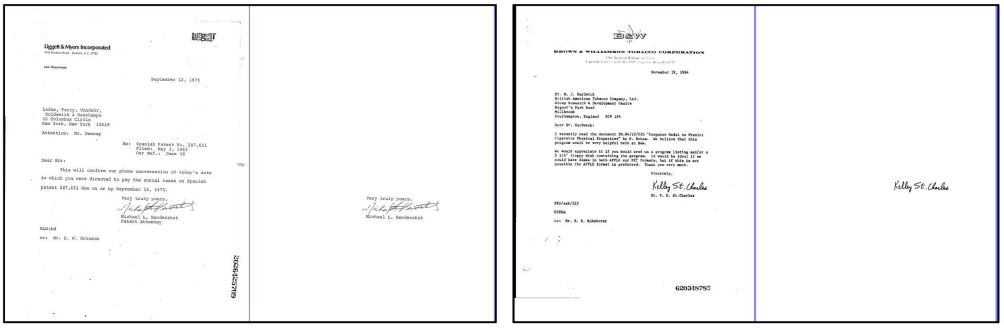
avoids overfitting [1] problem and it does not depend on testing datasets. Some correct cases and missing cases of the Method 3 are shown in Figure 6.11.

TABLE 6.3: Comparison of performance of RLSA (Method 1), Bloomberg's algorithm (Method 2) and our proposed method (Method 3)

	Method 1	**Method 2**	**Method 3**
Number of missing cases	10	409	10
Reduction	52.79%	83.11%	43.20%
Avg. computation time	0.97s	0.05s	0.68s



(a) correct cases



(b) missing cases

FIGURE 6.10: Examples of (a) correct cases and (b) missing cases because the remain area of logos are lower than 75% compared to the original logos.

Comparison based on a document retrieval system

In the second experiment, the document retrieval system presented in Chapter 4 is applied to compare between systems using original document database, segmented document database, segmented document database with ANN using kd-tree algorithm. We evaluate the systems based on the number of extracted key-points, average matching time, and three retrieval measures (MAP, MRP and MAPc presented in Chapter 4).

[1] Overfitting occurs when a model begins to "memorize" training data rather than "learning" to generalize from trend.

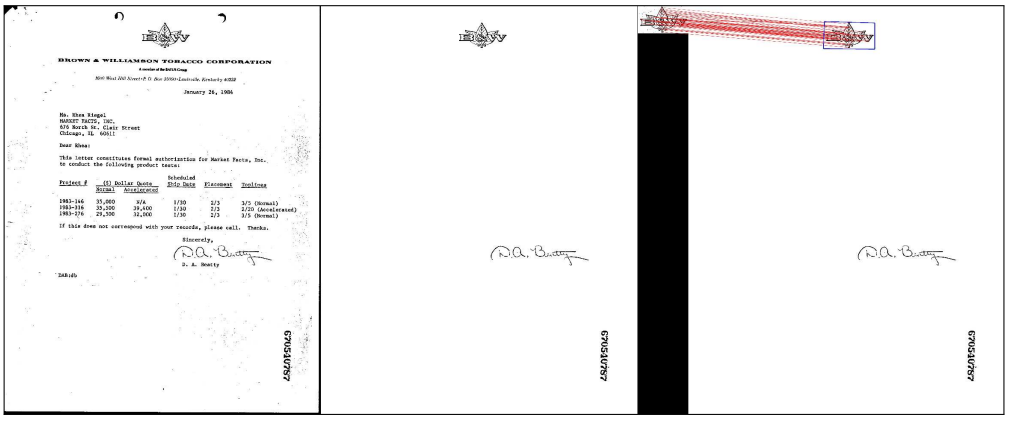
Table 6.4 presents the results. There is a small decline in precision by 3.36% – 1.54% compared to segmented document database and segmented document database with ANN using kd-tree algorithm, respectively. However, the matching time decreased sharply over 34% for the system using the segmented document database and over 56% for the system using segmented document database with ANN using kd-tree algorithm.

TABLE 6.4: Comparison of performance on Tobacco-800 database between systems using original document, segmented document database, segmented document database with ANN using kd-tree algorithm (the results, if we ignore the missing cases of the segmentation step, is shown in brackets).

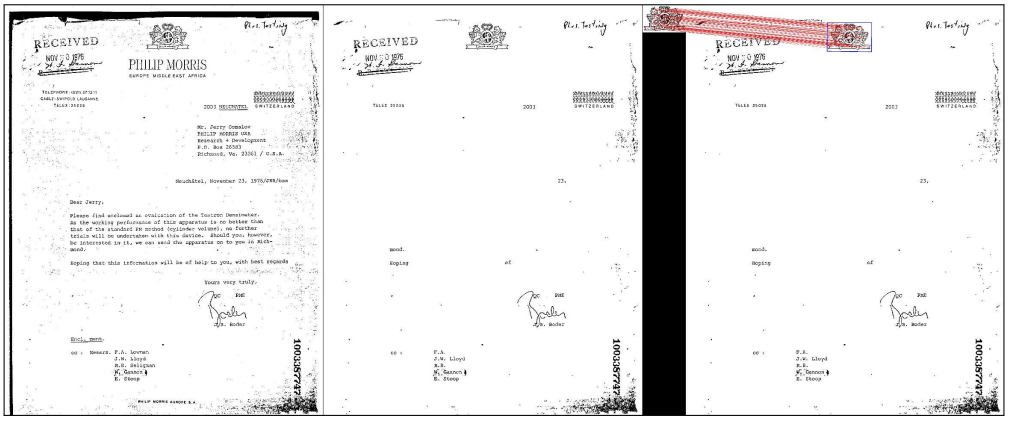
	Original documents	**Segmented documents**	**Segmented documents & ANN**
Number of key-points	17.9M	9.8M	9.8M
Avg. matching time	1.29s	0.84s	0.57s
MAP	88.31%	85.47% (86.89%)	85.33% (86.83%)
MRP	85.87%	83.37% (84.61%)	83.35% (84.60%)
MAPc	91.97%	88.61% (90.43%)	88.59% (90.41%)

6.3 Conclusions

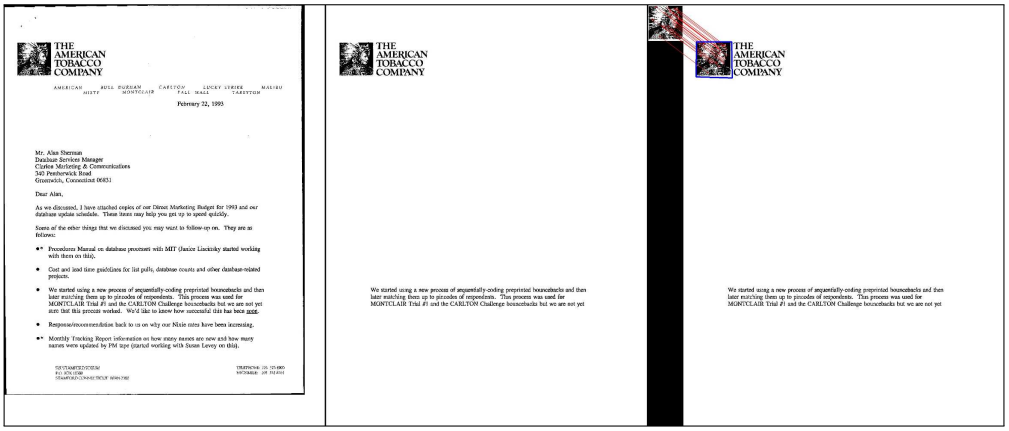
Since the key-point-based matching methods assume costly methods, in this chapter, we applied dimensionality reduction techniques to reduce the high dimensional vectors of local descriptors and ANN algorithms to optimize our proposed framework. Then, it can deal with the requirement of camera-based applications. In addition, the second part of this chapter presents the experiments of a document retrieval system on the text/graphics separated documents and ANN algorithms. The results show that the computation time of the system decreased sharply by 56% while its MAP, MRP and MAPc are not affected much with a slight decrease of 3%. However, if we ignore the missing cases of the segmentation step its MAP, MRP and MAPc decreases – all less that 1.5% – could be considered insignificant in our applications.



(a)



(b)



(c)

FIGURE 6.11: Some examples of logo matching on segmented documents and ANN algorithm on Tobacco-800 database. Column 1 is original documents, Column 2 is the documents after text/graphics separation, and the spotting results are shown in Column 3.

Chapter 7

Conclusions and Future Perspectives

7.1 Conclusions and Discussion

In this thesis, we have proposed a logo spotting framework applied to spotting logo images on document images and focused on document categorization and document retrieval problems. The spotting method is formulated in terms of searching for matches between all interest points of query logo images and of document images. Interest points are extracted from images and are described under local feature vectors by local detectors and local descriptors. As explained in Chapter 1, our work has been motivated by the specific problem of proposing a spotting methodology which is able to locate and recognize logos contained within a database of complete document images. A lot of interest exists worldwide for mass digitization of document collections and their storage in digital libraries. Moreover, we need effective and efficient methods for searching and indexing those documents. One important part of indexing documents is logo spotting.

We have built the proposed framework for logo spotting with four main steps. The first step aims to represent the logos and documents as sets of key-points. In the second step, matches between key-points of logos and documents are computed using key-point matching methods. The third step consists of segmentation and outliers reduction using a density-based clustering method. In the fourth step, geometric verification and localization are applied to re-filter the matches and to locate the logos. Throughout this thesis, we have made contributions in each step. A brief summary of these contributions follows:

- **Logo description**: A logo is a complex graphic element which contains a variety of different information, such as color, shape, texture, and many other features. To describe logos, as discussed in Chapter 2, a robust description technique, such as the point-based description technique is able to deal with the complex graphic elements as logos. Regarding the type of the local feature extraction, we have selected one of the rotation and scale invariant detectors because logos may occur on documents with different sizes or different document sizes. In addition, to deal with potential problems of scanned documents, descriptors are required to be robust or invariant to characteristics, such as brightness, contrast, rotation, scale, affine transforms, noise, etc. The experiments in Chapter 3 and Chapter 4 show that blob detectors and spectra descriptors are the most suitable description techniques to represent logos.

- **Matching methodology**: Once the suitable description technique has been chosen, we have organized all the information arising from the document collection to provide an efficient search access. Key-point-based approaches are used to match each key-point of the logos independently to the document key-points. Although key-point-based approaches assume costly approaches, they give best results. For instance, in the nearest neighbor technique, a sequential algorithm gives better results than other techniques. The optimization of logo spotting based on key-point matching are also discussed in this thesis.

  + *Key-point matching with nearest neighbor simple method* considers the first nearest neighbor as the best match. However, it also provides many false matches. Truly, there is a possible ambiguity between the first nearest neighbor and the second one.

  + To solve this ambiguity issue, *key-point matching with the 2-nearest neighbor matching rule* uses the ratio of the first and second nearest neighbors. If the ratio is greater than a given threshold, no match is found. Otherwise, the considered key-points are accepted as a match. This method gives the best matches, however, as discussed in Chapter 3 and Chapter 4, it also rejects some correct matches.

  + *Key-point matching with post-filter by a second descriptor* uses another descriptor in the key-point matching step. The method has the ability to get more correct matches without increasing the false positives. This method is able to provide better results than the others. It should be preferred in applications where correct identification of the incoming documents is of more importance than the false alarms (documents which do not contain any logos). On the other hand, if we expect a faster method or intend to minimize

the false alarms which may include misclassified documents, the key-point matching with 2-nearest neighbor matching rule should be considered.

- **Key-point grouping**: In practice, not all matches are located inside the logo region on the document, some of them are located in other regions of the document. In addition, there is a high concentration of matches on the correct logo while the concentration decreases when considering the incorrect logo. A density-based clustering method, here DBSCAN [60], is used to deal with these problems. The use of a density-based clustering method to group the matches can help not only segment the candidate logo region but also reject the incorrect matches as outliers.

- **Geometric verification and localization**: After grouping matches, there are still some incorrect matches which occur within the key-point groups. This can be a cause of imprecise localization or incorrect spotting. To maximize the performance of our logo spotting methods, these key-points need to be removed before the localization step starts. An algorithm with two stages is proposed for geometric verification based on homography with RANSAC. The proposed algorithm also aims at solving cases where a cluster region may not cover the logo region, but just a part of the logo region. Finally, the position of logos is located based on the homography matrix.

Since key-point-based approaches assume costly approaches, we have invested to optimize our proposed framework in Part II of this thesis. We have considered problems of: text and non-text separation, dimensionality reduction and approximate nearest neighbor search and these are studied.

- **Text and non-text separation**: Our work has also proposed a method for segmenting text and non-text in document images. The method is based on a set of powerful connected component features. These features utilize size, shape, stroke width, and position information of connected components. In addition, using surrounding context features helps to improve the performance in two stages: feature extraction and post-processing. Adaboosting with decision trees trains on those features to obtain a model for labeling connected components. Our results show that the method is simple, fast and is efficiently able to discriminate text from non-text, including the text that appears within graphical zones.

- **Dimensionality reduction and approximate nearest neighbor search**: We applied dimensionality reduction techniques to reduce the high dimensional vector of local descriptors and ANN algorithms to optimize our proposed framework.

Therefore, our solutions can deal with the requirement of camera-based applications. In addition, we have also conducted experiments for a document retrieval system on the text and non-text segmented documents and ANN algorithm. The results show that the computation time of the system decreased sharply by 56% while its accuracy decreased slightly by nearly 2.5%.

Our proposed approach is able to spot logos in documents of both simple and complex layout. This has been shown through the good achieved performance on the Tobacco-800 database (simple layout) and the magazine database (complex layout). The used techniques for preprocessing and post-processing have made our approach robust to noise such as the scanning noise in Tobacco-800 database and the bleed-through noise in the magazine database. The proposed framework is flexible enough to be used in different applications, such as: categorization, retrieval, etc. Each step (module) of the proposed framework can be optimized or replaced by other more powerful techniques in the future. For example replacing the logo description part by a better descriptor.

Overall, we have proposed an effective and efficient approach for solving the problem of logo spotting in document images. We have validated the performance of our approach on both standard and private databases. We have designed our approach to be flexible for future improvements by us and by other researchers. We believe that our work could be considered as a step in the direction of solving the problem of complete analysis and understanding of document images.

7.2 Future Perspectives

First of all, one of the directions for our future research is that the proposed framework should be tested on more databases with different kinds of logos and documents. More databases of different types of logos and documents can be collected for more reliable performance evaluations.

In terms of logo description, we would also like to further investigate the use of a different method than the one proposed in the thesis. We would like to use "shape" description where a logo is described as a set of basic geometric primitives and the spatial relationships among them. We believe that the spatial relationship of primitives could give more information and these data structures could be able to better describe logos. We would like also to investigate combining key-point descriptors with shape-based descriptors. These methods could be effective in spotting textureless logos.

In terms of text and non-text separation, we would like to consider using advanced preprocessing to resolve problems in distinguishing non-text in contemporary documents.

Moreover, to avoid engineered features which are designed for specific databases, we would like to investigate the use of automatic feature learning for text versus non-text discrimination.

Appendix A

Databases

A.1 Tobacco-800 Database

The Tobacco-800 database [34, 113] is a well-known public dataset for document analysis. The Tobacco-800 dataset has 1290 document images including 412 document images containing logos and 878 document images containing no logo. There are 35 different logos in the gallery, and for each logo, the number of document images containing this logo ranges from 1 to 83. It is a realistic database for document image analysis research as these documents were collected and scanned using a wide variety of equipment over time. In addition, a significant percentage of Tobacco800 is consecutively numbered multi-page business documents, making it a valuable testbed for various content-based document image retrieval approaches. Resolutions of documents in Tobacco800 vary significantly from 150 to 300 DPI and the dimensions of images range from 1200 by 1600 to 2500 by 3200 pixels. Figure A.1 shows examples of Tobacco-800 database.

The ground-truth of the Tobacco-800 database was created by the Language and Media Processing Laboratory, University of Maryland. This release includes the ground-truth information on both signatures and logos in this large complex document image collection. Based on the ground-truth, a set of 432 isolated logos extracted from Tobacco-800 database are used in our experiments. Figure A.2 shows some isolated logos extracted from Tobacco-800 database.

A.2 Private Advertising Magazine Database

We have collected this database from advertisement magazines. It contains 100 images scanned from these papers at 400dpi. The number of logos which appear in the database

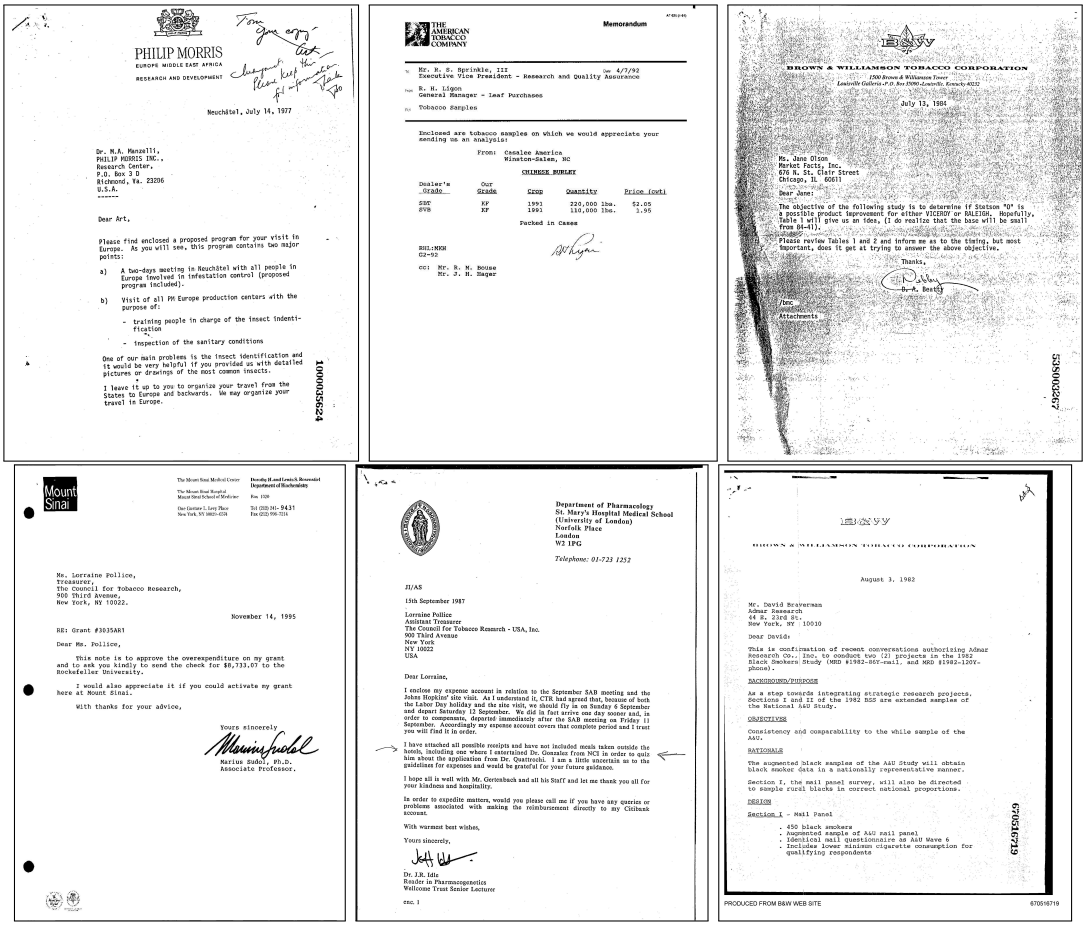


FIGURE A.1: Example documents of Tobacco-800 database.

FIGURE A.2: Some isolated logos extracted from Tobacco-800 database.

images are 113 logos with 6 different logo classes. Building this database aims to test our proposed logo spotting method in contemporary real documents which have complex layouts. We also create its ground-truth information as bounding box around logo regions. Figure A.3 and A.4 show examples of documents and logos of this database.

FIGURE A.3: Example documents of our collection dataset.

FIGURE A.4: Some logos of our collection dataset.

A.3 UW-III Database and PrimA Database

In Part II, to experiment on text/graphics separation, we used two well-known databases in this field. The first dataset is the standard University of Washington III (UW-III) database [114]. It contains 1600 binary scanned document images and its zone-level ground-truth is represented by bounding boxes. Those bounding boxes are labeled as non-text regions, page-header regions, text-body regions, page-footer regions, etc. In practice, we relabel those regions as text or non-text, because we are mainly concerned with the ability to distinguish text from non-text regardless of the type of non-text. Figure A.5 shows example documents from the UW-III dataset.

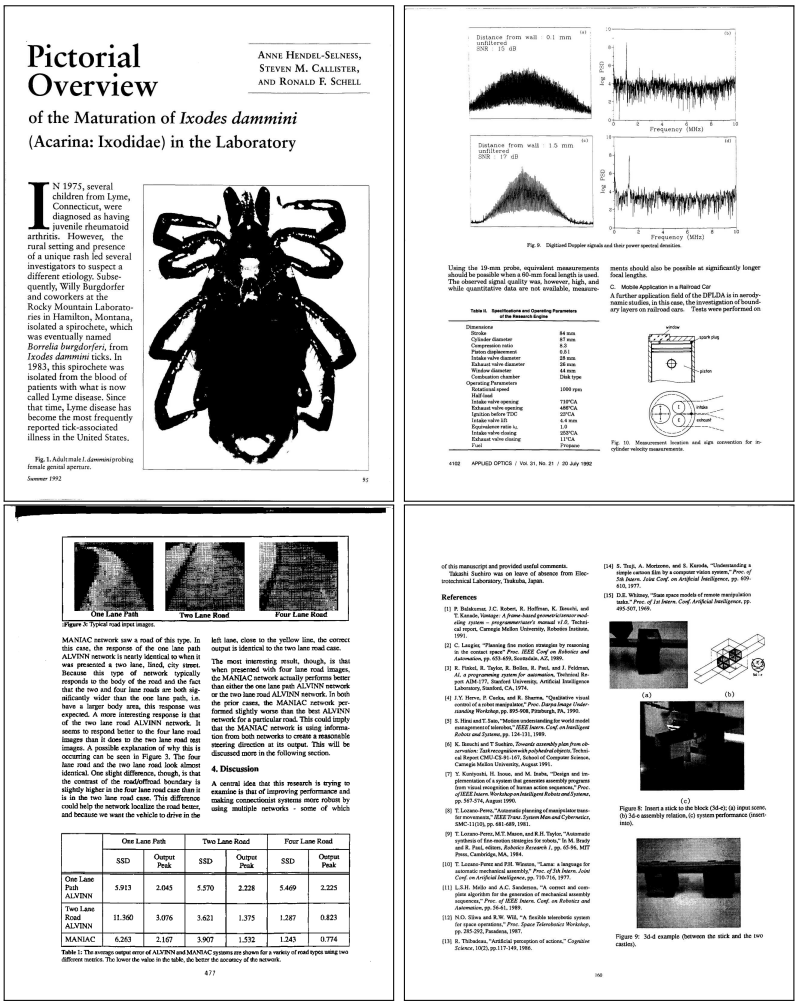


FIGURE A.5: Example documents of UW-III dataset, including text, image, graph, table and halftone regions.

The second dataset is ICDAR-2009 page segmentation competition dataset [115] for layout analysis of contemporary colored documents. In this dataset, there is a total of 55 images with different types of regions as discussed text, separator, graph, image, line art and noise. However, in our work, we only consider two classes: text and non-text to

FIGURE A.6: Example documents of ICDAR-2009 dataset and their region outlines.

perform segmentation. Therefore, we relabel the connected components in the ground truth text regions as text, and the components in all other regions regions as non-text. Figure A.6 shows example documents and their region outlines of ICDAR-2009 dataset.

Appendix B

List of Publications

This thesis has led to the following publications:

International Conference and Workshop Papers

- Viet Phuong Le, Muriel Visani, Cao De Tran, and Jean-Marc Ogier. "Logo spotting for document categorization." In Pattern Recognition (ICPR), 2012 21st International Conference on, pp. 3484-3487. IEEE, 2012.
- Viet Phuong Le, Muriel Visani, Cao De Tran, and Jean-Marc Ogier. "Improving logo spotting and matching for document categorization by a post-filter based on homography." In Document Analysis and Recognition (ICDAR), 2013 12th International Conference on, pp. 270-274. IEEE, 2013.
- Viet Phuong Le, Nibal Nayef, Muriel Visani, Jean-Marc Ogier, and Cao De Tran. "Document Retrieval Based on Logo Spotting Using Key-Point Matching." In Pattern Recognition (ICPR), 2014 22nd International Conference on, pp. 3056-3061. IEEE, 2014.
- Viet Phuong Le, Nibal Nayef, Muriel Visani, Jean-Marc Ogier, and Cao De Tran. "Text and non-text segmentation based on connected component features". In Document Analysis and Recognition (ICDAR), 2015 13th International Conference on, pp. 1096-1100. IEEE, 2015.
- Viet Phuong Le, and Cao De Tran. "Key-point matching with post-filter using SIFT and BRIEF in logo spotting." In Computing & Communication Technologies-Research, Innovation, and Vision for the Future (RIVF), 2015 IEEE RIVF International Conference on, pp. 89-93. IEEE, 2015.

- Quoc Bao Dang, Viet Phuong Le, Muhammad Muzzamil Luqman, Mickaël Coustaty, Cao De Tran, and Jean-Marc Ogier. "Camera-based document image retrieval system using local features - comparing SRIF with LLAH, SIFT, SURF and ORB". In the sixth International Workshop on Camera Based Document Analysis and Recognition, pp. 1211-1215, CBDAR 2015.

- Quoc Bao Dang, Muhammad Muzzamil Luqman, Mickaël Coustaty, Cao De Tran, Jean-Marc Ogier, and Viet Phuong Le. "Camera-based document image retrieval system for heterogeneous-content complex linguistic maps". In the eleventh International Workshop on Graphics Recognition, GREC 2015 (accepted).

- Viet Phuong Le, Quoc Bao Dang, and Cao De Tran. "Logo spotting on document images using local features". In the sixth International symposium on information and communication technology, SoICT 2015 (accepted).

Internal Journal Paper

- Viet Phuong Le and Cao De Tran. "Logo spotting on document images: document categorization and retrieval applications". In the Internal Journal of Can Tho University: , 2015 (in Vietnamese, accepted)

Bibliography

[1] Syed Saqib Bukhari, Al Azawi, Mayce Ibrahim Ali, Faisal Shafait, and Thomas M Breuel. Document image segmentation using discriminative learning over connected components. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, pages 183–190, 2010.

[2] Partha Pratim Roy, Josep Lladós, Umapada Pal, et al. *Multi-oriented and multi-scaled text character analysis and recognition in graphical documents and their applications to document image retrieval*. Universitat Autònoma de Barcelona,, 2011.

[3] Syed Saqib Bukhari, Faisal Shafait, and Thomas M Breuel. Improved document image segmentation algorithm using multiresolution morphology. In *IS&T/SPIE Electronic Imaging*, pages 78740D–78740D. International Society for Optics and Photonics, 2011.

[4] Marçal Rusiñol Sanabra. *Geometric and Structural-based Symbol Spotting. Application to Focused Retrieval in Graphic Document Collections*. 2009.

[5] Kenneth M Sayre. Machine recognition of handwritten words: A project report. *Pattern recognition*, 5(3):213–228, 1973.

[6] David Doermann and Karl Tombre. *Handbook of Document Image Processing and Recognition*. Springer Publishing Company, Incorporated, 2014.

[7] Toni M Rath and Raghavan Manmatha. Features for word spotting in historical manuscripts. In *Document Analysis and Recognition, 2003. Proceedings. Seventh International Conference on*, pages 218–222. IEEE, 2003.

[8] Toni M Rath and Raghavan Manmatha. Word image matching using dynamic time warping. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–521. IEEE, 2003.

[9] Shijian Lu and Chew Lim Tan. Retrieval of machine-printed latin documents through word shape coding. *Pattern Recognition*, 41(5):1799–1809, 2008.

[10] Josep Lladós and Gemma Sánchez. Indexing historical documents by word shape signatures. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, volume 1, pages 362–366. IEEE, 2007.

[11] Koichi Kise, Masaaki Tsujino, and Keinosuke Matsumoto. Spotting where to read on pages-retrieval of relevant parts from page images. In *Document Analysis Systems V*, pages 388–399. Springer, 2002.

[12] Nibal Nayef and Thomas M Breuel. On the use of geometric matching for both: Isolated symbol recognition and symbol spotting. In *Graphics recognition. new trends and challenges*, pages 36–48. Springer, 2013.

[13] Mathieu Delalandre, Ernest Valveny, and Josep Lladós. Performance evaluation of symbol recognition and spotting systems: An overview. In *Document Analysis Systems, 2008. DAS'08. The Eighth IAPR International Workshop on*, pages 497–505. IEEE, 2008.

[14] Marçal Rusiñol and Josep Lladós. *Symbol spotting in digital libraries: Focused retrieval over graphic-rich document collections*. Springer Science & Business Media, 2010.

[15] Marçal Rusinol and Josep Lladós. A region-based hashing approach for symbol spotting in technical documents. In *Graphics Recognition. Recent Advances and New Opportunities*, pages 104–113. Springer, 2008.

[16] Marçal Rusiñol, Josep Lladós, and Gemma Sánchez. Symbol spotting in vectorized technical drawings through a lookup table of region strings. *Pattern Analysis and Applications*, 13(3):321–331, 2010.

[17] Rashid Jalal Qureshi, Jean-Yves Ramel, Didier Barret, and Hubert Cardot. Spotting symbols in line drawing images using graph representations. In *Graphics Recognition. Recent Advances and New Opportunities*, pages 91–103. Springer, 2008.

[18] R Qureshi, J Ramel, and Hubert Cardot. Graphic symbol recognition using flexible matching of attributed relational graphs. In *proceeding of 6th IASTED International Conference on VIIP*, pages 383–388, 2006.

[19] Josep Llados and Gemma Sanchez. Graph matching versus graph parsing in graphics recognition—a combined approach. *International Journal of Pattern Recognition and Artificial Intelligence*, 18(03):455–473, 2004.

[20] Hervé Locteau, Sébastien Adam, Eric Trupin, Jacques Labiche, and Pierre Héroux. Symbol spotting using full visibility graph representation. In *Workshop on Graphics Recognition*, pages 49–50, 2007.

[21] Mickaël Coustaty, Stéphanie Guillas, Muriel Visani, Karell Bertet, and Jean-Marc Ogier. On the joint use of a structural signature and a galois lattice classifier for symbol recognition. In *Graphics Recognition. Recent Advances and New Opportunities*, pages 61–70. Springer, 2008.

[22] Salvatore Tabbone, Laurent Wendling, and Karl Tombre. Matching of graphical symbols in line-drawing images using angular signature information. *Document Analysis and Recognition*, 6(2):115–125, 2003.

[23] Salvatore Tabbone and Laurent Wendling. Recognition of symbols in grey level line-drawings from an adaptation of the radon transform. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 2, pages 570–573. IEEE, 2004.

[24] Salvatore Tabbone, Laurent Wendling, and Daniel Zuwala. A hybrid approach to detect graphical symbols in documents. In *Document Analysis Systems VI*, pages 342–353. Springer, 2004.

[25] Andrew D Bagdanov, Lamberto Ballan, Marco Bertini, and Alberto Del Bimbo. Trademark matching and retrieval in sports video databases. In *Proceedings of the international workshop on Workshop on multimedia information retrieval*, pages 79–86. ACM, 2007.

[26] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[27] Marçal Rusinol and Josep Llados. Logo spotting by a bag-of-words approach for document categorization. In *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on*, pages 111–115. IEEE, 2009.

[28] Marçal Rusinol, Vincent Poulain D'Andecy, Dimosthenis Karatzas, and Josep Lladós. Classification of administrative document images by logo identification. In *Graphics Recognition. New Trends and Challenges*, pages 49–58. Springer, 2013.

[29] Jerome Revaud, Matthijs Douze, and Cordelia Schmid. Correlation-based burstiness for logo retrieval. In *Proceedings of the 20th ACM international conference on Multimedia*, pages 965–968. ACM, 2012.

[30] Hichem Sahbi, Lamberto Ballan, Giovanni Serra, and Alberto Del Bimbo. Context-dependent logo matching and recognition. *Image Processing, IEEE Transactions on*, 22(3):1018–1031, 2013.

[31] The Anh Pham, Mathieu Delalandre, and Sabine Barrat. A contour-based method for logo detection. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 718–722. IEEE, 2011.

[32] Steve Seiden, Michael Dillencourt, Sandy Irani, Roland Borrey, and Timothy Murphy. Logo detection in document images. In *Proceedings of the International Conference on Imaging Science, Systems, and Technology*, pages 446–449. Citeseer, 1997.

[33] George Nagy and Sharad Seth. Hierarchical representation of optically scanned documents. In *Proceedings of International Conference on Pattern Recognition*, volume 1, pages 347–349, 1984.

[34] Guangyu Zhu and David Doermann. Automatic document logo detection. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, volume 2, pages 864–868. IEEE, 2007.

[35] Hongye Wang and Youbin Chen. Logo detection in document images based on boundary extension of feature rectangles. In *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on*, pages 1335–1339. IEEE, 2009.

[36] Rajiv Jain and David Doermann. Logo retrieval in document images. In *2012 10th IAPR International Workshop on Document Analysis Systems*, pages 135–139. IEEE, 2012.

[37] Yoav Freund, Robert E Schapire, et al. Experiments with a new boosting algorithm. In *ICML*, volume 96, pages 148–156, 1996.

[38] Enrico Francesconi, Paolo Frasconi, Marco Gori, Simone Marinai, JQ Sheng, Giovanni Soda, and Alessandro Sperduti. Logo recognition by recursive neural networks. In *Graphics Recognition Algorithms and Systems*, pages 104–117. Springer, 1998.

[39] Alessandro Sperduti and Antonina Starita. Supervised neural networks for the classification of structures. *Neural Networks, IEEE Transactions on*, 8(3):714–735, 1997.

[40] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50. Citeseer, 1988.

[41] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *Computer Vision–ECCV 2006*, pages 430–443. Springer, 2006.

[42] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: an efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2564–2571. IEEE, 2011.

[43] Stefan Leutenegger, Margarita Chli, and Roland Yves Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555. IEEE, 2011.

[44] Elmar Mair, Gregory D Hager, Darius Burschka, Michael Suppa, and Gerhard Hirzinger. Adaptive and generic corner detection based on the accelerated segment test. In *Computer Vision–ECCV 2010*, pages 183–196. Springer, 2010.

[45] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[46] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *Computer vision–ECCV 2006*, pages 404–417. Springer, 2006.

[47] Krystian Mikolajczyk and Cordelia Schmid. Indexing based on scale invariant interest points. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 525–531. IEEE, 2001.

[48] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image and vision computing*, 22 (10):761–767, 2004.

[49] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *Computer Vision–ECCV 2010*, pages 778–792. Springer, 2010.

[50] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. Freak: Fast retina keypoint. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 510–517. Ieee, 2012.

[51] Yan Ke and Rahul Sukthankar. Pca-sift: A more distinctive representation for local image descriptors. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–506. IEEE, 2004.

[52] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[53] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27(10):1615–1630, 2005.

[54] Koen EA Van De Sande, Theo Gevers, and Cees GM Snoek. Evaluating color descriptors for object and scene recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(9):1582–1596, 2010.

[55] Anna Bosch, Andrew Zisserman, and Xavier Muoz. Scene classification using a hybrid generative/discriminative approach. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(4):712–727, 2008.

[56] Joost Van de Weijer, Theo Gevers, and Andrew D Bagdanov. Boosting color saliency in image feature detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(1):150–156, 2006.

[57] Alaa E Abdel-Hakim, Aly Farag, et al. Csift: A sift descriptor with color invariant characteristics. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1978–1983. IEEE, 2006.

[58] Pablo F Alcantarilla, Luis M Bergasa, and Andrew J Davison. Gauge-surf descriptors. *Image and Vision Computing*, 31(1):103–116, 2013.

[59] Motilal Agrawal, Kurt Konolige, and Morten Rufus Blas. Censure: Center surround extremas for realtime feature detection and matching. In *Computer Vision–ECCV 2008*, pages 102–115. Springer, 2008.

[60] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, volume 96, pages 226–231, 1996.

[61] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[62] Martin A Fischler and Robert 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.

[63] Luc Vincent and Pierre Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE transactions on pattern analysis and machine intelligence*, 13(6):583–598, 1991.

[64] Foster J Provost, Tom Fawcett, et al. Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In *KDD*, volume 97, pages 43–48, 1997.

[65] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8): 861–874, 2006.

[66] Tuan D Pham. Unconstrained logo detection in document images. *Pattern recognition*, 36(12):3023–3025, 2003.

[67] David S Doermann, Ehud Rivlin, and Isaac Weiss. Logo recognition using geometric invariants. In *Document Analysis and Recognition, 1993., Proceedings of the Second International Conference on*, pages 894–897. IEEE, 1993.

[68] Francesca Cesarini, Enrico Francesconi, Marco Gori, Simone Marinai, JQ Sheng, and Giovanni Soda. A neural-based architecture for spot-noisy logo recognition. In *Document Analysis and Recognition, 1997., Proceedings of the Fourth International Conference on*, volume 1, pages 175–179. IEEE, 1997.

[69] Zhe Li, Matthias Schulte-Austum, and Martin Neschen. Fast logo detection and recognition in document images. In *Proceedings of the 2010 20th International Conference on Pattern Recognition*, pages 2716–2719. IEEE Computer Society, 2010.

[70] Marçal Rusiñol and Josep Lladós. Efficient logo retrieval through hashing shape context descriptors. In *Proceedings of the 9th IAPR International Workshop on Document Analysis Systems*, pages 215–222. ACM, 2010.

[71] Guangyu Zhu and David Doermann. Logo matching for document image retrieval. In *Document Analysis and Recognition, 2009. ICDAR'09. 10th International Conference on*, pages 606–610. IEEE, 2009.

[72] Marçal Rusiñol and Josep Lladós. A performance evaluation protocol for symbol spotting systems in terms of recognition and location indices. *International Journal on Document Analysis and Recognition (IJDAR)*, 12(2):83–96, 2009.

[73] Kwan Y. Wong, Richard G. Casey, and Friedrich M. Wahl. Document analysis system. *IBM journal of research and development*, 26(6):647–656, 1982.

[74] Oleg Okun, David Dœrmann, and Matti Pietikainen. Page segmentation and zone classification: the state of the art. Technical report, DTIC Document, 1999.

[75] M.A. Moll, H.S. Baird, and Chang An. Truthing for pixel-accurate segmentation. In *Document Analysis Systems, 2008. DAS '08. The Eighth IAPR International Workshop on*, pages 379–385, Sept 2008. doi: 10.1109/DAS.2008.47.

[76] Michael A Moll and Henry S Baird. Segmentation-based retrieval of document images from diverse collections. In *Electronic Imaging 2008*, pages 68150L–68150L. International Society for Optics and Photonics, 2008.

[77] Lloyd A. Fletcher and Rangachar Kasturi. A robust algorithm for text string separation from mixed text/graphics images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 10(6):910–918, 1988.

[78] Karl Tombre, Salvatore Tabbone, Loïc Pélissier, Bart Lamiroy, and Philippe Dosch. Text/graphics separation revisited. In *Document Analysis Systems V*, pages 200–211. Springer, 2002.

[79] F. Shafait, D. Keysers, and T.M. Breuel. Performance evaluation and benchmarking of six-page segmentation algorithms. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(6):941–954, June 2008.

[80] M.-W. Lin, Jules-Raymond Tapamo, and B. Ndovie. A texture-based method for document segmentation and classification. *South African Computer Journal*, 36: 49–56, 2006.

[81] Dan S Bloomberg. Multiresolution morphological approach to document image analysis. In *Proc. of the International Conference on Document Analysis and Recognition, Saint-Malo, France*, 1991.

[82] Horst Bunke and Patrick SP Wang. Handbook of character recognition and document image analysis, 1997.

[83] Ming-Kuei Hu. Visual pattern recognition by moment invariants. *Information Theory, IRE Transactions on*, 8(2):179–187, 1962.

[84] Krishna Subramanian, Prem Natarajan, Michael Decerbo, and David Castañòn. Character-stroke detection for text-localization and extraction. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, volume 1, pages 33–37. IEEE, 2007.

[85] Viet Cuong Dinh, Seong Soo Chun, Seungwook Cha, Hanjin Ryu, and Sanghoon Sull. An efficient method for text detection in video based on stroke width similarity. In *Computer Vision–ACCV 2007*, pages 200–209. Springer, 2007.

[86] Cheolkon Jung, Qifeng Liu, and Joongkyu Kim. A stroke filter and its application to text localization. *Pattern Recognition Letters*, 30(2):114–122, 2009.

[87] Xu-Cheng Yin, Xuwang Yin, Kaizhu Huang, and Hong-Wei Hao. Robust text detection in natural scene images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36(5):970–983, May 2014.

[88] John Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):679–698, 1986.

[89] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *Automatica*, 11(285-296):23–27, 1975.

[90] Omid Bonakdar Sakhi. *Segmentation of heterogeneous document images: an approach based on machine learning, connected components analysis, and texture analysis*. PhD thesis, Paris Est, 2012.

[91] Yoav Freund and Robert E Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In *Computational learning theory*, pages 23–37. Springer, 1995.

[92] Jerome Friedman, Trevor Hastie, Robert Tibshirani, et al. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The annals of statistics*, 28(2):337–407, 2000.

[93] Vladimir Vapnik. *The nature of statistical learning theory*. Springer Science & Business Media, 2013.

[94] Simon Haykin and Neural Network. A comprehensive foundation. *Neural Networks*, 2(2004), 2004.

[95] R. Smith. Hybrid page layout analysis via tab-stop detection. In *Document Analysis and Recognition, 2009. ICDAR '09. 10th International Conference on*, pages 241–245, July 2009.

[96] Trevor F Cox and Michael AA Cox. *Multidimensional scaling*. CRC Press, 2000.

[97] Brendan J Frey. *Graphical models for machine learning and digital communication*. MIT press, 1998.

[98] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics Springer, Berlin, 2001.

[99] Aapo Hyvarinen. Survey on independent component analysis. *Neural computing surveys*, 2(4):94–128, 1999.

[100] Jerome H Friedman, Jon Louis Bentley, and Raphael Ari Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software (TOMS)*, 3(3):209–226, 1977.

[101] Sunil Arya, David M Mount, Nathan S Netanyahu, Ruth Silverman, and Angela Y Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM (JACM)*, 45(6):891–923, 1998.

[102] Jeffrey S Beis and David G Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 1000–1006. IEEE, 1997.

[103] Chanop Silpa-Anan and Richard Hartley. Optimised kd-trees for fast image descriptor matching. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

[104] Ting Liu, Andrew W Moore, Ke Yang, and Alexander G Gray. An investigation of practical approximate nearest neighbor algorithms. In *Advances in neural information processing systems*, pages 825–832, 2004.

[105] David Nister and Henrik Stewenius. Scalable recognition with a vocabulary tree. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2161–2168. IEEE, 2006.

[106] Krystian Mikolajczyk and Jiri Matas. Improving descriptors for fast tree matching by optimal linear projection. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[107] Marius Muja and David G Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP (1)*, 2, 2009.

[108] Marius Muja and David G Lowe. Scalable nearest neighbor algorithms for high dimensional data. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36(11):2227–2240, 2014.

[109] Marius Muja and David G Lowe. Flann: Fast library for approximate nearest neighbors. URL http://www.cs.ubc.ca/research/flann.

[110] Marçal Rusinol, Dimosthenis Karatzas, and Josep Lladós. Spotting graphical symbols in camera-acquired documents in real time. In *Graphics Recognition. Current Trends and Challenges*, pages 3–10. Springer, 2014.

[111] Qin Lv, William Josephson, Zhe Wang, Moses Charikar, and Kai Li. Multi-probe lsh: efficient indexing for high-dimensional similarity search. In *Proceedings of the 33rd international conference on Very large data bases*, pages 950–961. VLDB Endowment, 2007.

[112] Friedrich M Wahl, Kwan Y Wong, and Richard G Casey. Block segmentation and text extraction in mixed text/image documents. *Computer graphics and image processing*, 20(4):375–390, 1982.

[113] Guangyu Zhu, Yefeng Zheng, David Doermann, and Stefan Jaeger. Multi-scale structural saliency for signature detection. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007.

[114] I Phillips. Users' reference manual. *CD-ROM, UW-III Document Image Database-III*, 1995.

[115] A. Antonacopoulos, S. Pletschacher, D. Bridson, and C. Papadopoulos. Icdar 2009 page segmentation competition. In *Document Analysis and Recognition, 2009. ICDAR '09. 10th International Conference on*, pages 1370–1374, July 2009.