Content-Based Document Image Retrieval using Sketch Queries

March 2016

Houssem Chatbri

Content-Based Document Image Retrieval using Sketch Queries

Graduate School of Systems and Information Engineering
University of Tsukuba

March 2016

Houssem Chatbri

Abstract

Document retrieval is a vital tool in many applications. In recent decades, content-based document retrieval emerged as an alternative to text-based retrieval due to its advantage of automatic annotation. In this thesis, I focus on visual documents and present a system for content-based document image retrieval (CBDIR). The proposed system takes sketch queries as input, allowing users to sketch their thoughts as queries. Advantages of this paradigm are multiple: It allows users to input several types of queries (e.g. drawing, mathematical expressions, diagrams, chemical equations). It performs automatic document annotation, which drastically saves labor time and effort. In addition, it permits cross-lingual retrieval since queries and document annotations do not depend on a particular language or script.

In order to build the system, research on several areas of pattern recognition has been conducted. First, I designed adequate data normalization which is necessary to transform raw data into a representation easier to process by matching and retrieval algorithms. Then, I introduced a novel shape descriptor that extracts keypoints from binary images by generating background information and using an objective measure of keypoint prominence. Afterwards, I proposed a spotting algorithm that is inspired from the human behavior, and rationalized with a theoretical model. Finally, I designed an indexing method that exploits the high redundancy of characters and symbols in document images. The indexing method is effective in dataset indexing, and it can be also used for dataset compression.

This thesis presents several contributions to the state of the art in pattern recognition, including a comparative study of contours and skeletons, novel and robust keypoint-based features, in addition to a general-purpose CBDIR method that is rationalized with a theoretical model. Evaluation of each building block of the system has been done using public datasets and comparison with state of the art methods. To assess the overall system, I used datasets of handwritten mathematical expression queries, as an initial application without loss of generality. Experimental results and comparative evaluation demonstrate the effectiveness of the proposed algorithms and their possible future improvement and extension.

Acknowledgments

I would like to express my deepest gratitude to my adviser, Prof. Keisuke Kameyama, for his guidance throughout my postgraduate studies and research. He forged my scientist personality with great patience and generosity. I have been so lucky to have him as my mentor. My achievements, if any, are owed firstly to him.

I am very grateful to my thesis committee, Profs. Ko Sakai, Kazuhiro Fukui, Hotaka Takizawa, and Itaru Kitahara for their precious time in reviewing my work.

I express my high gratitude to MEXT and I will always remain honored to be a Monbukagakusho laureate. I am also honored to be a student of the University of Tsukuba, where I found the perfect environment and the generous support.

I would like to thank my research collaborators, Profs. Paul Kwan and Richard Zanibbi, for their insightful suggestions and great support during my internships and beyond. My high gratitude goes to my former adviser, Prof. Kirmene Marzouki, for his continuous guidance since my undergraduate studies.

I dedicate this thesis to my parents Mohamed Sghaier Chatbri and Zina Mbarki Chatbri, for their tremendous support that I will never fulfill to reward, and their great wisdom that has taken me years to grasp. I also dedicate this thesis to my brother, Dr. Bassem Chatbri, who selfishly took all the good genes, and who always makes me proud.

Houssem Chatbri, March 2016

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In the recent decades, data digitization became an important trend that emerged due to the requirements of modern technologies and societal transformations. Digitization of documents allowed their easier access, safer storage, and their processing by data mining software[a]. Due to these advantages, huge projects are being carried out on data digitization, such as the Google Books Project, the Digital India Project[b], and the Large Scale Book Digitization Projects at Illinois USA[c], to name only a few.

To process digital data, computer software use application-specific formats to represent documents of various types (e.g. a scanned or a printed book, image, recording, video). Then, they are used in applications of document analysis [1, 2]. This thesis is concerned with visual documents and focuses on datasets of images. Usually, such datasets are indexed to allow their fast retrieval and efficient storage. Then, they are retrieved using online search systems. Retrieval requires document annotation, which is usually done using text labels. Afterwards, users retrieve documents using search engines such as Google and Yahoo!.

Image annotation is a crucial task for retrieval applications. In most existing systems, annotation is done manually by attributing text labels to images [3]. This paradigm, while being effective, suffers from three major limitations that hinder its application to truly big data: First, manual annotation is a labor-intensive and time-consuming task. Second, describing images with words is a subjective task that differs from one person to another and depends on the person's state of mind [4, 5]. Third, annotating images with text prevents cross-lingual retrieval. Fig. 1.1 illustrates some challenges of manual image annotation.

[a]New York Times; May 14, 2006; Scan This Book!
[b]https://digitizeindia.gov.in/
[c]http://www.library.illinois.edu/dcc/largescalebook.html

(a) (b)

Fig. 1.1: Challenges of manual image annotation: (a) Annotating the image with participant names would be a lot faster with automatic face recognition. (b) The portrait can be ambiguously interpreted as a young girl or an old woman [6].

Due to the aforementioned limitations of text-based image retrieval, content-based image retrieval (CBIR) was acknowledged as an interesting alternative of which the main advantage is automatic annotation using the image content. Research on CBIR traces back to 1992 with the pioneer work of Kato et al. [7] where they presented a system for sketch-based image retrieval (SBIR). The main advantage of SBIR over text-based systems and CBIR is allowing users to draw queries instead of typing them using a keyboard or presenting an example image. Since the work by Kato, SBIR became a hot research topic [8] and contributed in various applications such as finding criminal profiles using queries by a sketch artist [9], retrieving math documents using hand-drawn expression queries [10], etc. Fig. 1.2 shows an example of SBIR using the GazoPa system developed by Hitachi.

In this thesis, I present a system for content-based document image retrieval (CBDIR) using sketch queries. The system allows users to submit sketch queries in order to retrieve digital documents that are indexed offline. Sketches are binary images introduced by users using a sketching device, and documents images are binarized since the color information is irrelevant. Research has been conducted to find the adequate data representation, feature extraction, and matching algorithms to allow satisfactory performances. The main characteristics of the proposed system are being segmentation and recognition-free. Avoiding segmentation spares the system from erroneous image segmentation which affect retrieval performances, and avoiding recognition allows the system to be applicable to several types of queries (e.g. drawings, mathematical expressions, diagrams) that are not recognizable with optical character recognition (OCR).

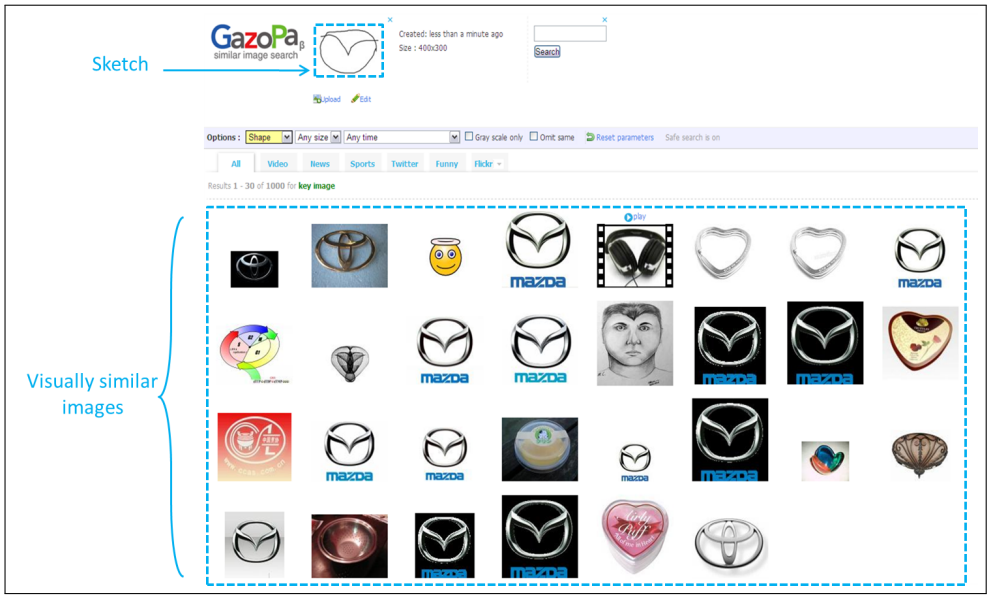


Fig. 1.2: SBIR using the GazoPa system by Hitachi.

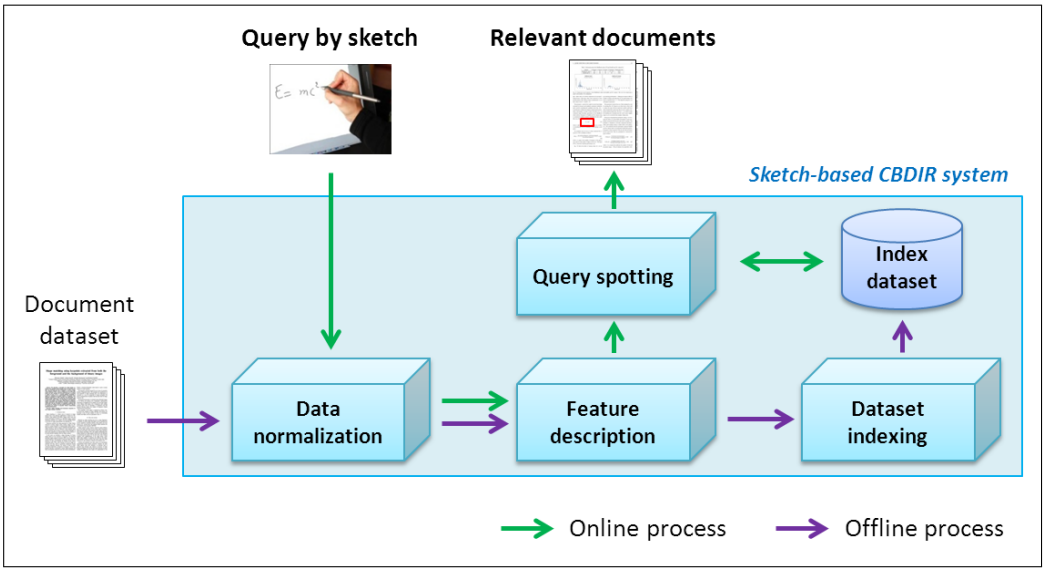


Fig. 1.3: Flowchart of the proposed system.

The proposed system is modular and operates as follows (Fig. 1.3): First, the user's query and the document are subjected to normalization which produces compact image representations that reduce the number of foreground pixels without altering the visual information. Next, features are extracted from the normalized images. Afterwards, the query is spotted inside the document using local and global feature matching. Indexing is used to produce an index dataset of the documents for the sake of efficiency.

Each building block of the system is evaluated using public datasets, and the overall performance is evaluated in CBDIR of mathematical expression queries using a dataset of handwritten mathematical expressions and printed document images, as an initial application without loss of generality. Experimental results and comparative evaluation demonstrated the effectiveness of the proposed algorithms and their possible future improvement and extension.

Research on CBDIR systems and sketching is underway and advances are reported by several research groups that specialize in different applications. André van der Hoek's group builds tablet and electronic whiteboard systems that support informal sketching by software designers [11]. Christoph Meinel's group works on lecture video indexing using OCR [12, 13]. David Doermann's group conducts research on CBDIR using queries as logos [14, 15]. John Collomosse's group presented methods for video retrieval using queries that indicate object motion in the form of sketched arrows [16]. Richard Zanibbi's group focuses on CBDIR using mathematical expression queries that are snapshots [17] or handwritten [10]. This thesis' contributions to the state of the art mainly include designing an algorithm for CBDIR that is not application-specific [18], robust shape features [19], in addition to adequate image preprocessing [20] and indexing [21].

In the remainder of this thesis, each chapter is devoted to a building block of the proposed system (Fig. 1.3), with a state of the art review, originally proposed methodology and experimental evaluation. In Chapter 2, I report a comparative study of contours and skeletons in order to determine the adequate compact representation for my application. In Chapter 3, I introduce a novel keypoint descriptor that is based on background information generation and a measure of keypoint prominence. In Chapter 4, I rationalize query spotting in document images with a theoretical model, and present an algorithmic implementation to adapt the model to noisy and fluctuated handwritten images. In Chapter 5, I present a method for dataset indexing using connected components clustering. Finally, the summary and conclusions of this research, and my out-take on its future applications are reported in Chapter 6.

Chapter 2

Data normalization by compact representation

In numerous pattern recognition tasks, it is routine to produce an intermediate image representation with reduced dimensionality in order to filter out noise and remove redundant information [22]. In case of binary images, contours and skeletons are widely accepted as adequate representations for subsequent classification and matching algorithms using statistical shape features [23, 24]. Both representations share the common and interesting property of encoding the visual information of an object by using a limited set of points, and both have been used for interest points detection (e.g. curvature points, corners, etc.).

While contour extraction is a trivial task in binary images that involves detecting an object's pixels located on the boundary, skeletonization requires more complex algorithms in order to extract the skeleton of an object, which is the locus of the symmetric points of the local symmetries of the shape [25, 26]. Algorithms for skeletonization have also been called medial axis transformations, with the resulting skeleton being known as medial line or medial axis [27].

Skeletons are an interesting object representation due to their ability to capture various aspects of shapes. Not only could they give access to both object interiors and object boundaries, but they can also provide rich geometrical relationships among objects [23, 28]. In the literature, it is often assumed that skeletons are particularly suitable for describing elongated shapes [28–31]. At the same time, the skeletal representation is also an active research area in the cognitive science community for its role in human perception [32]. However, there has been no systematic study to date for assessing the strength of skeletons in different types of image variations, that include both digital (e.g. noise, blurring, etc.) or biologically plausible variations (e.g. size, morphing, etc.).

In this chapter, the performances of contours and skeletons as object representations for shape matching are compared. Shape matching experiments are conducted on test image datasets involving thick, elongated, and nearly thin objects. In addition, different image variations like contour noise, blurring, and size reduction are generated. Finally, their matching performances are evaluated using objective and commonly adopted metrics.

The remainder of this chapter is organized as follows: In Sec. 5.1, the related literature on contour detection and skeletonization, selected applications in shape matching, and related studies in cognitive science are overviewed. Sec. 2.2 presents my experimental platform, including the shape feature descriptor, the shape matching algorithms and the metrics. Experimental evaluation and results are discussed in Sec. 2.3. Lastly, concluding remarks are given in Sec. 2.4.

2.1 Related work

In this section, the focus is on the literature related to contour detection and skeletonization in 2D binary images, shape matching, and related studies in cognitive science.

2.1.1 Contour detection and skeletonization

Contour detection has been an early concern for image processing researchers for its necessity in multiple applications [22]. In case of binary images, contour detection can be easily achieved since the contour corresponds to boundary points that have at least one background neighbor. Furthermore, sophisticated algorithms have been presented to improve processing efficiency [33, 34].

On the other hand, Skeletonization is a more complicated task that can be done using one of several approaches. *Mathematical morphology* operations such as *erosion* and *dilation* can be used to produce a skeleton [35]. *Thinning* refers to the category of algorithms that perform iterative removal of boundary pixels in a way that preserves topology and connectivity [28]. The final skeleton should be centered within the object and reflects its geometrical features. Another approach is based on the *Distance Transform (DT)*, which is a replica of the object, where the foreground elements are replaced with their distances from the background [36]. The DT is interpreted as a landscape, where the label of a pixel indicates its height. Then, the skeletal elements can be directly identified by using pixel neighborhood analysis. Skeletons can also be extracted by using *Voronoi diagrams* [37]. The principle of

a Voronoi diagram is to determine for each object point p_i a region that contains the points which are closer to p_i that to the remaining object points $\{p_j\}_{j\neq i}$. The resultant Voronoi diagram corresponds to the locus of regions borders. Usually, a set of object points is sampled and used to compute the Voronoi diagram, and a post-processing step is anticipated to prune non-necessary branches of the skeleton [38].

A skeletonization algorithm is considered desirable if it meets the following properties [28, 39]:

- Produce a thin or nearly thin skeleton,
- Preserve the connectivity of the original object, which means that connected parts in the original object should stay connected in the skeleton,
- Preserve the visual topology of the original object, which means that although the skeleton is a compact representation of the original object, it should deliver the same visual information.
- Be robust against contour noise.

Research on improving skeletonization algorithms focused on improving efficiency [40], insuring topology preservation [41, 42], and adding robustness against noise using filtering [43, 44] and pruning [45]. Application-specific algorithms have also been presented such as extracting skeletons of handwriting, fingerprints, and cereal plant images [43, 46]. Comparison of skeletonization algorithms can be found in [28, 47].

It is generally accepted that skeletons are a suitable representation for elongated shapes [28–31]. On the other hand, skeletonization of thick images produces skeletons that look very differently from the original objects, which often appears counterintuitive to the human observer [48]. In this case, contours would be a more suitable representation (Fig. 2.1).

2.1.2 Shape matching

Shape matching is an old central problem of pattern recognition that is concerned with matching an image against another image or against a template [49, 50]. Usually, the input image is subjected to a preprocessing step for noise removal and dimensionality reduction, then shape features are extracted and used for matching. In case of binary images, shape features are essential because of the absence of other

Table 2.1: Average statistics of the image datasets.

Dataset	# classes	Image per class	Image size	% object pixels	% contour pixels	% skeleton pixels
Thick (197 images)	20	[8,10]	768×576	104,200	2.02%	0.7%
Elongated (174 images)	20	[6,10]	768×576	41,616	4.09%	1.61%
Nearly thin (200 images)	20	10	888×276	10,747	44.64%	16.7%

features (e.g. color) [51]. In addition, behavioral studies have shown that shape features are preferable to humans even when other features are available [52]. In this section, I focus on methods that use statistical shape features. For information on different paradigms, I refer the reader to [24, 53].

Contours and skeletons have been widely used in shape matching as a dimensionality reduction step prior to feature extraction. Skeletons have been used in a variety of applications including optical character recognition (OCR) [54], document image analysis [55], biometric authentication [56], medical imaging [57], signature verification [58], CBIR using sketch queries [59], etc. In OCR, skeletons are used as a normalized representation to insure invariance to pen thickness and handwriting styles. In fingerprint and retinal images, skeletonization is used in order to produce one-pixel-wide objects whose geometrical and topological properties are input to the identification process. In document analysis, signature verification and CBIR using sketch queries, skeletonization is used as a preprocessing and normalization step.

Contours are a widely used representation that dramatically reduces the number of points, yet preserves the shape visual information [60]. Consequently, they have been used in different applications including shape classification and matching [60, 61], CBIR using sketch queries [7], shape matching using interest points detection [62], etc. Xu et al. argue that contours are the best features for content-based retrieval of spine X-ray images [63].

Few studies have tackled the comparison between contours and skeletons in shape matching problems. Leung and Chen preferred using contours of thick objects and skeletons of elongated objects, in the context of trademark image retrieval [64]. Chalechale et al. used contours of colored images in a database and skeletons of hand-drawn user queries for CBIR [65].

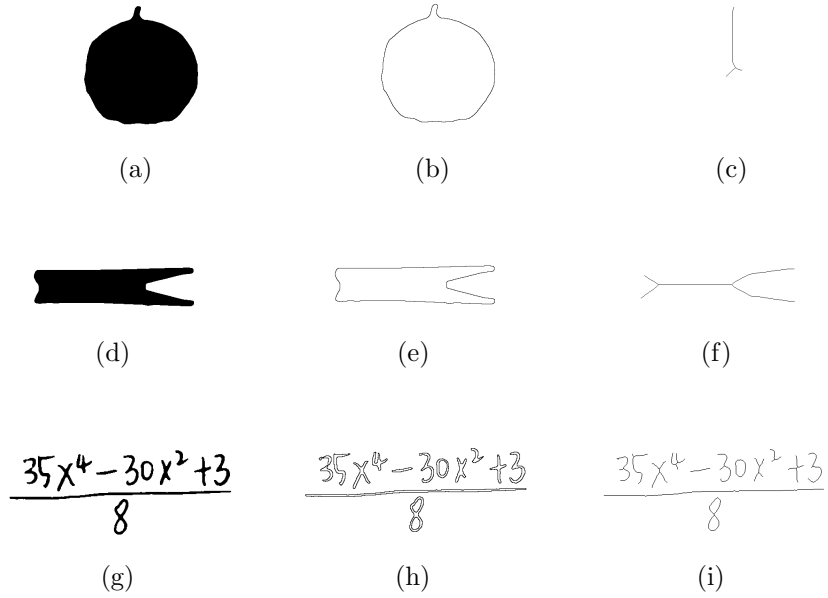


Fig. 2.1: Datasets and different image representations: The first row shows images from Dataset 1 (thick objects), the second row shows images from Dataset 2 (elongated objects), and the third row shows images from Dataset 3 (nearly thin objects). The first column shows original images, the second column shows contour images, and the third column shows skeleton images.

2.1.3 Related studies in cognitive science

Contours and skeletons belong to an important research topic in cognitive science, and many studies have focused on their neural representations and roles in human perception [25].

A number of studies have theoretically favored skeletons in object recognition [32]. However, few physiological studies have reported supportive results [66].

Recently, several studies provided experimental evidence for the response of neurons at the primary visual cortex (V1) to contours and skeletons [67–70]. Although a retinal image of an object often includes noise on the contour, the physiological based skeletal representation appears to be robust against noise [70]. In contrast, engineered skeletons are sensitive to contour noise. To my knowledge, no studies have been proposed to compare the two representations for human perception.

2.2 Experimental platform for compact representation

The experimental platform is generic and consists of extracting shape features from an intermediate image representation, and then use them as input to a shape matching algorithm. The algorithm's performance is evaluated using objective metrics.

In the following, the image representations are presented in Sec. 2.2.1, the shape feature descriptor in Sec. 2.2.2, and the shape matching algorithms and evaluation metrics in Sec. 2.2.3.

2.2.1 Image representations

Contours and skeletons are extracted using the following procedures: Pixel neighborhood analysis is used for contour extraction, where a pixel is considered a contour pixel if it has at least one background neighbor. Skeletons are extracted using a thinning algorithm [71]. The thinning algorithm extracts skeletons by applying successive iterations of rule-based boundary pixel removal. The final result is a skeleton that preserves connectivity and topology of the original object. Fig. 2.1 illustrates the image representations.

Different algorithms can be found in the literature and they are endowed with robustness against image variations or tuned for specific applications (Sec. 2.1.1). For instance, contour simplification and skeleton pruning have been used to remove irrelevant branches caused by contour perturbations [45], and scale-space filtering has been applied for noise-robust skeletonization by optimal filtering scale selection [44]. In the present work, I use the procedures mentioned above since my goal is to investigate the effect of image variations on shape matching and not to neutralize it.

2.2.2 Shape feature descriptor

The histogram of pixel distributions in polar coordinates was used as a feature descriptor. Features are extracted by calculating the distances and angles of pixels inside a circular layout located at the shape centroid (Fig. 3.4).

The similarity between two images I_1 and I_2 is expressed by the *Histogram Intersection* measure, S, computed from their corresponding histograms. S is cal-

culated as follows:

$$S = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} min(H_{ij}^1, H_{ij}^2) \tag{2.1}$$

where H^1 and H^2 are the histograms corresponding to images I_1 and I_2, and M and N are the histogram dimensions.

This descriptor is efficient and global, which makes it conform more with human perception [72].

2.2.3 Shape matching algorithms and metrics

I used two shape matching algorithms: object classification (OC) and content-based image retrieval (CBIR). OC performances are evaluated using the *Classification Accuracy* metric, and CBIR performances are evaluated using the *F-Measure* metric.

Object classification

OC is done using a K-Nearest Neighbors algorithm. Given an image I, the corresponding feature histogram H is matched against the N images of the dataset. Then, the K most similar images are used to identify the class of I in a majority voting manner. A rejection class is attributed to I if the number of majority votes is less than $K \times 40\%$. The algorithm's performance is estimated using the *Classification Accuracy* metric, that is expressed as a percentage and calculated as follow:

$$Classification\ Accuracy = 100 \times \frac{1}{N} \sum_{k=1}^{N} score_k \tag{2.2}$$

where $score_k$ takes 1 when the relevant class has the majority voting, and 0 otherwise.

CBIR

Similarly to OC, the N_R most similar images to a query I are retrieved. Then, the retrieval performance is estimated using the $F\text{-}Measure$ metric, that is expressed as a percentage and calculated as follows:

$$F\text{-}Measure = 100 \times \frac{1}{N} \sum_{k=1}^{N} \frac{2 \times precision_k \times recall_k}{precision_k + recall_k} \tag{2.3}$$

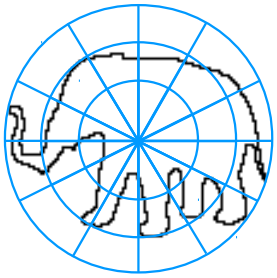


Fig. 2.2: Feature extraction layout.

where $precision_k$ and $recall_k$ are calculated as follows:

$$precision_k = \frac{number\ of\ retrieved\ images\ of\ class\ k}{N_R} \tag{2.4}$$

$$recall_k = \frac{number\ of\ retrieved\ images\ of\ class\ k}{|class_k|} \tag{2.5}$$

Precision expresses the ability to find relevant instances among the retrieved images, and *recall* expresses the ability to find all relevant images of the image class. CBIR is concerned with *precision* and *recall* of retrieval, while OC takes into account only the majority appearance inside the retrieval set.

2.3 Experimental results

In this section, I present the image datasets I used (Sec. 2.3.1), the image variations I analyzed (Sec. 2.3.2), and the results (Sec. 2.3.3).

2.3.1 Image datasets

The image datasets include:

- Dataset 1: Images of *thick* objects.
- Dataset 2: Images of *elongated* objects.
- Dataset 3: Images of *nearly thin* objects.

Images of Dataset 1 and Dataset 2 have been collected from the ALOI dataset [73], while images of Dataset 3 were chosen from Zanibbi and Yu's dataset [10]. The image datasets can be obtained from the authors. Fig. 2.1 shows examples of the images and their derived representations, while Table 2.1 presents a summary of the

datasets. It can be seen from Table 2.1 that the percentage of points in contours and skeletons is smallest in thick images, and largest in nearly thin images.

2.3.2 Image variations

I analyze the effect of contour noise, blurring, and size reduction. Contour noise is a common challenge for skeletonization algorithms [44] and a biologically plausible image variation [70]. In my experiments, additive contour noises were generated by randomly removing a percentage of contour points that ranges from 0% (original images) to 100% (all contour points removed). Additive border noise is expected to dramatically alter the structure of skeletons [68]. This synthetically simulates the binary images affected by border noise that are often produced by scanners or sketching devices [74]. Fig. 2.3 shows samples of the noisy images and their corresponding representations.

Image blurring is a widely used image processing procedure and has been used to as a remedy to contour noise [43, 44, 75]. In this experiment, I generate multiple blurred images using a Gaussian filter of scale σ that ranges from 0 (original images) to 15. This range of blurring scales insures an interval of fine to coarse. After blurring, image binarization is applied using a standard algorithm [76]. Fig. 2.5 shows samples of the blurred images and their corresponding representations.

Size is a biologically plausible property that is frequently changed [77]. Size change affects the level of details existing in an image; Large images contain more local details, while smaller images have less local details but preserve global details. Images of reduced sizes were generated with a size reduction factor that ranges from 1 (original images) to 5 using a pixel area re-sampling algorithm [78]. This range insures a study of the performances on image that vary from large to small. Fig. 2.7 shows samples from the generated images.

2.3.3 Results

Throughout the experiments, the parameter K in *Classification Accuracy* is set to $K = \sqrt{N}$ where N is the total number of images in the dataset, as suggested in [79]. The parameter N_R used in $F\text{-}Measure$ takes the same value as K. The feature histogram dimensions are set to $M = 5$ and $N = 12$, respectively.

Figures 2.4, 2.6, and 2.8 show the results of OC and CBIR using the different representations exposed to noise, blurring, and size reduction. Contours outperformed skeletons in most cases regardless of the image category and variation. The performances expressed in OC are higher than in CBIR due to the

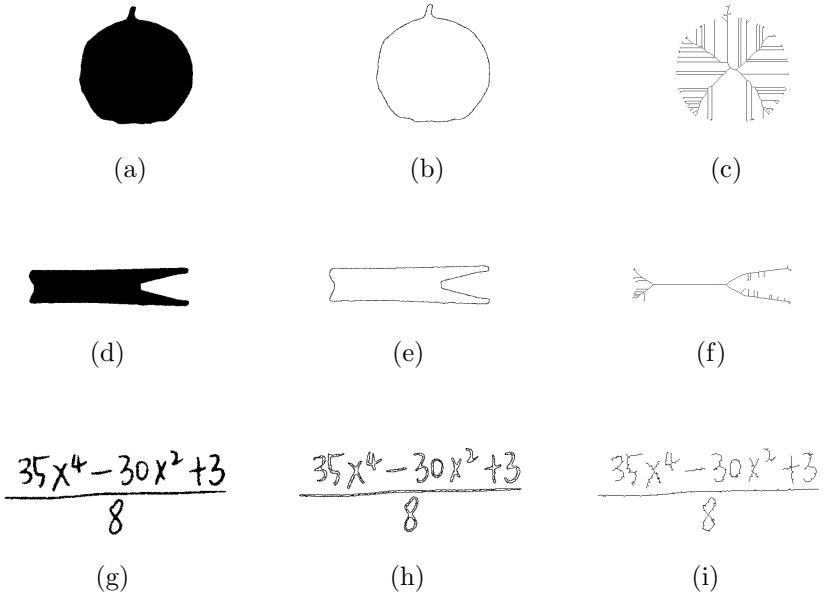


Fig. 2.3: Effects of noise. Images of Fig. 2.1 with 20% additive contour noise and their corresponding representations.

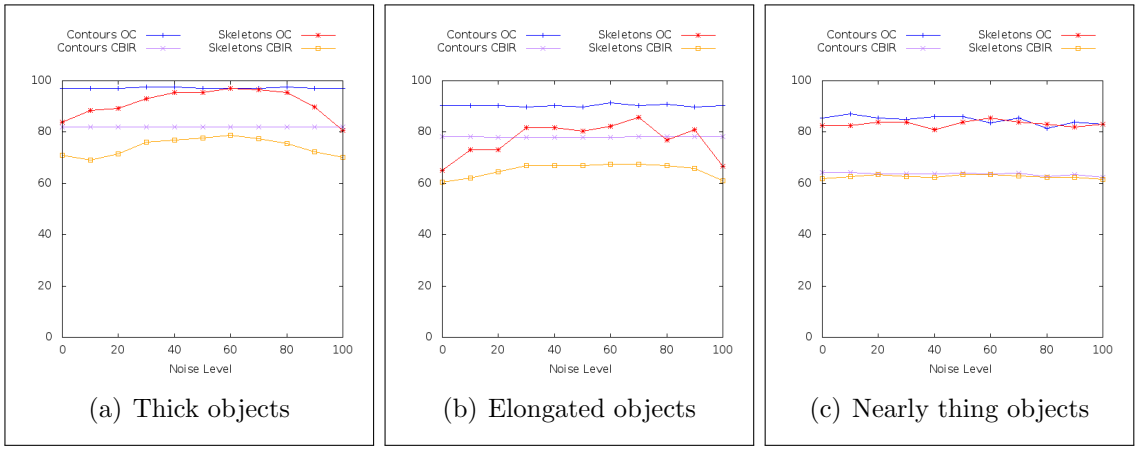
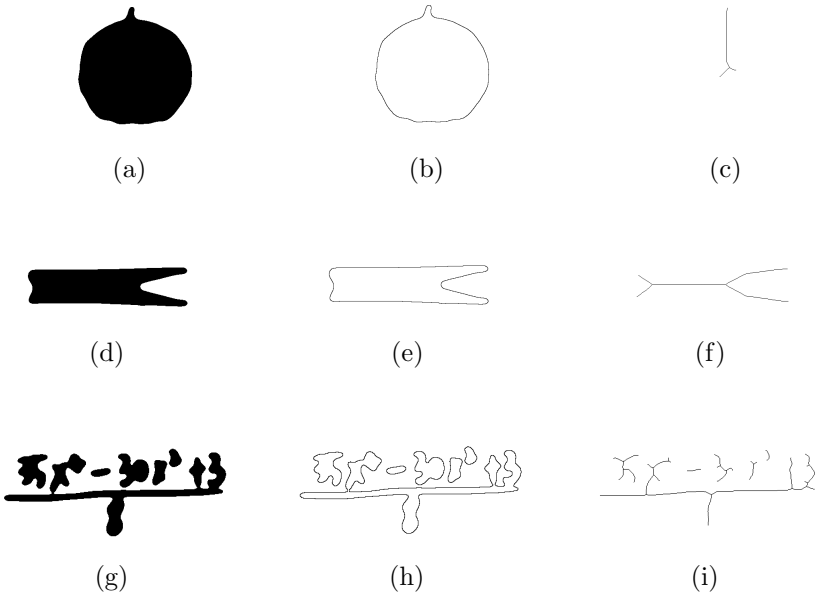


Fig. 2.4: Performances in presence of noise.

intrinsic difference between the corresponding metrics *Classification Accuracy* and *F-Measure* respectively. *Classification Accuracy* considers the number of relevant images in the voting pool, while *F-Measure* takes into account also the precision and recall of retrieval. For instance, say I have an image I of class C that contains 10 instances in total, and a number of voters $K = 5$ where 3 belong to C. In this case, $Classification\ Accuracy = 100\%$ as C gets the majority vote, while $F\text{-}Measure = 40\%$ (since $precision = \frac{3}{5}$ and $recall = \frac{3}{10}$).

Contours are stable in presence of noise regardless of the image category. In

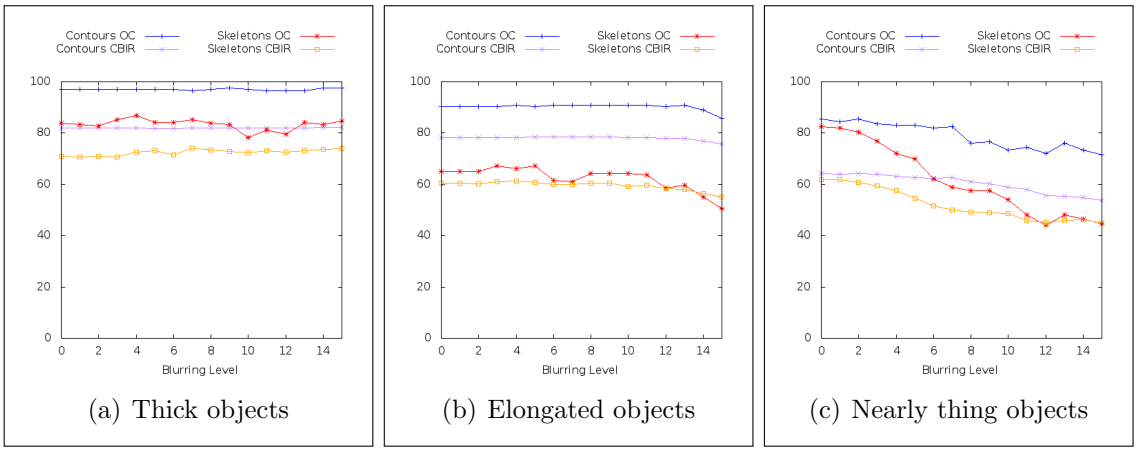


(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Fig. 2.5: Effects of blurring. Images of Fig. 2.1 after blurring using a Gaussian filter of scale $\sigma = 5$ and binarization, and their corresponding representations.



(a) Thick objects (b) Elongated objects (c) Nearly thing objects

Fig. 2.6: Performances in presence of blurring. The *Blurring Level* corresponds to the Gaussian filter scale σ.

case of blurring, performances remain stable with respect to a moderate amount of blurring, and start decreasing when the blurring becomes significant. In case of size reduction, performances remain stable.

Skeletons noticeably improve in the presence of noise. The improvement is particularly noticeable in case of thick and elongated images. In case of blurring, skeletons' stability seems to be dependent on object thickness. Large amount of blurring cause significant shape alterations on nearly thin objects, particularly be-

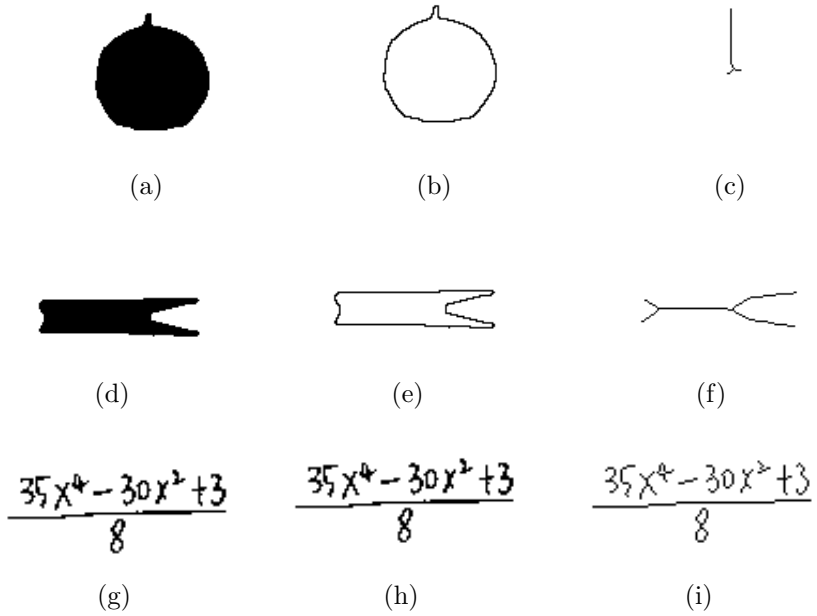


Fig. 2.7: Effects of size reduction. Images of Fig. 2.1 after size reduction by a factor of 4, and their corresponding representations.

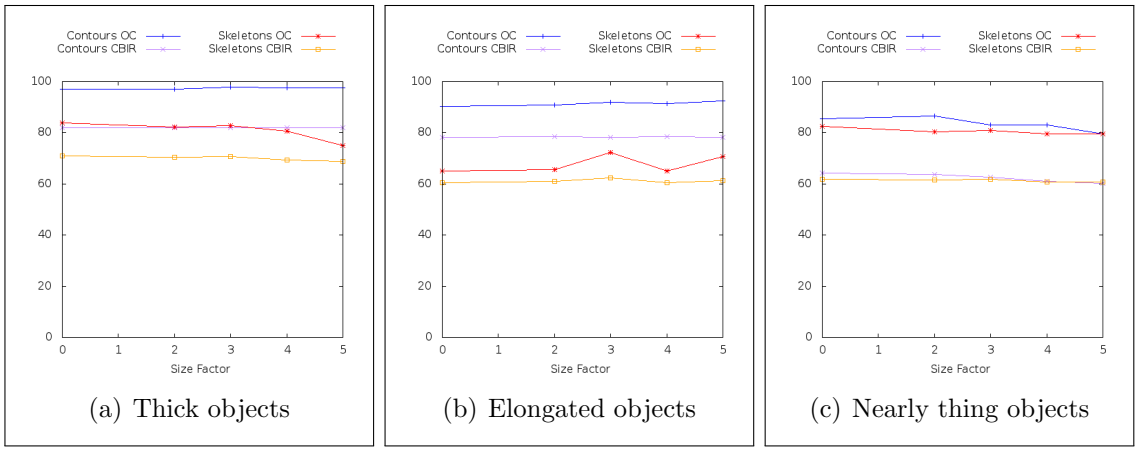


Fig. 2.8: Performances in presence of size reduction.

cause they are multi-component objects, and hence cause performances to decrease. In case of size reduction, performances remain stable.

I observe that the performances of contours and skeletons track each other consistently in case of large amount of blurring and size reduction. In these cases, the delivered skeletons and contours are distorted and lose local details which leads to decreasing performances in both representations. In case of noise, skeletons are more sensitive than contours and the performances by the two representations evolve

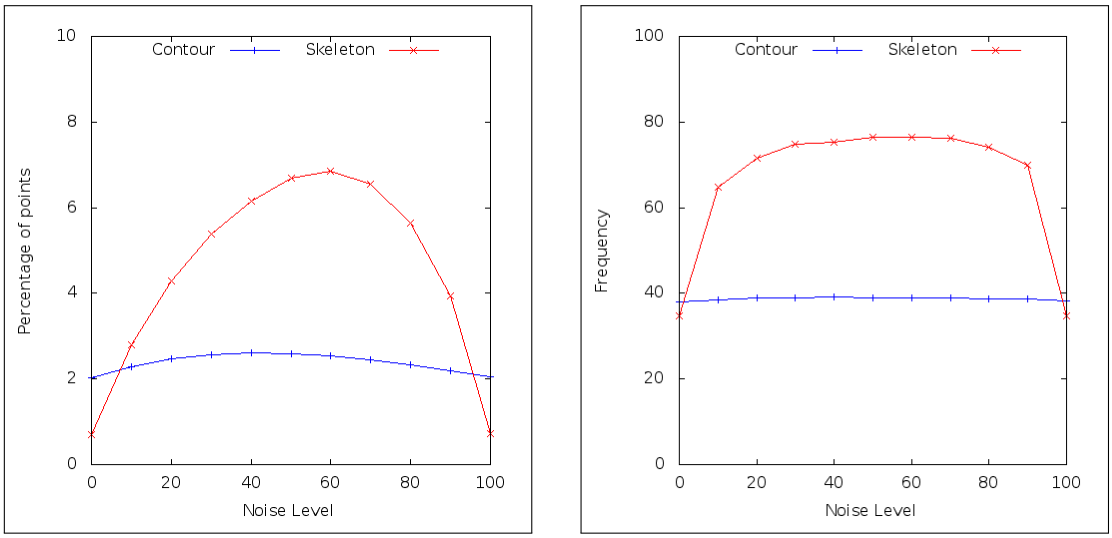


Fig. 2.9: Percentage of skeleton points and image spatial frequency as function of the level of noise corresponding to skeletons extracted from thick images.

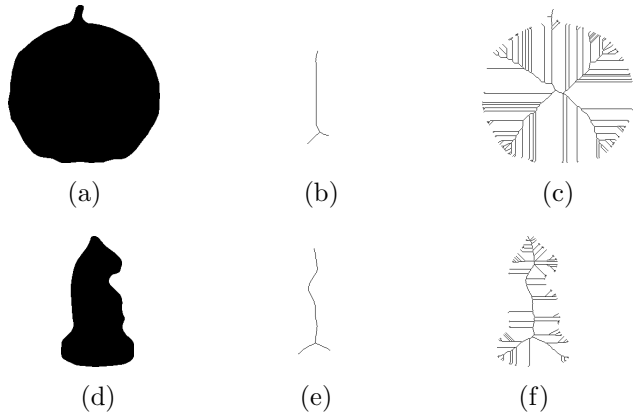


Fig. 2.10: Uniqueness of skeletons: The objects in images (a) and (d) are visually dissimilar, yet their respective skeletons in images (b) and (e) are visually similar. Adding noise increased dissimilarity between the skeletons in images (c) and (f).

differently.

The improvement of skeletons' performances in presence of noise is an interesting finding, as it is often assumed that noise is counter-productive for shape matching. When contour noise is added, branches appear in locus of contour noise and result in increasing the number of skeleton points (Fig. 2.9(a)) and the image spatial frequency (Fig. 2.9(b)). The spatial frequency is approximated by the average percentage of active histogram bins. This shows that additive contour noise results in skeletons that recover more spread feature points from the original object, and hence improve the uniqueness property of skeletons (Fig. 2.10). In the contrary, noise does not significantly affect the contour representation.

The effect of noise on the dataset distribution is estimated by calculating the

Distinctiveness as a function of the level of noise. The *Distinctiveness* measure is calculated as follows:

$$Distinctiveness\ Measure = \frac{\sigma_{inter} - \sigma_{intra}}{\sigma_{inter} + \sigma_{intra}} \tag{2.6}$$

where σ_{inter} and σ_{intra} refer to the standard deviations of inter-class similarity and intra-class similarity. Fig. 2.11 shows the plot of *Distinctiveness Measure* in case of thick objects subjected to noise. When the noise level increases, *Distinctiveness Measure* corresponding to skeletons increases which indicates that feature vectors are clustering and that these clusters are spreading away from each others. On the other hand, increasing the level of noise does not affect *Distinctiveness Measure* of contours.

I relate my results to the field of CBIR and classification of medical images for aiding physicians' diagnosis. Specifically, content-based spine X-ray image retrieval is a particularly challenging research problem due to poor image quality and low inter-class variations [63, 80, 81]. There, shape is the main feature due to absence of other image cues, and existing methods for shape representation and matching seem to rely essentially on contours [63, 81]. Due to ambiguous vertebral boundaries, segmentation is a difficult task and the resulted regions usually have contours with local perturbations [82]. Using boundary filtering for perturbation reduction is nontrivial because local changes in the contour might indicate important features that are crucial to effective retrieval or lesion detection [81]. Antani et al. made a comparison between different shape representations on a set of 250 vertebra boundary images and the best result was a 55.94% performance score [81], which is an indicator of the problem difficulty.

I notice the absence of using skeletons in the field of content-based retrieval of spine X-ray images. My results show that skeletons are more sensitive to boundary changes than contours and presence of boundary perturbations may even improve matching performances. Skeletons may be able to emphasize the local changes more than contours and hence overcome the small inter-class variations. Also, filtering can be applied in a scale-space fashion for optimal skeletonization. In such a scenario, a single skeleton may be selected according to an objective metric from multiple candidates produced from filtered images of different levels.

2.4 Conclusion

In this chapter, I reported a comparative evaluation of contours and skeletons as shape representations used for statistical feature extraction. Despite the widespread

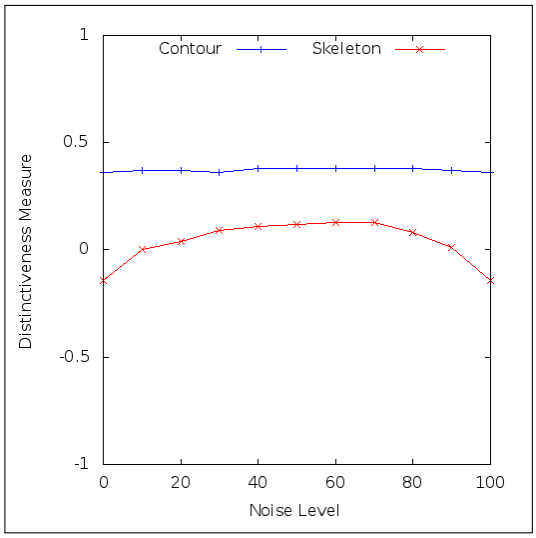


Fig. 2.11: Distinctiveness Measure as function of the level of noise corresponding to thick images.

of these representations in pattern recognition and their importance in human perception, no existing studies to compare their performances have been reported.

Different image datasets including thick, elongated, and nearly thin images were prepared. Moreover, various image variations including noise, blurring, and size reduction were generated. Performance evaluation is done using object classification and content-based image retrieval algorithms, and measured by objective and commonly adopted metrics.

Results indicate that contours outperform skeletons and that they are stable in case of moderate image variations. In addition, a noteworthy finding is the improvement of skeletons in the presence of noise, due to increase in the number of skeleton points and the image spatial frequency.

This study highlighted the beneficial nature of investigating these two shape representations in pattern recognition and cognitive science. I hope my findings can contribute in the following goals:

- Guide the choice of an image representation according to its performances using a class of images subjected to image variations that I investigated. I conjecture that using skeletons and scale-space filtering can be beneficial for characterizing some classes of medical images.

- Improvement observed in skeletons in the presence of noise may lead to the design of novel shape descriptors. Contrary to general belief about skeletons, I observed that the branches created in locus of noise could improve distinctiveness which leads me to think that they contain significant clues for distinction.

- Since contours and skeletons are interesting topics for researchers on cognition, my findings could become a relevant feedback on the computational importance of these representations in image matching.

The findings of this study, related to the supremacy of contours over skeletons, will be exploited in the subsequent chapters.

Chapter 3

Feature description by keypoints

Shape matching is a vibrant area of research on image analysis due to the numerous applications it allows [83]. Particularly, when dealing with binary images where color and texture information are absent (e.g. silhouette images, scanned documents, sketches, etc.), shape is the only available feature to be used for image representation and matching [51].

Numerous methods have been presented for shape feature extraction in binary images [24, 84]. Usually, images are subjected to contour detection or skeletonization before feature extraction in order to remove redundant information and reduce processing time [85]. Moreover, some methods select certain *keypoints* and use them to extract features [61, 86–93]. In these cases, keypoints are selected based on their saliency or by using uniform sampling from the shape contours.

Due to the absence of background information in binary images, keypoints are extracted from the foreground pixels (i.e. regions, contours, or skeletons) and the background is omitted. In this work, I introduce a shape descriptor that approaches the problem differently by generating background information in binary images. The main steps of the descriptor are the following:

- Keypoint extraction: An image transformation is used to generate background information on the original binary image. Then, keypoints are extracted from the transformed image using point local area analysis.

- Keypoint selection: An objective measure of keypoint prominence is used to automatically select the most important keypoints and filter out the redundant and sensitive ones.

- Feature representation: A feature vector is calculated for each keypoint by using the distributions of contour points in the local area of the keypoint.

I report evaluation of BIK using silhouette images of the Kimia 216 dataset [94] and the MPEG-7 CE-shape-1 part B [95], handwritten mathematical expressions of Zanibbi and Yu's dataset [10], hand-drawn diagram sketches of Liang et al.'s [96], and noisy scanned logo images of the Tobacco 800 dataset [14]. Experimental results on various types of images and a comparative evaluation demonstrate that BIK is competitive compared with state-of-the-art methods.

The remainder of this chapter is organized as follows: Sec. 5.1 reviews key methods of shape matching. I present my descriptor in Sec. 3.2 and evaluate it in Sec. 3.3. Concluding remarks and future work are presented in Sec. 5.4.

3.1 Related work

Research on shape matching has led to a large repository of methods [84] where shape descriptors can be classified into methods using global and local features [24], graph-based methods [96], contour-based methods and skeleton based methods [85], in addition to methods using keypoints [61, 86, 87].

Global methods extract features using the coarse information of the shape, and hence do not convey much information about the local details. Such methods include shape signatures [26], Fourier descriptors [97], and angular partitioning [65]. Global methods are robust against noise but on the detriment of representing fine details. On the other hand, other methods take into consideration the local region of the shape points, which makes them capable of capturing fine details of the shape. Such methods include curvature scale space (CSS) [86], shape contexts [61], and variations of binary local patterns [59].

Graph-based methods represent features using graph structures in contrast to statistical methods which use statistical natures of appearances [96]. Advantages of graph-based methods are their ability to represent spatial and hierarchical relationships between the object parts [98]. In addition, graph structures permit partial matching. On the other hand, graph matching is time-consuming and thus it is common to transform a graph to a numerical feature vector in order to speed up computations, which often result in information loss [92, 93]

Contours and skeletons have been used as an intermediate representation before feature extraction. Contours are more robust against noise than skeletons, as skeletons tend to generate noisy branches and artifacts in presence of shape border perturbations [85]. On the other hand, skeletons are more suitable in applications that require the segmentation of the original object into constituent parts for subsequent graph-based feature representation [88, 99].

Keypoint-based methods use a number of shape points for feature extraction. A number of descriptors use uniform keypoint sampling from the shape contours without special consideration about the keypoints curvature or location [61, 87, 100, 101]. Other methods extract specific interest points from the shape contour. High curvature points of the contour have been used for keypoint extraction [89–91]. Curvature scale space (CSS) uses scale space filtering [102] to extract contour inflection points [86]. Then, the contour deformation and merging of inflection points caused by scale space filtering are used for feature extraction.

Scale-space filtering has also been used to extract distinctive keypoints in intensity images in the well-known SIFT descriptor [103]. However, it has been shown that SIFT keypoints are suboptimal compared to keypoints that are uniformly sampled from the shape contours when using complex binary images such as historical hieroglyphs [104]. This result is due to the absence of local changes of intensity in binary images that hinders scale-space filtering from detecting distinctive keypoints and attributing them characteristic scales.

In the following, I present a novel keypoint descriptor. An earlier version of this work has been reported in [19].

3.2 Image prominent keypoints

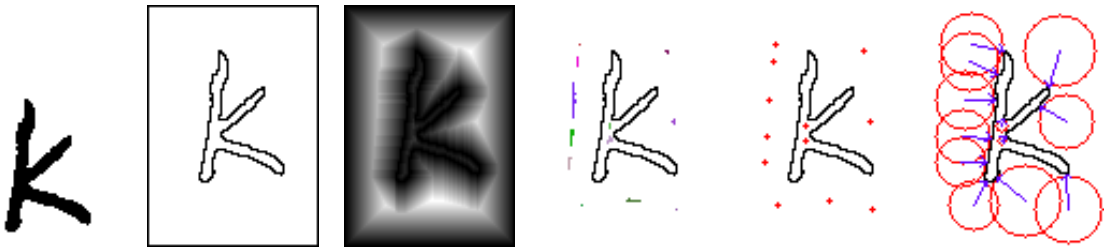
The proposed descriptor operates as follows: First, keypoints are extracted (Sec. 3.2.1). Then, a number of keypoints are selected among the extracted ones and the others are filtered out (Sec. 3.2.2). Finally, a feature vector is calculated for each keypoint (Sec. 3.2.3).

3.2.1 Keypoint extraction

In this step, a transformation is applied on the input binary image in order to generate background information. Then, points having specific characteristics in their local areas are used to extract keypoints.

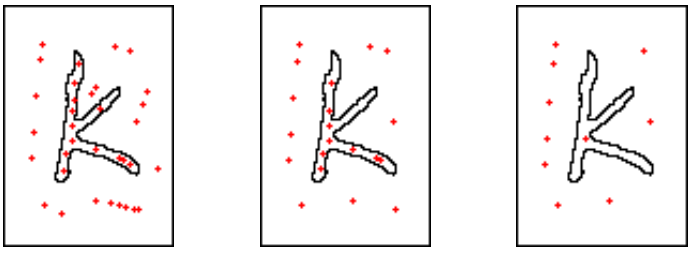
For my image transformation, I use the distance transform (DT) [105]. DT generates a grayscale image where the intensity of each pixel corresponds to its distance from the nearest foreground pixel (Fig. 3.1(c)). Here, the distance between pixels is equal to their Manhattan distance as commonly used in DT implementations [106].

Keypoints are extracted as follows: First, the original image (Fig. 3.1(a)) is normalized by applying contour detection and image translation (Fig. 3.1(b)). Then, background information is generated using DT (Fig. 3.1(c)). Before applying



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

Fig. 3.1: Keypoint extraction steps: (a) Original binary image. (b) $W_F \times H_F$ image after normalization ($a = 0.25$). (c) DT image. (d) Regions of equal maximal intensity highlighted in different colors (e) Keypoints ($k = 11$). (f) Keypoint vectors ($\alpha = 1$): Circle radii correspond to the keypoint distance from the nearest contour point, and arrows show the orientation of the vector delimited by the keypoint and its nearest contour point.



(a) $k = 5$ (b) $k = 9$ (c) $k = 15$

Fig. 3.2: Effect of the parameter k on the number of keypoints.

DT, a 1-pixel-width border frame is added to the normalized image in order to delimit the object so DT does not systematically generate maxima at the borders. Next, regions of equal maximal intensity are detected on the DT image using a $k \times k$ square window (Fig. 3.1(d)). Finally, the detected regions are represented using their centers of masses which are taken as keypoints (Fig. 3.1(e)).

Contour detection is used to produce a compact representation of the original image that reduces the number of foreground pixels but preserves the visual information [85, 107]. Due to using contours, keypoints can be extracted from regions inside and outside the object (Fig. 3.1(e)).

The dimensions (W_F, H_F) of the frame used before applying DT are calculated as follows:

$$W_F = (1 + a)\ W_{BB}, \quad H_F = (1 + a)\ H_{BB} \tag{3.1}$$

where W_{BB} and H_{BB} are the dimensions of the object's bounding box, and $a \geq 0$ is introduced to allow for a space between the object contours and the frame pixels in order to extract keypoints in these regions. The object is translated towards the center of the frame.

Regions of equal maximal intensity are detected using a $k \times k$ square window located at each DT image pixel. The parameter k affects the number of extracted local maxima. The larger k gets, the fewer keypoints are detected (Fig. 3.2).

Due to using DT to generate background information, the extracted keypoints are in locus of symmetry between foreground pixels and thus characterize the object using its local symmetries. I anticipate the significance of such keypoints in shape representation due to the importance of symmetry as a characteristic of patterns that is exploited in human perception [108] and in computational image matching [109].

3.2.2 Keypoint selection

The initial number of keypoints can be reduced by filtering out the redundant and sensitive keypoints. Redundant keypoints duplicate representing the same details of the image, and keypoints that are located very close to contours are sensitive to insignificant changes in image local details.

A measure of keypoint prominence is introduced for keypoint ranking and selection. A prominent keypoint is defined according to two aspects:

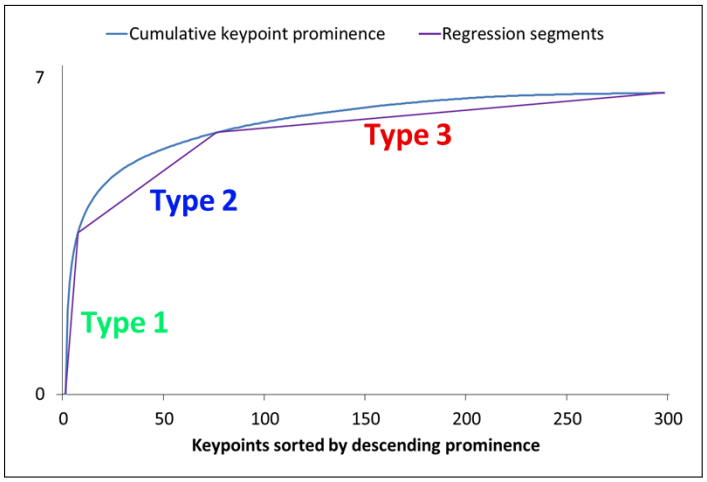
- It has few keypoints in its local area, and thus it is non-redundant.
- It is not located very close to foreground points, and thus it is robust against insignificant changes in image local details.

Formally, the prominence $\gamma(i)$ of a keypoint K_i is calculated as follows:

$$\gamma(i) = \frac{d_i}{1 + n_i} \tag{3.2}$$

where d_i is the distance from keypoint K_i to its closest contour or frame border point, and n_i is the number of close keypoints. A keypoint K_j is considered close to K_i if it is located within a distance to K_i proportional to d_i.

My hypothesis for automatically selecting the most prominent keypoints is as follows: I observe that the range of prominence values commonly indicates three types of keypoints (Fig. 3.3(c)). The first type corresponds to few keypoints with extreme prominence values, the second type corresponds to a larger number of keypoints with increasing redundancy, and the third type corresponds to keypoints with high redundancy and closeness to the contours or frame borders. Since keypoints of the third type are redundant and sensitive, they are filtered out.



(a)

(b)

(c)

Fig. 3.3: Keypoint selection: (a) Curve approximation by three segments applied on image (b), (c) keypoints of the first type in green, keypoints of second type in blue, and keypoints of third type in red. Automatic keypoint selection reduces the number of keypoints from 298 to 78.

In order to filter out keypoints of the third type, I calculate the cumulative keypoint prominence $\Gamma(i)$ for a number i of keypoints ranked in their descending prominence measures, as follows:

$$\Gamma(i) = ln\ (\ \sum_{j=1}^{i} \gamma(j)\ ) \tag{3.3}$$

Fig. 3.3(a) shows a typical curve of Γ as a function of the number of accumulated keypoints. The curve of Γ can be roughly segmented into three parts corresponding to the types of keypoints. In order to find keypoints of each type, a two-dimensional search is used to detect the three segments that minimize the area between them and the curve of Γ. Then, keypoints corresponding to the first and second types are selected. In the literature, a similar strategy has been reported in [110] to automatically detect salient corner points in online sketches using scale-space filtering and digital ink attributes (e.g. pen speed, curvature).

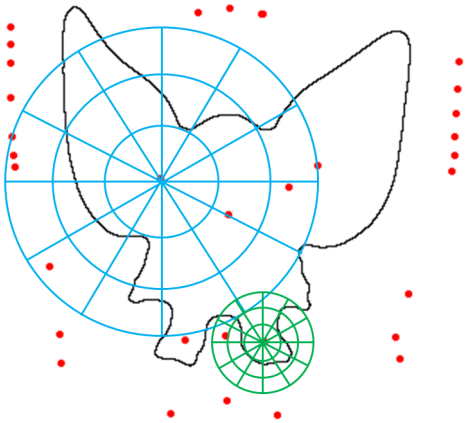


Fig. 3.4: Keypoint feature extraction using a layout which radius is proportional to the distance between the keypoint and its nearest contour or frame point. The feature vector is calculated using the distribution of contour pixels and not frame pixels.

3.2.3 Feature represenation and matching

The last step is to calculate a feature vector to each keypoint K_i. For this purpose, I use a scale-invariant circular layout which radius r_i is proportional to the distance between the keypoint K_i and its closest contour point (Fig. 3.4):

$$r_i = \alpha \times d_i \tag{3.4}$$

where α is a constant. Then, a histogram h_i is extracted by calculating the distribution of contour points in distance and angle bins. The distance between two histograms is expressed by the $\mathcal{X}^2$ statistic:

$$\mathcal{X}^2(h_1, h_2) = \frac{1}{2} \sum_{j=0}^{N_B-1} \frac{[h_1(j) - h_2(j)]^2}{h_1(j) + h_2(j)} \tag{3.5}$$

where N_B is the number of bins in a keypoint histogram. Using the distance d_i to set the radius of the feature layout makes the descriptor scale-invariant.

The dissimilarity d between two images I_1 and I_2 is estimated by the cumulative minimum distance between the images' keypoint histograms:

$$d(I_1, I_2) = \frac{1}{N_1} \sum_{i=0}^{N_1-1} \min_{0 \le j < N_2} \{\mathcal{X}^2(h_i^1, h_j^2)\} \tag{3.6}$$

where N_1 and N_2 are the number of keypoints in images I_1 and I_2. Because $d(I_1, I_2)$ is asymmetric, I express the distance between two images I_1 and I_2 as follows:

$$D(I_1, I_2) = \frac{d(I_1, I_2) + d(I_2, I_1)}{2} \qquad (D \in [0, 1]) \tag{3.7}$$

Table 3.1: Information about the datasets.

Dataset	# images	# classes	# instances
Kimia	216	18	12
MPEG-7	1400	70	20
Zanibbi and Yu	200	20	10
Liang et al.	1086	35	[17, 22]
Tobacco	412	35	[1, 68]

The smaller $D(I_1, I_2)$ is, the more similar I_1 and I_2 are.

The feature vector is translation-invariant due to using the object's bounding box for image normalization, and scale-invariant due to using keypoint-dependent feature extraction layouts. Rotation-invariance can be insured by using the orientation of the vector delimited by the keypoint and its nearest contour point as a reference orientation (Fig. 3.1(f)), or by using shifted matching of the keypoints' feature vectors.

3.3 Evaluation

In this section, I report the evaluation of my descriptor and compare it against other methods. I start first by introducing the datasets used during the experiments.

3.3.1 Datasets

Evaluation is done using five datasets: The Kimia 216 dataset [94] and the MPEG-7 dataset [95] include silhouette images that are neat and which contain single component objects. Zanibbi and Yu's dataset [10] contains handwritten mathematical expressions which exhibit handwriting fluctuations and component displacement, which also appear in Liang et al.'s dataset [96] of hand-drawn diagram sketches. The Tobacco 800 dataset [14] contains logo images that are taken from scanned documents and they are the noisiest compared to the other datasets. Table 3.1 and Fig. 3.5 show information about the datasets and samples of the images.

3.3.2 Descriptor evaluation

Before evaluating the descriptor, I set its parameters as follows: The parameter for setting the normalization frame's dimensions is set $a = 0.25$, which insures a scale-invariant frame with space between its borders and the object contours. A keypoint

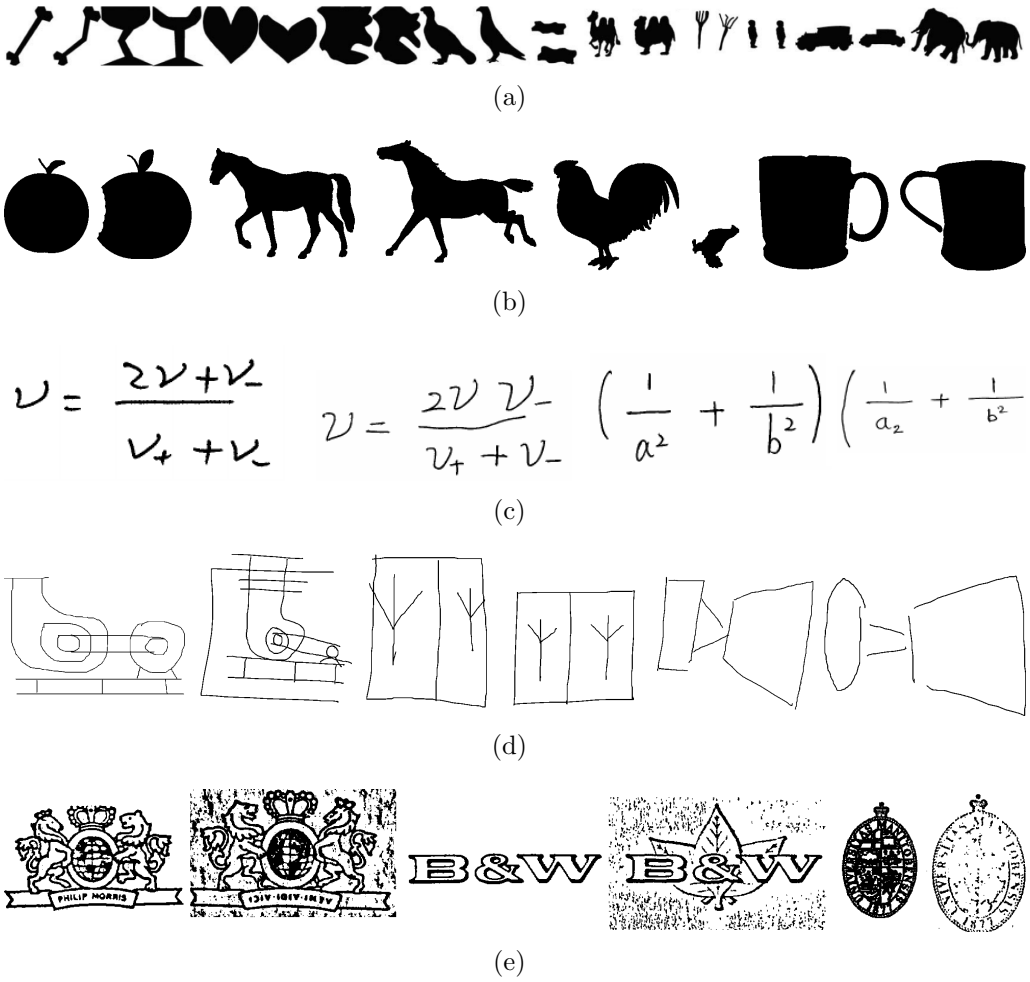


Fig. 3.5: Samples of the dataset images: (a) Kimia's dataset [94], (b) MPEG-7 dataset [95], (c) Zanibbi and Yu's dataset [10], (d) Liang et al. dataset [96], (e) Tobacco 800 logo dataset [14].

K_j is considered close to a keypoint K_i if the distance between them is equal or less than $\frac{d_i}{4}$, where d_i is the distance between keypoint K_i and its closest contour or frame border point. The radial and angular numbers of bins in the keypoint descriptor are set as 4 distance bins and 8 angle bins in order to make a trade-off between distinctiveness and robustness. A small number of bins compromises the descriptor's distinctiveness, while a larger number of bins causes sensitivity to noise and fluctuations [104]. The constant for configuring the keypoint-dependent feature layout radius is set $\alpha = 1.5$ in order to insure taking into account the closest contour points in the smallest distance bins.

Evaluation is done using the *precision at n* metric [111], denoted $P@n$, which is calculated as follows:

$$P@n = \frac{|\{n\ retrieved\ images\} \cap \{relevant\ images\}|}{|\{n\ retrievd\ images\}|} \tag{3.8}$$

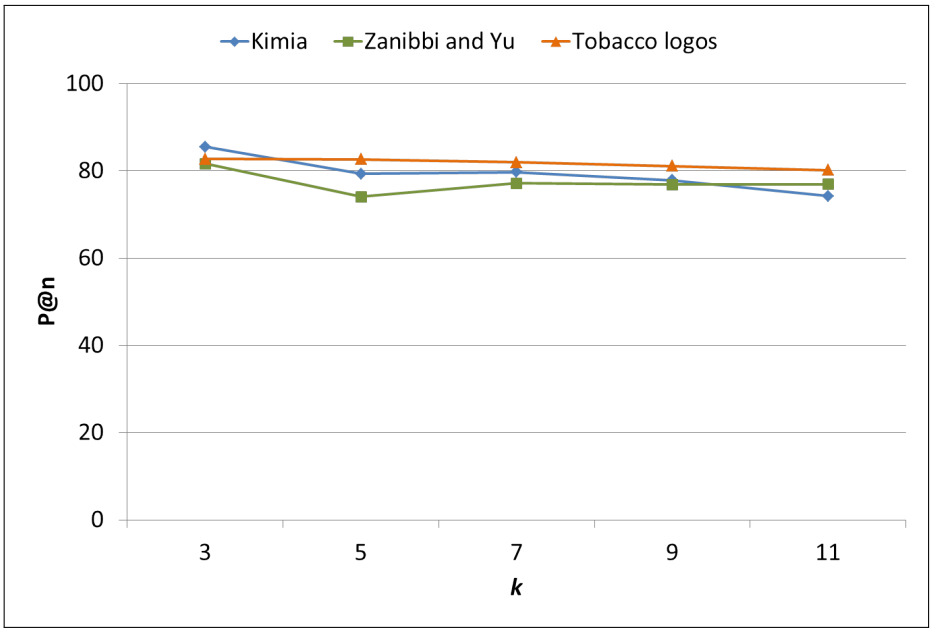


Fig. 3.6: Effect of varying the parameter k on $P@n$.

I set the number of retrieved images n as query-dependent and equivalent to the number of the query's class instances. The larger $P@n$ is, the better matching performances are.

Keypoint sampling evaluation

During keypoint extraction, the parameter k defines the size of the local maxima detection window and thus affects the number of extracted keypoints (Fig. 3.2). I evaluate the effect of this parameter on matching performances.

Fig. 3.6 shows curves of $P@n$ as a function of k using the Kimia 216 dataset, Zanibbi and Yu's dataset, and Tobacco logos dataset. For all datasets, the best matching performances correspond to $k = 3$, which means that the best way is to keep a maximum number of keypoints that will be later filtered during the keypoint selection step. According to the results of this experiment, I set $k = 3$ empirically and use it in subsequent experiments.

Keypoint distinctiveness evaluation

The distinctiveness of BIK's keypoints is assessed by comparison with equidistant sampling which is used in numerous descriptors, namely shape contexts [61]. I perform experiments of image retrieval using the Kimia 216 dataset where each image is used as a query and the average $P@n$ is calculated for all queries. I extract the same number of keypoints using BIK and shape contexts and perform matching

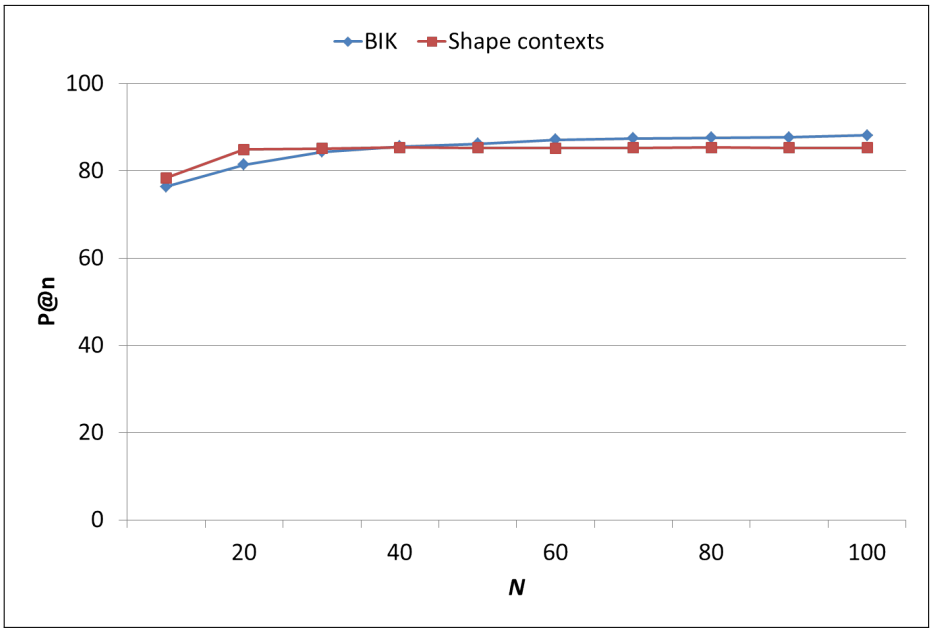


Fig. 3.7: $P@n$ as a function of the number of keypoints N for BIK and shape contexts on the Kimia 216 dataset.

using my keypoint matching steps (Sec. 3.2.3). In order to make the comparison between BIK keypoints and shape contexts fair, I introduced two modifications on the shape contexts: Features are extracted from equidistant keypoints from the contour and all the remaining contour points are considered when calculating the keypoint's histogram, unlike the original shape context descriptor where only the sampled keypoints are considered. In addition, scale-invariance is introduced by making the circular feature extraction layout's size adaptive to the shape by calculating the distance between each keypoint and its farther contour point, instead of using static log-polar layouts. Consequently, these modifications led to better results when compared with the original shape contexts considering only equidistant keypoints and using static log-polar layouts for feature extraction.

Fig. 3.7 shows performances of BIK keypoints and shape contexts. For small numbers of extracted keypoints, using equidistant keypoints outperforms BIK keypoints. Then, starting from 40 keypoints, BIK outperforms shape contexts and the gap increases in correlation with the number of keypoints. In fact, using 40 BIK keypoints outperforms using 100 shape contexts. This result shows that my keypoints are distinctive and outperform the widely-used equidistant keypoint sampling scheme.

Keypoint selection evaluation

The keypoint selection step aims to reduce the number of keypoints by removing the redundant ones and the ones too close to the shape contour. Fig. 3.8 shows

Table 3.2: $P@n$ and number of keypoints N using BIK with all keypoints and with selected keypoints.

Implementation	All keypoints		Selected keypoints	
	$P@n$	N	$P@n$	N
Kimia 216	88.27 %	147	85.49 %	58
Zanibbi and Yu	78.0 %	1610	81.65 %	564
Tobacco logos	77.21 %	1203	82.74 %	379

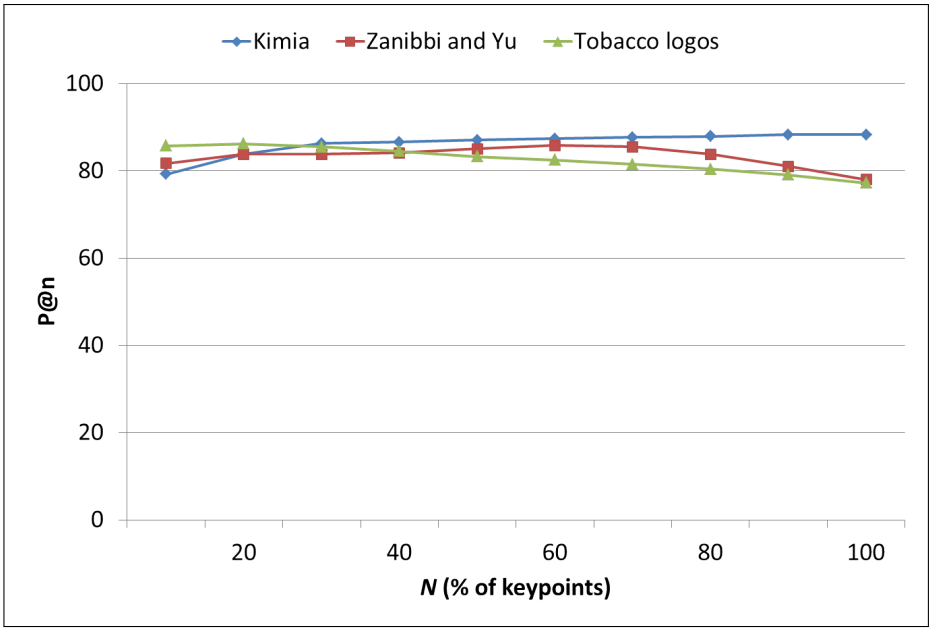


Fig. 3.8: $P@n$ as a function of the percentage of used keypoints relative to the total number of extracted keypoints using BIK.

retrieval performances expressed in $P@n$ as a function of the percentage of keypoints using the Kimia 216 dataset, Zanibbi and Yu's dataset, and Tobacco logos dataset. For the Kimia 216 dataset, performances increase when the percentage of keypoints increases. As for Zanibbi and Yu's and Tobacco logos datasets, optimal performances are obtained when not all of the keypoints are used (when 20% and 60% of keypoints are selected respectively).

Table 3.2 shows retrieval performances of BIK when all keypoints are used and when keypoint selection is performed. For all datasets, the reduction in number of keypoints is significant and roughly makes the third of total keypoints. In case of Zanibbi and Yu's and Tobacco logos datasets, matching performances improve. However, they decrease in case of Kimia 216 dataset. This result suggests that my keypoint prominence-based selection is effective when the initial number of keypoints is relatively large (cases of Zanibbi and Yu's and Tobacco logos datasets). When the initial number of keypoints is relatively small (case of Kimia 216 dataset), the keypoint selection step would better be skipped. This can be done using a threshold on the initial number of keypoints.

Table 3.3: Characteristics of the methods used for comparison.

Method	Keypoint -based	Feature extraction		Feature representation	
		Local features	Global features	Graphs	Feature vectors
BIK	X	X			X
SRD [59]		X	X		X
SC [61]	X	X			X
PSSG [88]	X	X		X	
CS [89]	X	X			X
TSDIZ [90]	X	X			X
SSD+GF [91]	X	X			X
MST [92]	X	X	X	X	
LS+G [93]	X	X		X	X
TPG [96]		X		X	X

3.3.3 Performance comparison with other descriptors

The proposed descriptor is compared with other existing methods using five datasets. For Kimia 216, Liang et al. and MPEG-7 datasets, I compare with state-of-the-art methods and calculate performance measures that are used in available published results. As for Zanibbi and Yu's and Tobacco logos datasets, comparison is done using the $P@n$ performance metric and comparison is done with SRD. Other keypoint-based and graph-based methods are omitted here because they need special considerations regarding keypoint sampling and matching of multi-component images. In case of Kimia's dataset, I calculate the retrieval performance metric reported in several published papers, that is the number of relevant retrieved images for each of the top 6 ranks and the percentage calculated by summing these numbers. In case of Liang et al.'s dataset is done using the *mean average precision (MAP)* metric [111]. In case of the MPEG-7 dataset, the $P@10$ metric is used.

Table 3.3 shows general characteristics of the methods used for comparison. Characteristics are usage of keypoints, feature extraction using local or global features, and feature representation using graphs or feature vectors. Some methods combine local and global features in order to reach higher distinctiveness and robustness against noise. Due to high computations required for graph matching, some methods extract a feature vector from the graph initially produced.

Table 3.4 reports the results of the comparison using five datasets. Overall, BIK yields competitive performances relative to state-of-the-art methods. BIK outperforms shape contexts which use equidistant sampling of contour points, and methods based on contour salient keypoints such as contour salience (CS) [89], min-

Table 3.4: Results of the comparison. For each dataset, a specific metric is used to make the comparison compatible.

Method	Kimia 216 dataset [94]	MPEG-7 dataset [95]	Zanibbi and Yu's dataset [10]	Liang et al. dataset [96]	Tobacco logos dataset [14]
Metric	$P@n$	$P@10$	$P@n$	MAP	$P@n$
BIK	**93.83 %**	**75.48 %**	**82.74 %**	**83.83 %**	**81.65 %**
SRD [59]	86.96 %	.	47.6 %	.	82.55 %
SC [61]	92.12 %	.	.	.	.
PSSG [88]	99.22 %	.	.	.	.
CS [89]	.	36 %	.	.	.
TSDIZ [90]	.	81 %	.	.	.
SSD+GF [91]	.	85 %	.	.	.
MST [92]	.	.	.	29.8 %	.
LS+G [93]	.	.	.	50.9 %	.
TPG [96]	.	.	.	61.6 %	.

imal spanning tree (MST) [92] and Laplacian spectrum with geometry (LS+G) [93]. BIK is competitive compared with methods that combine local and global features such as SRD [59], and methods that use graphs for feature representation including MST [92], LS+G [93], and TPG [96].

3.3.4 Summary and discussion

The proposed descriptor is able to extract distinctive keypoints as demonstrated by comparison with similar numbers of shape context keypoints extracted using equidistant contour points sampling on the Kimia 216 dataset. In fact, BIK is able to outperform shape contexts using significantly fewer keypoints. This is further proven when BIK outperforms methods that detect salient points in the image contour using the other datasets. An interesting direction motivated by these results is to combine BIK keypoints with salient keypoints of the contour for the sake of better distinctiveness.

Experiments on challenging images, such as fluctuated handwritten mathematical expressions of the Zanibbi and Yu's dataset and hand-drawn diagram sketches of Liang et al.'s dataset, demonstrate the reliability of BIK, as it outperforms largely other methods. Methods used for comparison include graph-based descriptors which are known for their high matching performances and ability to perform partial matching.

Reliability of BIK is also shown when assessed on the noisy scanned images of

the Tobacco logos dataset. Results on this dataset are particularly promising given that no preprocessing has been applied for noise reduction.

The keypoint selection based on keypoint prominence is effective in reducing the number of keypoints without significantly compromising the descriptor's distinctiveness. However, the performances improve when the initial number of keypoints is relatively large. For this purpose, a threshold on the initial number of keypoints can be used to activate or skip the prominence-based keypoint selection.

3.4 Conclusion

In this chapter, I introduced a descriptor for binary image matching using image prominent keypoints. The proposed binary image keypoints descriptor (BIK) generates background information in binary images, then extracts keypoints using pixels that have specific characteristics in their local areas. A measure of keypoint prominence is used for automatically selecting the most prominent keypoints and filtering out the redundant and sensitive ones.

The proposed descriptor has been evaluated using five public datasets of silhouette images, handwritten mathematical expressions, hand-drawn diagram sketches, and scanned logo images. Experimental results and comparison with state-of-the-art methods demonstrated that BIK has competitive matching performances when applied on various types of images, including challenging images of fluctuated handwriting and noisy scanned images.

In the following chapter, BIK will be integrated in my document image retrieval system as a measure of candidate relevance. Since BIK is relatively time-consuming due to the large number of its keypoints, it will be used after pre-selecting a number of candidates using an efficient descriptor.

Chapter 4

Query spotting by feature combination

Query spotting in document images is the task aiming to locate a query inside a larger document image. When queries correspond to handwritten words, retrieval is often done by using optical character recognition (OCR) to perform *recognize-then-retrieve* [112]. On the other hand, in case of non-text queries, and when documents are degraded, multilingual or multi-authored, different approaches are called for to insure content-based retrieval [113, 114]. In the latter scenario, features are extracted from the query and used to *spot* its occurrences inside the document image. Usually, segmentation is applied in order to extract relevant regions from the document image [115].

In this chapter, I introduce a method for content-based document image retrieval (CBDIR) that is both segmentation and recognition-free. My method proceeds as follows: First, connected components of the query are paired with their corresponding matches in the document image using shape features. A similarity threshold is then used to select the components of the document image that are most similar to the query components. Next, the selected components are used to recover candidate occurrences of the query in the document image by using size-adaptive bounding boxes. Finally, a score is calculated for each candidate and used for candidate ranking.

To demonstrate its effectiveness, I evaluate my method on CBDIR of handwritten mathematical expression queries. This field has received a growing attention in recent years but retrieval performances are still low [116]. Handwritten mathematical expressions are particularly challenging to tasks of retrieval and recognition due to faulty document image segmentation and expression structural ambiguity [117, 118]. In addition, handwriting fluctuations contribute to making the problem

harder. The difficulty can be illustrated by the results of the latest edition of the International Competition on Recognition of Online Handwritten Mathematical Expressions (CROHME 2013), where the best and second best expression recognition systems among eight participants yielded recognition rates of 60.36% and 23.40% respectively when tested on 671 handwritten expression images [119]. Offline handwritten expressions, which are the focus of my evaluation, are even more challenging since they do not preserve the writing time and stroke order information that usually assist symbol extraction.

The outline of this chapter is as follows: In Sec. 5.1, I review related approaches for recognition-free CBDIR. My theoretical model is presented in Sec. 4.2 and its algorithmic implementation in Sec. 4.3. I evaluate my method on CBDIR of handwritten mathematical expressions and present experimental results in Sec. 4.4. Finally, my concluding remarks and future directions are given in Sec. 5.4.

4.1 Related work

As stated above, methods for CBDIR can be categorized as recognition-based or recognition-free. In this section, I review references of recognition-free methods, as my contribution is of this category. I refer the reader to references [112, 120] for information on recognition-based methods.

Methods for CBDIR using word queries often start by segmenting the document image into lines and words using a priori knowledge about the distance between characters and words. An early method of this type has been introduced by Manmatha et al. [121], as a new alternative to OCR at the time. The authors presented two algorithms for word spotting by estimating the shift between the query and the words in the document image. The document image is subjected to normalization and segmentation into words. Then, the number of words is pruned using the areas and aspect ratios of the words. The two spotting algorithms calculate the shift between the query word and the document image words using the Euclidean distance and Scott and Longuet Higgins' algorithm. Similar methods using word queries have been presented in [122, 123].

Other CBDIR methods incorporate a priori knowledge about the documents and queries' language. For instance, Lu and Tan presented a method for CBDIR of Chinese documents based on a modified Hausdorff distance for Chinese characters [124]. Sari and Kefali presented a method for CBDIR of Arabic documents using specific features of Arabic characters (e.g. diacritics, loops) [125].

Other researchers aimed for language-invariance. In [126], Lee et al. used

the SIFT descriptor [103] for CBDIR word queries. The user's query is introduced online and normalized in a specific font. Then, SIFT is used for feature extraction and matching of the query in the document image. Query occurrences are detected using clustering. The authors evaluated their approach using English and Korean documents and obtained language-invariant results.

Approaches for non-word queries have been introduced. For instance, Zhu et al. presented a CBDIR framework specific to signature queries [127]. Their approach is based on the view that signatures posses a characteristic multiscale structural saliency. In [10], Zanibbi and Yu introduced an approach for CBDIR using mathematical expression queries. Their approach works as follows: First, Recursive X-Y Cutting [128] is used to produce X-Y trees for the document image and the query, and pruning is used to discard irrelevant regions such as text. Then, spotting is done by looking up the query in the document image index using features of its X-Y tree, producing a set of candidates. Candidate ranking is done using Dynamic Time Warping [129].

In a previous work [55], I proposed a voting-based segmentation and recognition-free approach. First, connected components are extracted from the query and document image, and then matched using shape features. Next, connected components of the document image vote for possible locations of the query using component similarity scores and displacement vectors calculated from the query components and their matches in the document image. Voting produces a grayscale image where brighter spots indicate possible occurrence locations, and these spots are used to extract occurrence candidates that are ranked according to their similarity with the query. Similarity is estimated using a shape descriptor [59]. Later, I presented an optimized version using Genetic Algorithms to remove incorrect components in candidate occurrences [130].

The present paper is an improvement of my previous work by making it simpler. I also modify several processing stages that need the query, therefore making offline dataset indexing feasible. In addition, my method does not rely on script or class-specific features unlike the aforementioned methods.

4.2 Theoretical model for feature combination

A CBDIR method relies on a *spotting* stage to find the location of a query inside a larger document image. My method for spotting mimics the intuitive way humans follow to perform the same task when unable to read or identify the query. The human's analogy can be illustrated by the example of a foreign tourist who tries to

find a train station's name on a map that is written in the local language script, to which I suppose the tourist is totally unfamiliar. In case the station's name is composed of several components, the tourist might decompose it and then try to do the spotting component by component. Finally, an occurrence is validated if it contains all the components of the station's name. During this process, the tourist might ignore words or patterns that are obviously irrelevant to the targeted station's name.

My model is based on the assumption that spotting can be considered a Bayesian inference process that uses the local and global similarities of the query and its occurrences in the document image. Here, the local similarities provide *prior knowledge* and lead to calculating $P(A^R)$, that is the probability of a set A of document image components being a relevant occurrence of the query. Then, the suitability of A, including the global similarity, will be evaluated via multiple observed attributes of A, which can be expressed as a vector $\mathbf{x}$. By introducing the likelihood $p(\mathbf{x}|A^R)$, the posterior probability $P(A|\mathbf{x})$ can be evaluated.

4.2.1 Prior knowledge

The query image and the document image contain equations, words, figures, drawing, etc. When considered from a micro level point of view, the image contains *connected components* that can be alphabets, symbols, geometrical primitives, etc. The connected components, or simply components, of the query image I_Q and the document image I_{DOC} are denoted $\{C_i^Q\}_{i=1}^M$ and $\{C_j^{DOC}\}_{j=1}^N$ respectively, where M and N are the number of components in I_Q and I_{DOC}.

Each query component C_i^Q defines a class ω_i. $\{C_j^{DOC}\}_{j=1}^N$ are treated as patterns to be classified into a class among $\{\omega_i\}_{i=1}^M$, corresponding to $\{C_i^Q\}_{i=1}^M$.

A component classifier is used to calculate $P(\omega_i|C_j^{DOC})$, which is the probability that C_j^{DOC} corresponds to the class ω_i. Each component C_j^{DOC} is then assigned the class ω_i having the largest probability. For each document image component C_j^{DOC}, I have $\sum_{i=1}^M P(\omega_i|C_j^{DOC}) = 1$.

After attribution to a class among $\{\omega_i\}_{i=1}^M$, the components $\{C_j^{DOC}\}_{j=1}^N$ are used to form candidates of I_Q occurrences in I_{DOC}. A candidate is a set A of document image components and it is defined as follows:

$$A = \{C_{\phi(i)}^{DOC}\}_{i=1}^M \tag{4.1}$$

where $\phi(i)$ is a function that returns the index j of C_j^{DOC} that is assigned to ω_i. $\phi(i)$ insures that A has a document image component from each class ω_i.

At this stage, A is a relevant candidate if it contains components from all the classes $\{\omega_i\}_{i=1}^{M}$. A^R denotes the event that set A is a relevant candidate. I take as the initial prior probability of A^R as follows:

$$P(A^R) = \prod_{i=1}^{M} P(\omega_i | C_{\phi(i)}^{DOC}) \tag{4.2}$$

Here, I assume that $\{P(\omega_i | C_{\phi(i)}^{DOC})\}_{i=1}^{M}$ are independent.

4.2.2 Observation

Eq. 4.2 does not take into account the locations of components relative to each other inside a candidate A. Therefore, multiple (K) observations $\mathbf{x} = [x_1 \dots x_K]$ concerning the global resemblance and suitability of A are introduced by way of a likelihood function $p(\mathbf{x}|A^R)$.

4.2.3 Inference

The *evidence* provided by $\mathbf{x}$ is used to update the relevance probability using Bayes' theorem:

$$P(A^R|\mathbf{x}) = \frac{p(\mathbf{x}|A^R) \times P(A^R)}{p(\mathbf{x})} \tag{4.3}$$

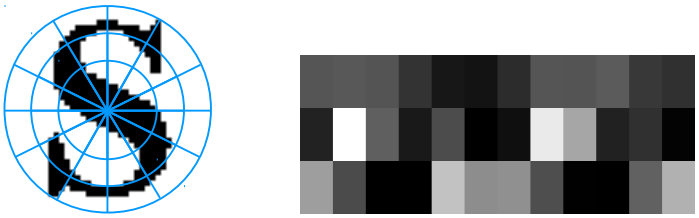
which shows that the posterior probability $P(A^R|\mathbf{x})$ is maximized when the quantity $p(\mathbf{x}|A^R) \times P(A^R)$ is maximized. Without loss of generality, I have $P(A^R|\mathbf{x}) \propto p(\mathbf{x}|A^R) \times P(A^R)$.

4.2.4 Decision function

Using the ln operator, the decision function is expressed as follows:

$$\begin{aligned} D(A) &= ln(P(A^R|\mathbf{x})) \\ &= ln(p(\mathbf{x}|A^R)) + \sum_{i=1}^{M} ln(P(\omega_i | C_{\phi(i)}^{DOC})) \end{aligned} \tag{4.4}$$

Therefore, a candidate A that maximizes $D(A)$ can be judged to be relevant to query I_Q.



(a) (b)

Fig. 4.1: Feature extraction from connected components: (a) Feature extraction layout. (b) Illustration of a feature vector (the brighter the bin region the larger the value).

4.3 Algorithmic implementation

My algorithm proceeds as follows: First, features are extracted from the components of I_Q and I_{DOC} (Sec. 4.3.1) and used to detect candidate occurrences of I_Q in I_{DOC} (Sec. 4.3.2). Next, a score is calculated for each candidate to express its relevance to the query (Sec. 4.3.3).

4.3.1 Component feature extraction and matching

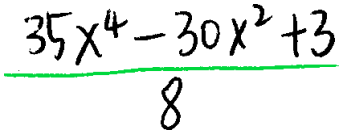
A feature vector $\mathbf{V}$ is produced for each component of I_Q and I_{DOC}. Features are extracted by calculating the distribution of pixels inside a bounding circular layout of which the origin is the component's centroid (Fig. 4.1). The similarity between two components C_i and C_j is equivalent to the histogram intersections between their corresponding vectors, which is calculated as follows:

$$S(C_i, C_j) = \sum_{r=0}^{R-1} \sum_{\theta=0}^{\Theta-1} min(\mathbf{V}^i_{r,\theta}, \mathbf{V}^j_{r,\theta}) \tag{4.5}$$

where R and Θ refer to the radial and angular number of sections. Two components C_i and C_j are considered similar if they satisfy $S(C_i, C_j) \geq \alpha$, where $\alpha \in [0, 1]$ is a similarity threshold. $S(C_i, C_j)$ is the practical implementation of $P(\omega_i \mid C_j^{DOC})$ defined in Sec. 4.2.1.

4.3.2 Detection of query occurrence candidates

One component of the query, that I call *main component* $\hat{Q}$, is determined and used as a seed for candidate occurrence detection. In this implementation, $\hat{Q}$ is chosen as the largest component in terms of number of pixels. Then, components of the document image I_{DOC} which are similar to $\hat{Q}$ are detected. The set of components of I_{DOC} which are similar to $\hat{Q}$ is denoted $B = \{C_j^{DOC} \mid S(\hat{Q}, C_j) \geq \alpha : 1 \leq$



(a)

$$\phi_0(x) = 1 \qquad \phi_1(x) = x$$
$$\phi_2(x) = \frac{3x^2 - 1}{2} \qquad \phi_3(x) = \frac{5x^3 - 3x}{2}$$
$$\phi_4(x) = \frac{35x^4 - 30x^2 + 3}{8} \qquad \phi_5(x) = \frac{63x^5 - 70x^3 + 15x}{8}$$

(b)

Fig. 4.2: Illustration of the bounding box-based spotting procedure: (a) An example of a handwritten query with the *main component* $\hat{Q}$ highlighted in green. (b) Matches of $\hat{Q}$ are highlighted in green. The blue bounding box refers to a relevant candidate, and the two red bounding boxes refer to irrelevant candidates (other red bounding boxes are omitted for clarity).

$j \leq N\}$. The neighboring components of an element of B possibly belong to an occurrence of I_Q in I_{DOC} and they are extracted to form a candidate A. Neighboring components extraction is done using a bounding box that is calculated using the query's dimensions (W_Q, H_Q) and $\hat{Q}$ (Fig. 4.2). The bounding box's dimensions are calculated as follows:

$$(W, H) = (W_Q, H_Q) \times \frac{size\ of\ \hat{G}}{size\ of\ \hat{Q}} \times \beta \tag{4.6}$$

where $\hat{G}$ denotes a match of $\hat{Q}$ in B, *size* of a component is expressed by the number of its pixels, and β is a parameter to control the size of the bounding box which is introduced to account for handwriting fluctuations. The normalization using the components' sizes makes the boxes size-adaptive.

In order to account for component disconnectedness or merging, spotting is done using a number $N_{\hat{Q}}$ of *main components* instead of one. The extracted *main components* are the $N_{\hat{Q}}$ largest components of I_Q.

4.3.3 Candidate score

The last step is to compute a score for each set A that expresses its relevance as a query occurrence candidate. For this purpose, $p(\mathbf{x}|A^R)$ is estimated as a multidimensional observation $\mathbf{x} = [a\ b\ c]$, where:

- $a = S(A, I_Q)$ is the matching score between the image produced by A and the query I_Q using a shape descriptor (Sec. 4.3.1). Specifically here, the feature extraction layout's centroid corresponds to the centroid of $\hat{G}$ instead of the

global centroid of A, and all components' points located inside the circular layout are considered.

- b is equivalent to the maximum value of a when calculated for the *large components* of A. The *large components* of A are the components having their sizes superior to the average component size in A. b is introduced to account for component disconnectedness and merging.

- c is equal to the number of query components that have similar counterparts in the candidate divided by the total number of query components. c is introduced to penalize cases when a single component of the query is matched to several components by mistake.

The scores a, b, and c are normalized and fall in the interval $[0, 1]$. Large values indicate similarity between A and I_Q while small values indicate dissimilarity.

Assuming that the components of $\mathbf{x}$ are independent, the combined probability $p(\mathbf{x}|A^R) = p(a|A^R)\ p(b|A^R)\ p(c|A^R)$ is integrated in Eq. 4.4, which gives:

$$\begin{aligned} D(A) = & ln(P(a|A^R)) + ln(P(b|A^R)) + ln(P(c|A^R)) \\ & + \sum_{i=1}^{M} ln(P(\omega_i|C^{DOC}_{\phi(i)})) \end{aligned} \tag{4.7}$$

Each candidate A is assigned a score that expresses its relevance as a query occurrence candidate. $score(A)$ is calculated as follows:

$$\begin{aligned} score(A, \gamma) = & ln(1 + a) + ln(1 + b) + ln(1 + c) \\ & + \frac{\gamma}{N_{\hat{Q}}} \sum_{u=1}^{N_A} ln(1 + \max_{1 \leq i \leq M} S(C_i, C_u)) \end{aligned} \tag{4.8}$$

where 1 is added to avoid the ln of zero probability values. Finally, the candidates are ranked in their descending *score*.

$score(A, N_{\hat{Q}})$ is a direct implementation of the theoretical model (Eq. 4.4). When component disconnectedness and merging in a candidate A are significant, the quantity of the score $\sum_{u=1}^{N_A} ln(1\ + \max_{1 \leq i \leq M} S(C_i, C_u))$ accumulates incorrect similarity values that eventually increase the score and cause A to be judged as relevant to query I_Q incorrectly. The parameter $\gamma \in [0, N_{\hat{Q}}]$ is thus introduced to mitigate this effect.

Table 4.1: Information about the MathBrush subset [131] and Zanibbi and Yu's dataset [10].

Dataset	# images	# classes	# instances	Expression size
MathBrush subset	739	50	[11, 16]	19.21
Zanibbi and Yu	240	20	10	13.41

4.4 Experimental Results

In this section, I evaluate my algorithm by investigating the effects of its parameters and comparing it with a state-of-the-art method. I start by introducing the datasets used during the experiments.

4.4.1 Datasets

My method is evaluated using the MathBrush dataset [131] and Zanibbi and Yu's dataset [10] which contain offline expressions[a]. From the original MathBrush dataset, I use a subset that consists of 739 images (50 printed and 679 are handwritten) that belong to 50 classes containing between 11 and 16 instances each which are produced by 20 writers. My subset preparation is based on choosing images that have at least one similar instance. Zanibbi and Yu's dataset contains 200 documents images, 40 printed queries, and 200 handwritten queries provided by 10 writers. The document images are collected from the CVPR 2008 conference proceedings, their size is 2560×3310 pixels and resolution is 300dpi. Table 4.1 summarizes information about the datasets and Fig. 4.3 shows histograms of the expression sizes (i.e. average number of components) in each dataset. Fig. 4.4 illustrates the difficulty of the data with examples taken from Zanibbi and Yu's dataset.

The images are subjected to binarization [76] followed by contour detection based on previous results that demonstrate the effectiveness of contours as a compact shape representation (Chapter 2) [20].

4.4.2 Parameter setting

The radial and angular numbers of sections in the shape descriptor (Sec. 4.3.1) have to be set in a way to cope with the data. Small values of R and Θ compromise the descriptor's distinctiveness, while large values cause sensitivity to noise and fluctuations [104]. Based on these considerations, I set their values to be $R = 5$ and $\Theta = 10$.

[a] I use datasets of offline expressions for the sake of generality. For a survey on datasets of online expressions, the reader is referred to [116].

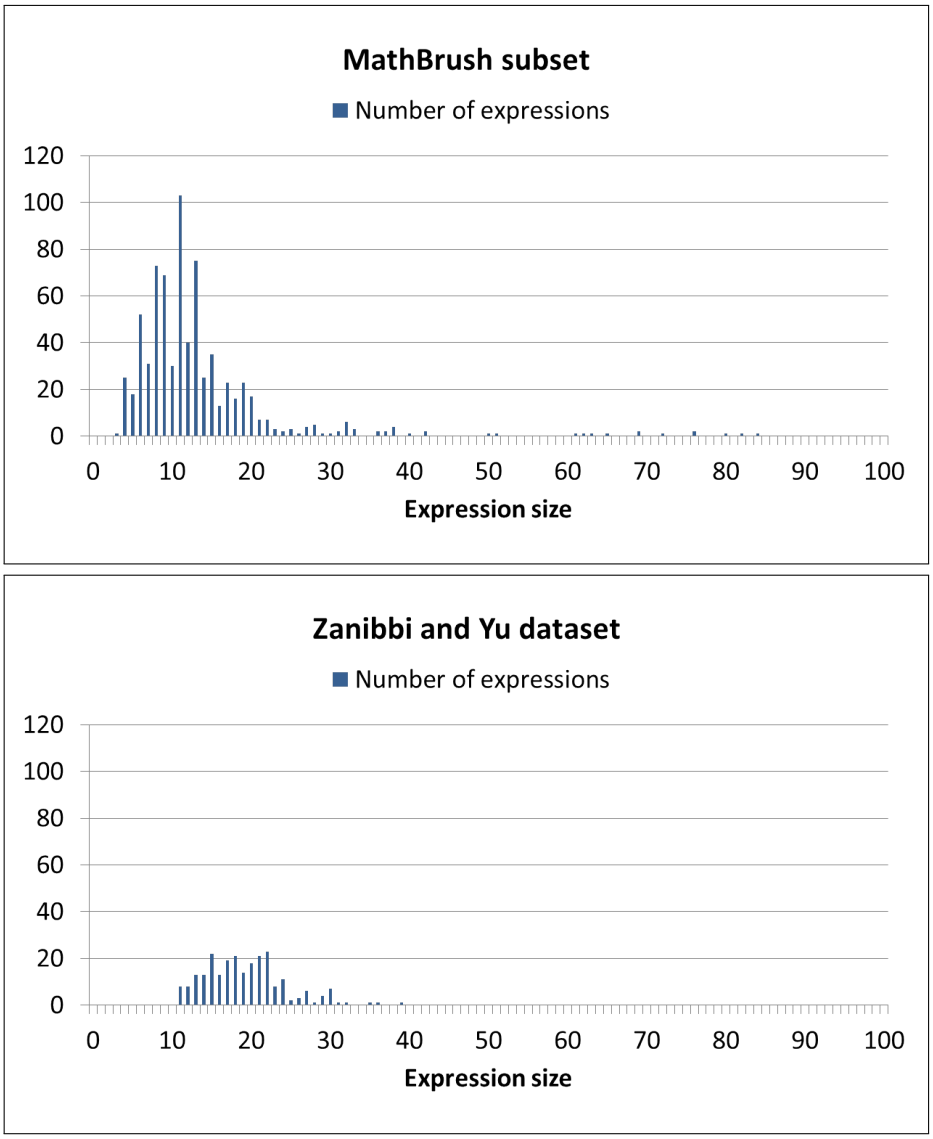


Fig. 4.3: Expression size histograms of the MathBrush subset and Zanibbi and Yu's dataset. The size of an expression is equal to the number of its components.

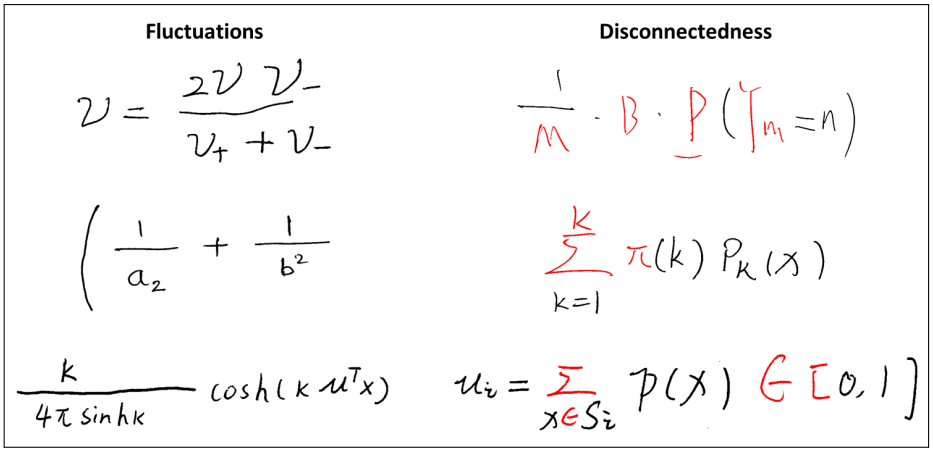


Fig. 4.4: Data challenges. Left: examples of fluctuated queries. Right: examples of component disconnectedness (highlighted in red).

The parameter γ controls the weight of a priori knowledge estimated by the query and candidate component similarities and is used for calculating the candidate scores (Eq. 4.8). γ is introduced to mitigate the frequent component disconnectedness and merging. In order to observe the effect of γ, I conducted image matching experiments on the MathBrush subset and the 240 queries of Zanibbi and Yu's dataset using a symmetrical similarity measure calculated from the candidate score (Sec. 4.3.3):

$$Similarity(I_i, I_j) = \frac{score_{I_i}(I_j, \gamma) + score_{I_j}(I_i, \gamma)}{2} \tag{4.9}$$

where I_i and I_j are dataset images, and $score_{I_i}(I_j, \gamma)$ refers to calculating the score of candidate I_j when taking I_i as the query.

For evaluation, the *precision at n* metric, denoted $P@n$, is used [111]. $P@n$ is defined as follows:

$$P@n = \frac{|\{n\ retrieved\ images\} \cap \{relevant\ images\}|}{|\{n\ retrieved\ images\}|} \times 100 \tag{4.10}$$

where n is equal to the number of instances in the query's class and $|S|$ is the number of objects in the set S. The larger $P@n$ is, the better matching performances are.

Fig. 4.5 shows the effect of varying values of γ on image matching performances. Although the datasets differ in number of images and expression size, best performances for both are obtained when $\gamma = 2$. I adopt this setting for γ in subsequent experiments.

The parameter β controls the size of the bounding box used for spotting (Eq. 4.6). Based on my observation, writers tend to reduce the space between expression symbols and hence deliver handwritten expressions that are more compact compared to printed expressions. I set $\beta = 1.1$ in order to make the bounding box calculated from the writer's query slightly larger so as to account for the compact writing style.

Before retrieval operations on Zanibbi and Yu's dataset, components of the query that have less than 10 pixels plus thick and large components of the document images are filtered out. A component is considered thick if its contour pixels are less than 30% of its total pixels, and large if the total number of pixels exceeds 1000 pixels. This procedure filters out on average 22.85% of each document image foreground pixels corresponding mostly to binarized figures. The number of main components used for spotting is taken as $\frac{3}{4}N$, and a candidate is discarded if its number of components is smaller or larger than N by $\frac{N}{4}$. In order to maintain a reasonable processing time, I apply dataset indexing using connected component clustering (Chapter 5) [21]. During the query's main component matching, a maxi-

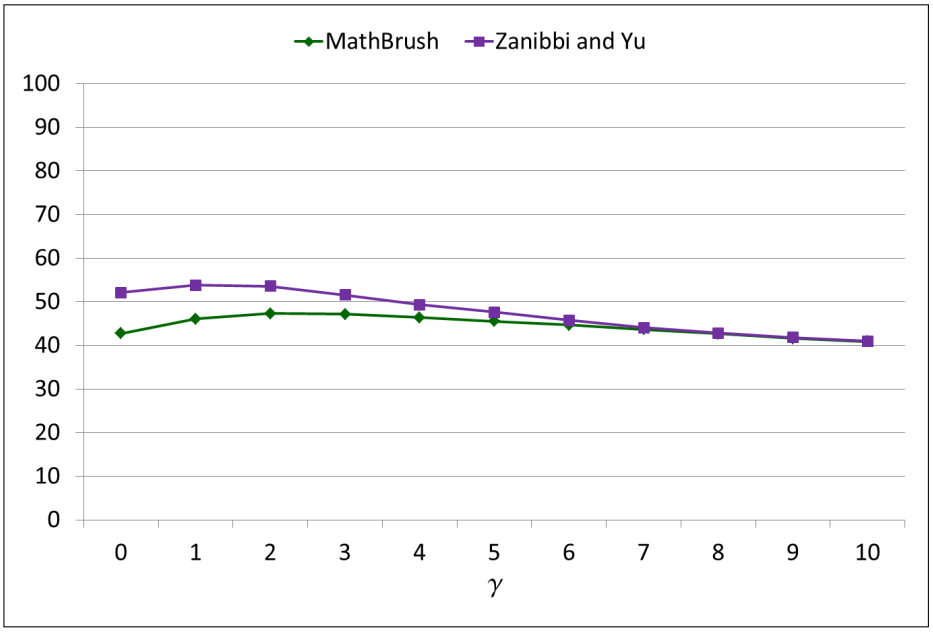


Fig. 4.5: Curves of $P@n$ when γ is varied. The MathBrush subset and Zanibbi and Yu's dataset are used.

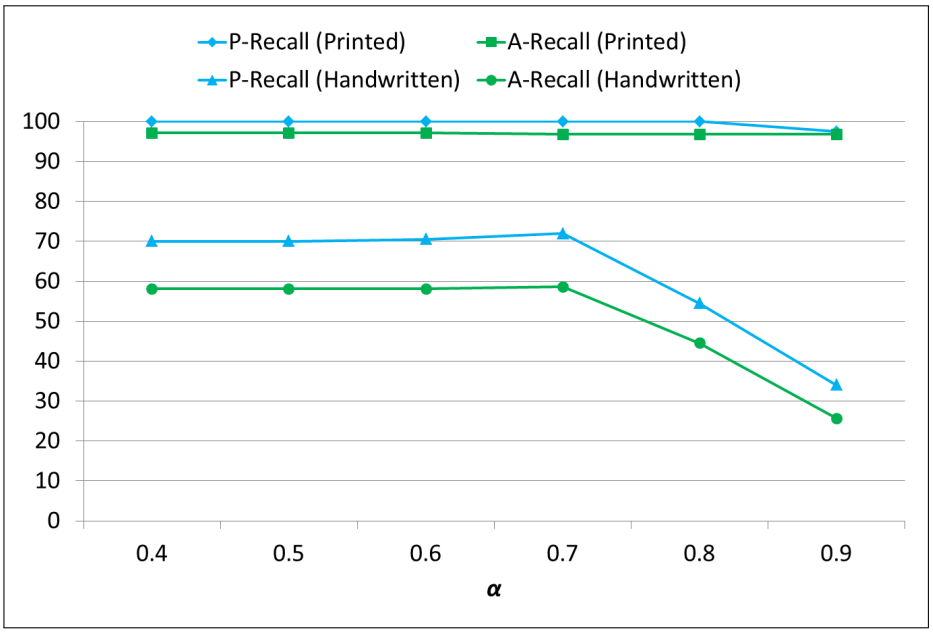


Fig. 4.6: Curves of $(P\text{-}Recall, A\text{-}Recall)$ when α is varied. Zanibbi and Yu's dataset is used.

mum of 50 similar clusters are retrieved.

Subsequent evaluation is done using the *P-Recall* and *A-Recall* metrics that are used in [10]. They are calculated for n retrieved images as follows:

$$P\text{-}Recall = \frac{\#\ relevant\ retrieved\ images}{\#\ relevant\ images\ in\ dataset} \times 100 \tag{4.11}$$

$$A\text{-}Recall = \frac{Candidate\ bound.\ box\ area}{Ground\ truth\ bound.\ box\ area} \times 100 \tag{4.12}$$

where n is a constant that indicates the number of retrieved document images. *P-Recall* indicates the algorithm's ability to retrieve the relevant document images (i.e. the correct *pages*), and *A-Recall* indicates the ability to spot the correct *area* of the relevant query's occurrence in the document image. Since n is fixed, both metrics express the precision of the algorithm.

Now, I evaluate the effect of parameter α which is the component similarity threshold (Sec. 4.3.1). The value of α controls the trade-off between true negative and false positive document image components matched to the query's main components. Fig. 4.6 shows the result of CBDIR experiments for different values of α. Printed and handwritten queries lead to different performance behavior when α is changed. For printed queries, performances remain stable for values of α up to 0.6, while performances of handwritten queries start to decrease when $\alpha > 0.5$. This result is explained by the neat quality of printed fonts in contrary to fluctuations and noise in handwritten queries. According to this result, I set $\alpha = 0.5$.

4.4.3 Comparative evaluation

Table 4.2 shows my comparative results. Using $\gamma = 2$ improves performances as anticipated earlier (Fig. 4.5). Results of my method are presented when the candidate score (Sec. 4.3.3) is used, and when BIK (Chapter 3) is used for candidate re-ranking after initial ranking of 20 candidate groups by the candidate score. The proposed method gives better results when BIK re-ranking is used, and it outperforms Zanibbi and Yu's method [10]. BIK improves significantly retrieval performances especially in case of handwritten queries. For printed queries, results are slightly better when the candidate score is used.

My method outperforms Zanibbi and Yu's method in case of printed and handwritten queries for $n = \{1, 5, 10\}$. This result is due to two fundamental differences between the methods: (1) Zanibbi and Yu's algorithm uses an X-Y cutting-based segmentation step [128] that produces a tree index of the query and the document image. The authors pointed to the brittleness of X-Y cutting when handling handwritten queries especially to variations of the gap between characters. In contrary, my method is segmentation-free, which spares it from erroneous segmentation results. (2) After building the tree indexes, Zanibbi and Yu's method uses a set of features to represent the indexed regions. Among the features, they rely on structural features such as tree depth and the number of nodes. Structural features are vulnerable to noisy patterns such as handwriting [98]. On the other hand, my

Table 4.2: Average values of *P-Recall* and *A-Recall* calculated for $n = 1, 5, 10$ and when $\alpha = 0.7$. Boldface indicates the best results.

Method	n	Printed queries		Handwritten queries	
		P-Recall	*A-Recall*	*P-Recall*	*A-Recall*
My method ($\gamma = M$)	1	85.0%	62.51%	17.0%	6.2%
	5	100%	94.29%	36.5%	15.34%
	10	100%	98.56%	46.5%	20.62%
My method ($\gamma = 2$)	1	100%	94.28%	40.0%	27.83%
	5	100%	96.78%	63.5%	51.15%
	10	100%	96.78%	73.5%	57.92%
My method + BIK ($\gamma = 2$)	1	92.5%	89.29%	54.0%	47.84%
	5	100%	96.29%	70.0%	59.89%
	10	**100%**	**96.78%**	**75.0%**	**62.43%**
Zanibbi and Yu [10]	1	.	90%	38.6%	26.7%
	5	.	90%	54.9%	39.8%
	10	.	90%	63.2%	43.3%

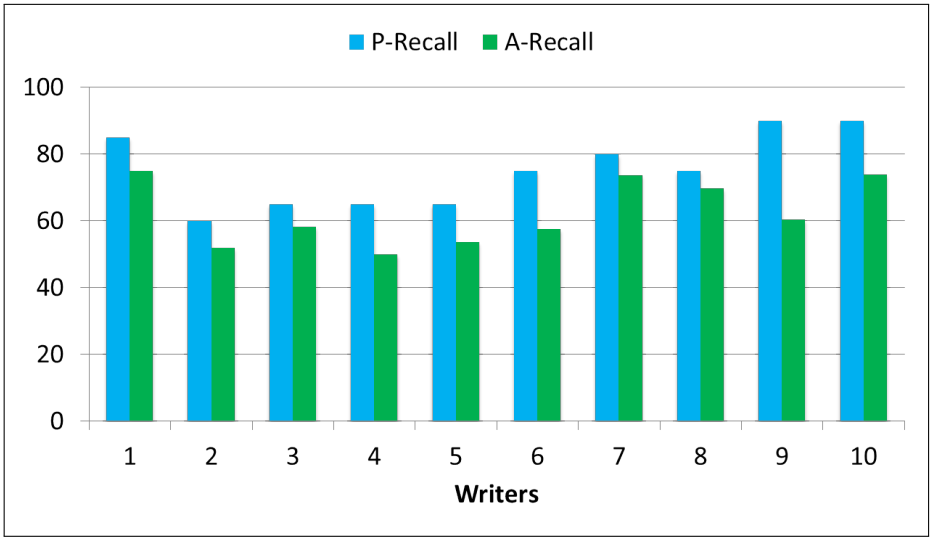


Fig. 4.7: Retrieval performances per writer when $\alpha = 0.5$, $n = 10$ using Zanibbi and Yu's dataset.

method relies essentially on statistical features.

Fig. 4.7 shows the retrieval performances per writer. When the handwriting fluctuations and component displacement are limited, retrieval performances are high (e.g. writer 5 having *P-Recall* = 90% and *A-Recall* = 81.02%). Lower performances by other writers were caused by significant component alteration and displacement. An example is writer 7 who has lower results because of their particular compact writing style, reported similarly by Zanibbi and Yu's [10]. Fig. 4.8 shows examples of queries delivered by a writer that led to satisfactory retrieval performances.

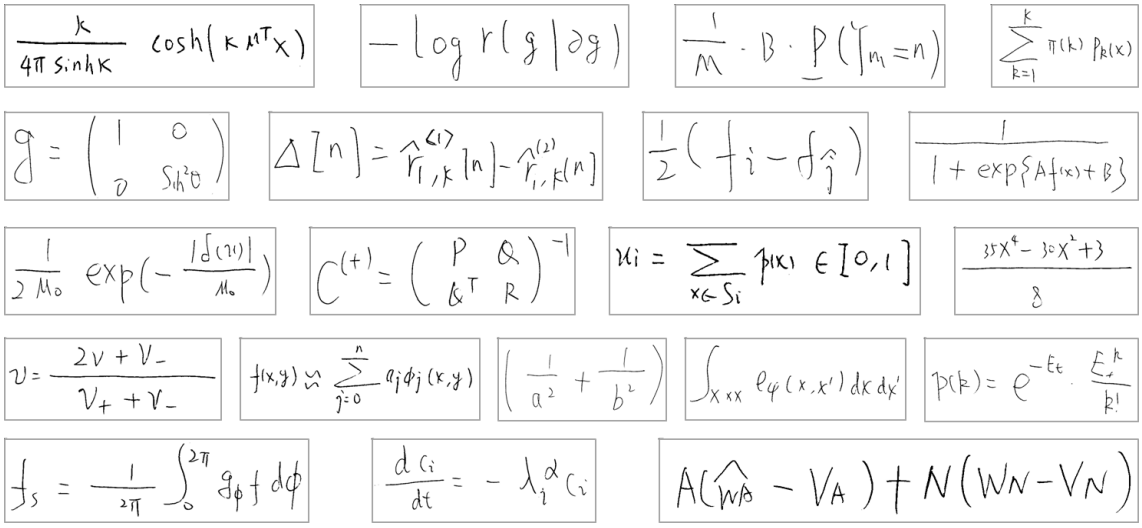


Fig. 4.8: Queries by writer 7 that led to results *P-Recall* = 80% and *A-Recall* = 73.73%.

4.5 Conclusion

In this chapter, I introduced a CBDIR method that is both segmentation and recognition-free. By avoiding segmentation, my method is spared from erroneous segmentation results. Recognition is avoided for the sake of generality and applicability in domains different than text. My method is underpinned by a theoretical model that exploits Bayes' rule and introduces an algorithmic implementation that copes with noises and fluctuations. To prove my method, I evaluate it on CBDIR of mathematical expression queries. Experiments on two datasets and a comparative evaluation show that my method outperforms a state-of-the-art segmentation-based algorithm [10].

Chapter 5

Dataset indexing by clustering

Due to the availability of large storage media, document image datasets are a widespread medium of storing information. Nowadays, such datasets are becoming more and more large scale [132]. In order to insure satisfactory efficiency performances, dataset indexing has been used in retrieval applications [133]. Indexing methods produce a representation of the data that is optimized for online querying. In addition, indexing methods have been used for dataset compression by exploiting data redundancy [134].

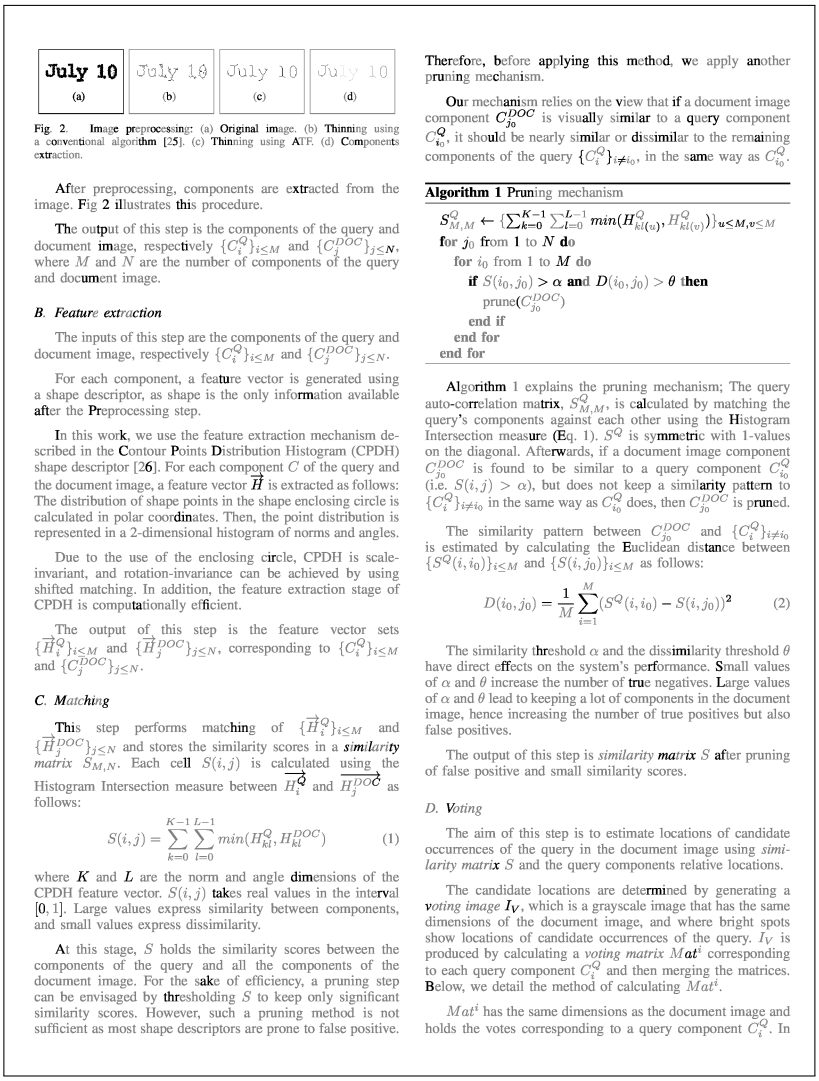
In this chapter, I present a method for document image dataset compression and indexing using redundant information in document images (Fig. 5.1). My method exploits redundancy by performing clustering of similar connected components extracted from document images (Fig. 5.2). Comparing to previous techniques (Sec. 5.1), my method stands out with the following aspects:

- My algorithm is based on similarity estimation between connected components instead of character pattern images (Sec. 5.2.1), which makes it language-independent and more general.

- I introduce an optimized component encoding mechanism that uses some of the components' points and not all of them (Sec. 5.2.2).

- I save the compressed indexing as a text file that is further compressed, which enhances compression performances (Sec. 5.2.3).

I evaluated the proposed algorithm in indexing and compression. Experimental results demonstrate the usefulness of my algorithm as an indexing process for document retrieval (Sec. 5.3.2), and competitive performances comparing with two compression standards, namely the ZIP and XZ formats (Sec. 5.3.1).



July 10 (a) July 10 (b) July 10 (c) July 10 (d)

Fig. 2. Image preprocessing: (a) Original image. (b) Thinning using a conventional algorithm [25]. (c) Thinning using ATF. (d) Components extraction.

After preprocessing, components are extracted from the image. Fig 2 illustrates this procedure.

The output of this step is the components of the query and document image, respectively $\{C_i^Q\}_{i\leq M}$ and $\{C_j^{DOC}\}_{j\leq N}$, where M and N are the number of components of the query and document image.

B. Feature extraction

The inputs of this step are the components of the query and document image, respectively $\{C_i^Q\}_{i\leq M}$ and $\{C_j^{DOC}\}_{j\leq N}$.

For each component, a feature vector is generated using a shape descriptor, as shape is the only information available after the Preprocessing step.

In this work, we use the feature extraction mechanism described in the Contour Points Distribution Histogram (CPDH) shape descriptor [26]. For each component C of the query and the document image, a feature vector $\overrightarrow{H}$ is extracted as follows: The distribution of shape points in the shape enclosing circle is calculated in polar coordinates. Then, the point distribution is represented in a 2-dimensional histogram of norms and angles.

Due to the use of the enclosing circle, CPDH is scale-invariant, and rotation-invariance can be achieved by using shifted matching. In addition, the feature extraction stage of CPDH is computationally efficient.

The output of this step is the feature vector sets $\{\overrightarrow{H}_i^Q\}_{i\leq M}$ and $\{\overrightarrow{H}_j^{DOC}\}_{j\leq N}$, corresponding to $\{C_i^Q\}_{i\leq M}$ and $\{C_j^{DOC}\}_{j\leq N}$.

C. Matching

This step performs matching of $\{\overrightarrow{H}_i^Q\}_{i\leq M}$ and $\{\overrightarrow{H}_j^{DOC}\}_{j\leq N}$ and stores the similarity scores in a *similarity matrix* $S_{M,N}$. Each cell $S(i,j)$ is calculated using the Histogram Intersection measure between $\overrightarrow{H_i^Q}$ and $\overrightarrow{H_j^{DOC}}$ as follows:

$$S(i,j) = \sum_{k=0}^{K-1}\sum_{l=0}^{L-1} min(H_{kl}^Q, H_{kl}^{DOC}) \qquad (1)$$

where K and L are the norm and angle dimensions of the CPDH feature vector. $S(i,j)$ takes real values in the interval $[0,1]$. Large values express similarity between components, and small values express dissimilarity.

At this stage, S holds the similarity scores between the components of the query and all the components of the document image. For the sake of efficiency, a pruning step can be envisaged by thresholding S to keep only significant similarity scores. However, such a pruning method is not sufficient as most shape descriptors are prone to false positive. Therefore, before applying this method, we apply another pruning mechanism.

Our mechanism relies on the view that if a document image component $C_{j_0}^{DOC}$ is visually similar to a query component $C_{i_0}^Q$, it should be nearly similar or dissimilar to the remaining components of the query $\{C_i^Q\}_{i\neq i_0}$, in the same way as $C_{i_0}^Q$.

Algorithm 1 Pruning mechanism

$S_{M,M}^Q \leftarrow \{\sum_{k=0}^{K-1}\sum_{l=0}^{L-1} min(H_{kl(u)}^Q, H_{kl(v)}^Q)\}_{u\leq M, v\leq M}$
for j_0 from 1 to N **do**
  for i_0 from 1 to M **do**
    if $S(i_0,j_0) > \alpha$ **and** $D(i_0,j_0) > \theta$ **then**
      prune($C_{j_0}^{DOC}$)
    end if
  end for
end for

Algorithm 1 explains the pruning mechanism; The query auto-correlation matrix, $S_{M,M}^Q$, is calculated by matching the query's components against each other using the Histogram Intersection measure (Eq. 1). S^Q is symmetric with 1-values on the diagonal. Afterwards, if a document image component $C_{j_0}^{DOC}$ is found to be similar to a query component $C_{i_0}^Q$ (i.e. $S(i,j) > \alpha$), but does not keep a similarity pattern to $\{C_i^Q\}_{i\neq i_0}$ in the same way as $C_{i_0}^Q$ does, then $C_{j_0}^{DOC}$ is pruned.

The similarity pattern between $C_{j_0}^{DOC}$ and $\{C_i^Q\}_{i\neq i_0}$ is estimated by calculating the Euclidean distance between $\{S^Q(i,i_0)\}_{i\leq M}$ and $\{S(i,j_0)\}_{i\leq M}$ as follows:

$$D(i_0,j_0) = \frac{1}{M}\sum_{i=1}^{M}(S^Q(i,i_0) - S(i,j_0))^2 \qquad (2)$$

The similarity threshold α and the dissimilarity threshold θ have direct effects on the system's performance. Small values of α and θ increase the number of true negatives. Large values of α and θ lead to keeping a lot of components in the document image, hence increasing the number of true positives but also false positives.

The output of this step is *similarity matrix* S after pruning of false positive and small similarity scores.

D. Voting

The aim of this step is to estimate locations of candidate occurrences of the query in the document image using *similarity matrix* S and the query components relative locations.

The candidate locations are determined by generating a *voting image* I_V, which is a grayscale image that has the same dimensions of the document image, and where bright spots show locations of candidate occurrences of the query. I_V is produced by calculating a *voting matrix* Mat^i corresponding to each query component C_i^Q and then merging the matrices. Below, we detail the method of calculating Mat^i.

Mat^i has the same dimensions as the document image and holds the votes corresponding to a query component C_i^Q. In

Fig. 5.1: Illustration of component redundancy in a document image (the clusters components are highlighted in black, and the redundant component are gray).

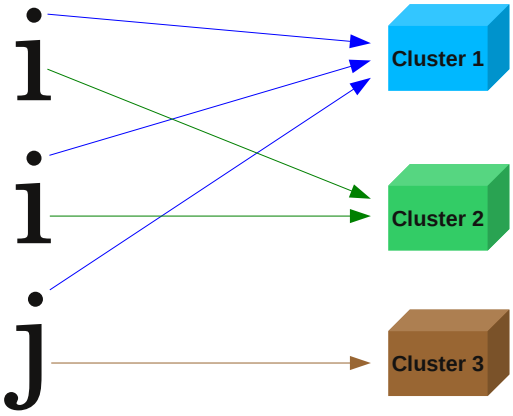


Fig. 5.2: Similarity-based component clustering.

5.1 Related work

Approaches for document image compression using redundant information have been proposed. Haffner et al. presented a method for high resolution color document image compression by separating the image into text, pictures, and background [135][136]. Then, specific compression is applied to each category. For text compression, they use character pattern matching and substitution.

Imura and Tanaka presented a method for document image compression by grouping similar components [137]. First, characters are extracted using image segmentation, and considered as pattern images. Then, a "pseudo-code" is generated for each character image using statistical features and Principal Component Analysis. The groups of characters are used to produce the compressed file. Evaluation showed that this method is language-dependent.

A similar method has been presented by Shiah and Yen [138]. Their technique is specific to Chinese documents and they use a priori knowledge to perform adequate image segmentation, Chinese character merging, and specific features extraction and matching. In both methods [137, 138] the compression error is not evaluated with objective metrics.

Shiah and Yen presented a method for Chinese document image compression [138]. First, image segmentation is done using a priori knowledge about the documents. Then, Chinese characters are extracted using specific techniques of stroke merging. Compression is done using specific feature extraction and matching.

Imura and Tanaka presented a similar method and evaluated it using English and Japanese documents [137]. They obtained language-dependent results. In both methods [137, 138], the compression error is not evaluated with objective metrics.

5.2 Similarity-based connected component clustering

The proposed algorithm takes as input a document image dataset, and produces a compressed file using sequential clustering of connected components and text file compression. The algorithm proceeds as follows: The document image dataset contains M images. For each image I_i, the connected components, $\{C_j\}_{j=1}^{N_i}$, are extracted, where N_i refers to the number of components in I_i. Then, a discrete function $f(C_j)$ returns the cluster index corresponding to C_j if it has been already registered in hash table $Table$, or -1 otherwise. Consequently, C_j is registered in

$Cluster_k$, or a new cluster $Cluster_{k_0}$ is created for C_j. This processing populates $Table$ with clusters of connected components. Then, $Table$ is saved in a text file $TxtFile$. Finally, the output $CompressedFile$ is produced by compressing $TxtFile$ using any text compressing algorithm.

In the following, I explain the mechanism for component similarity estimation (Sec. 5.2.1), component encoding (Sec. 5.2.2), and hash table compression (Sec. 5.2.3).

5.2.1 Component similarity estimation

Similarity between components is estimated using shape features extracted from connected components as done in [55][130]: For a component C_j, a feature vector $\overrightarrow{V_j}$ is extracted by calculating the distribution of pixels in polar coordinate where the origin is the component's centroid. The similarity between two components C_a and C_b is equivalent to the Histogram Intersection between their corresponding vectors, which is calculated as follows:

$$S(C_a, C_b) = \sum_{r=0}^{R-1} \sum_{\theta=0}^{\Theta-1} min(V^a_{r,\theta}, V^b_{r,\theta}) \tag{5.1}$$

where R and Θ refer to the radial and angular number of sections. Two components C_a and C_b are considered similar if they satisfy $S(C_a, C_b) > \delta$, where $\delta \in [0, 1]$ is a similarity threshold.

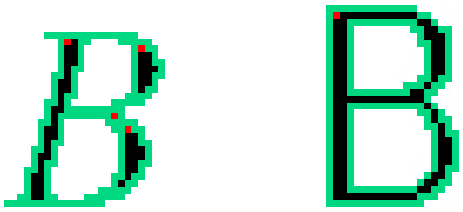
Using this feature extraction and matching mechanism, the function $f(C_j)$ is implemented as follows:

$$f(C_j) = \begin{cases} k, & \textbf{if} \quad \exists C_k[S(C_j, C_k) > \delta] \\ -1, & \textbf{otherwise} \end{cases} \tag{5.2}$$

where C_k refers to the cluster center of $Cluster_k$.

5.2.2 Component encoding

For the sake of optimal compression, the number of points in a component is reduced before saving it in the text file $TxtFile$. The component encoding algorithm extracts the necessary points to reconstruct a component. For a component C_j, the contour points and several non-contour, or *interior points*, are sufficient to reconstruct the component by connected component analysis. Therefore, only those points are needed to be saved. Fig. 5.3 shows examples of original components and



(a) (b)

Fig. 5.3: Reconstruction points in components (contour points are highlighted in green and interior points are highlighted in red): (a) In case of a nearly thin component, the number of encoded points is not significantly reduced. Here, *encoding ratio* = 73%. (b) In case of a thick component, the number of encoded points is significantly reduced. Here, *encoding ratio* = 48%.

their corresponding reconstruction points.

Algorithm 1 shows the component encoding steps: $List_j^R$ refers to the list of points needed for component reconstruction. First, the contour CP and an *interior point* IP are extracted, and added to $List_j^R$. Then, the reconstructed component C_j^R is produced using $List_j^R$. L points $\{P_l\}_{l=1}^{L}$ which exists in C_j but not in C_j^R are detected. Then, one point from $\{P_l\}_{l=1}^{L}$, P_1, is added to $List_j^R$. The iterations of producing C_j^R are repeated until C_j^R and C_j match.

Algorithm 1 Component encoding

define $List_j^R$: List of Points in the jth component
$CP \leftarrow$ ContourPoints(C_j)
$IP \leftarrow$ InteriorPoint(C_j)
$List_j^R \leftarrow CP, IP$
while stop = false **do**
  $C_j^R \leftarrow$ ReconstructComponent($List_j^R$)
  $\{P_l\}_{l=1}^{L} \leftarrow$ DifferencePoints(C_j^R, C_j)
  if $\{P_l\}_{l=1}^{L}$ is empty **then**
    stop = true
  else
    $List_j^R \leftarrow P_1$
  end if
end for

5.2.3 Hash table compression

The hash table, *Table*, is saved in a text file that is used of image reconstruction. In the text file, a header contains information about the images' names and sizes, and the rest of the file contains information about clusters which are the location of the connected component (centroid and image index), *interior points* and contour points, and locations of similar connected components.

Afterwards, the text file is compressed using any available text compression mechanism to produce a compressed indexing of the document image dataset. The idea behind using a text file is to exploit the character redundancy inside a plain text which is a main feature of text compression algorithms. After compressing the text file, the result is a binary file that has a reduced size.

5.3 Experimental results

I evaluate the algorithm's performances in terms of compression and indexing. Throughout the experiments, I set the component descriptor dimensions to $R = 3$ and $\Theta = 12$, and the similarity threshold to $\delta = 0.99$. In the following, I call my method C3 as abbreviation to Connected Components Clustering.

5.3.1 Compression performances

Evaluation procedure

I used three printed binary document image datasets that have been collected as follows:

- Dataset 1: 356 document images taken from the book of abstract of the 2014 World Congress on Computational Intelligence. The images are compressed in PNG-ZIP format, their size is 2479 × 3508 and their resolution is 300 dpi.
- Dataset 2: 159 document images taken from the book "Memoirs of John R. Young Utah Pioneer 1847"[a]. The images are compressed in PNG-ZIP format, their size is 2489 × 3518 and their resolution is 300 dpi.
- Dataset 3: 1320 document images taken from the book "Soothill-Hodous: A Dictionary of Chinese Buddhist Terms"[b]. Images contain English and Chinese

[a]Available at http://www.gutenberg.org/ebooks/46391
[b]Available at http://dev.ddbc.edu.tw/glossaries/

words. The images are compressed in TIFF-Group4 format, their size is 2479 × 3508 and their resolution is 300 dpi.

The evaluation procedure consists of calculating the size of the compressed file and the error rate. The *error rate* ξ quantifies the number of pixel differences between the reconstructed image and its corresponding original image over the dataset, and it is calculated as follows:

$$\xi = \frac{1}{M} \sum_{i=1}^{M} \frac{1}{H \times W} \sum_{x=0}^{H-1} \sum_{y=0}^{W-1} |I_i^R(x,y) - I_i(x,y)| \tag{5.3}$$

where M is the number of images in the dataset, I_i and I_i^R refer to the original and reconstructed images, and H and W are the height and width of I_i.

I compare my method, C3, combined with another standard compression method, namely ZIP or XZ [139], against using the standard compression method directly on the dataset.

Results and discussion

Compression performance

Table 5.1 shows the compression results: For all three datasets, C3 achieved higher compression comparing with using the ZIP or XZ compression directly. The best compression came with combining my method with XZ compression, in which case the compression rates (i.e. the size of the compressed file divided on the size of the original dataset) were respectively 6.4%, 2.2% and 16.6%. As for ZIP and XZ, their performances is explained by the fact that the images are already compressed. Therefore, no further significant compression can be achieved.

The performance of the proposed method is affected by the component redundancy in the document image dataset (Fig. 5.1). This can be seen particularly by the compression rates of Dataset 1 and Dataset 2, being 6.4%, 2.2% respectively. In case of these datasets, the number of redundant components is at the order of 10^3. For Dataset 3, the compression rate is 16.6%, as the number of redundant components is at the order of 10^2. The performances are also due to the optimized component encoding using a reduced number of points. Table 5.2 shows the *encoding ratio* which is equal to the number of encoded points divided by the initial number of points. The *encoding ratio* is affected by the thickness of connected components (Fig. 5.3); The thicker a component is, the less number of points needed for reconstruction comparing with the initial number.

Table 5.1: Compression and information preservation results using three datasets

Dataset	Original size	Compression method	Size after compression	Error rate ξ
Dataset 1	107 MB	ZIP **C3-ZIP** XZ **C3-XZ**	102.7 MB **10.5 MB** 102.5 MB **6.9 MB**	1.5×10^{-6}
Dataset 2	50.2 MB	ZIP **C3-ZIP** XZ **C3-XZ**	34.4 MB **1.7 MB** 34.2 MB **1.1 MB**	0.1×10^{-6}
Dataset 3	44 MB	ZIP **C3-ZIP** XZ **C3-XZ**	37.6 MB **13.3 MB** 26.9 MB **7.3 MB**	0.3×10^{-6}

Table 5.2: Compression results using three datasets

Dataset	# components	# clusters	Encoding ratio
Dataset 1	2 713 162	1 031	94.2 %
Dataset 2	414 854	239	75.7 %
Dataset 3	1 835 719	10 792	52.4 %

Information loss

The proposed compression method is lossy, and that is due to the tolerance of the descriptor used to estimate component similarity (Sec. 5.2.1). In my experiments, the error rate values were very low and I observe that it does not affect the document image readability. The component similarity threshold δ can be used as a parameter that controls the trade-off between the compression rate and the error rate.

5.3.2 Indexing performances

I implemented the proposed algorithm as an indexing mechanism for my ongoing document retrieval project [55, 130]. Then, I conducted retrieval experiments using Zanibbi and Yu's dataset [10]. This dataset contains 200 document images taken from a conference proceedings, and 240 printed and handwritten query images of mathematical expressions.

A core part of my document retrieval algorithm is comparing the connected components of the query against the connected components of the dataset images. In case of non-indexed implementation, all components of the document images

are considered. While in case of an indexed implementation, only the components forming the clusters are considered.

I report the average duration of a component comparison process using a desktop computer equipped with a 3.40 GHz CPU. In case of a non-indexed implementation, the average duration to run a comparison was equal to $3,579$ ms. While in case of indexing, the average duration was equal to 705 ms. The improvement in efficiency is then equal to 507%.

5.4 Conclusion

In this chapter, I presented a method for document image dataset indexing and compression by clustering of connected components. My method extracts connected components from each dataset image and performs sequential clustering to make a hash table that is a compressed indexing the dataset. Then, the hash table is saved in a text file, and the text file is further compressed using any available compression methodology. Component encoding in the text file is done using a reduced number of points which are sufficient for component reconstruction.

Experimental results showed that my algorithm improves efficiency when used for indexing in a content-based document retrieval application, and that the compression performances are competitive. Compression produced very low compression errors that do not compromise the document readability.

Chapter 6

Summary and conclusions

This thesis presents a contribution in the area of content-based document image retrieval (CBDIR) using sketch queries, and enables future research applications. In this chapter, the main findings and contributions are summarized and discussed. Then, future applications are overviewed.

6.1 Summary of contributions

Comparing contours and skeletons

The initial stage of the system is preprocessing and normalization of queries and document images. For this purpose, contours and skeletons have been compared as both compact representations that preserve the visual information of objects. Results of the comparison indicate that contours outperform skeletons and that they are stable in case of moderate image variations. In addition, a noteworthy finding is the improvement of skeletons in the presence of noise, due to increase in the number of skeleton points and the image spatial frequency.

This study highlighted the beneficial nature of investigating these two representations in pattern recognition. It provides a guide to the choice of an image representation according to its performances in certain image classes and variations. In addition, it reveals the unexpected improvement observed in skeletons in presence of noise, which may inspire the design of novel shape descriptors. Furthermore, this study is a valuable feedback to researchers on cognitive science, where contours and skeletons are investigated in human perception.

Shape matching using keypoints

The binary image keypoints (BIK) descriptor has been introduced. BIK is able to extract distinctive keypoints by generating background information in binary images, and using a keypoint selection criteria to rank keypoints and select the most important ones automatically. The descriptor is translation-invariant due to using the object's bounding box for image normalization, and scale-invariant by using keypoint-dependent feature extraction layouts. Rotation-invariance can be insured by using the orientation of the vector delimited by the keypoint and its nearest contour point as a reference orientation, or by using shifted matching. Evaluation using five public datasets indicate that BIK is competitive when applied on various types of images.

Experimental results demonstrated that BIK is competitive compared with state of the art methods. Further improvement of BIK can be done by implementing different approaches for keypoint extraction, selection and feature representation using domain knowledge. It can also be extended to color images by applying edge detection as a preprocessing.

Query spotting in document images

A content-based document image retrieval (CBDIR) method is presented. The method avoids segmentation which spares it from erroneous segmentation results, and avoids recognition for the sake of generality and applicability in domains different than text. The proposed method is underpinned by a theoretical model that exploits Bayes' rule and introduces an algorithmic implementation that copes with noises and fluctuations. Experiments on handwritten mathematical expression queries and comparison with a segmentation-based method [10] showed that the proposed method is competitive.

The proposed theoretical model and its implementation are highly modular. Improvement can focus on enhancing each of the modules, and conducting evaluation in different challenging problem domains.

Document image dataset indexing

A method for document image dataset indexing using connected components clustering has been presented. Experimental results showed its effectiveness in improving efficiency when used for indexing, in addition to its successful applicability in compression.

The proposed method can be improved in several aspects: In the present implementation, centers of clusters are connected components that are extracted using pixel connectivity analysis, and centers similarity is estimated using shape features. In other applications, centers of clusters and centers similarity can be defined according to the image classes (e.g. texture patterns in case of texture images, strokes in case of handwritten signature images, etc.). When image variations such as rotation and scale change are anticipated, the centers descriptor can be tuned or a robust descriptor can be used. Moreover, the centers similarity threshold can be made loose to account for component variations caused by noise.

6.2 Research applications

The proposed system allows for several research directions. For instance, domains specifications can be used to improve performances by designing a query normalization procedure. Taking mathematical expression queries as an example, symbol recognition [140] can be used to convert handwritten queries introduced online to printed queries, allowing much higher performances to be reached. In addition, once the type of queries is fixed, irrelevant patterns can be filtered out from the documents using heuristics (e.g. no need to keep text blocks in case of diagram queries).

This thesis introduces a theoretical model that rationalizes the way humans perform query spotting, and presents an algorithmic implementation according to this model. It would be interesting to further investigate the theoretical model by an experiment of human perception using an eye tracking device. In such an experiment, subjects would be asked to spot a query inside a scene, assuming that both the query and the scene objects are unreadable. Then, the experimental output, expressed by the subjects' gaze trajectories, would be beneficial to further improve the model, in addition to its relevance to researchers on human perception and visual attention.

An interesting extension of my system is application on video databases. Particularly, the Khan Academy video library is a good candidate[a] due to its large size (more than 5000 videos) and increasing popularity (more than 2 million subscribers). Extending my system to this library can be done using one of two approaches:

- Transforming the videos into a document image dataset by using *keyframe extraction*.

[a]Khan Academy channel: https://www.youtube.com/user/khanacademy

- Modifying the dataset indexing method (Chapter 5) to process video inputs.

List of publications

Journal papers

1. Houssem Chatbri, Keisuke Kameyama, and Paul Kwan, "A comparative study using contours and skeletons as shape representations for binary image matching." Pattern Recognition Letters (In Press), 2015.

2. Houssem Chatbri and Keisuke Kameyama, "Using Scale Space Filtering to Make Thinning Algorithms Robust Against Noise in Sketch Images." Pattern Recognition Letters 42 (2014): 1-10.

Peer-Reviewed International Conferences

3. Houssem Chatbri, Keisuke Kameyama, and Paul Kwan, "Towards a segmentation and recognition-free approach for content-based document image retrieval of handwritten queries." In Proc. IAPR Asian Conference on Pattern Recognition (ACPR), To Appear, 2015.

4. Houssem Chatbri, Kenny Davila, Keisuke Kameyama, and Richard Zanibbi, "Shape matching using keypoints extracted from both the foreground and the background of binary images." In Proc. IEEE International Conference on Image Processing Theory, Tools and Applications (IPTA), pp.205-210, 10-13 November 2015.

5. Houssem Chatbri and Keisuke Kameyama, "Document Image Dataset Indexing and Compression Using Connected Components Clustering." In Proc. IAPR International Conference on Machine Vision and Applications (MVA), pp.267-270, 18-22 May 2015.

6. Houssem Chatbri, Paul Kwan, and Keisuke Kameyama, "A Modular Approach for Query Spotting in Document Images and Its Optimization Using Genetic Algorithms." In Proc. IEEE World Congress on Computational Intelligence (WCCI), pp.2085-2092, 6-11 July 2014.

7. Houssem Chatbri, Paul Kwan, and Keisuke Kameyama, "An Application-Independent and Segmentation-Free Approach for Spotting Queries in Document Images." In Proc. IEEE International Conference on Pattern Recognition (ICPR), pp.2891-2896, 24-28 Aug. 2014.

8. Houssem Chatbri, Keisuke Kameyama, and Paul Kwan, "Sketch-Based Image Retrieval By Size-Adaptive and Noise-Robust Feature Description." In Proc. IEEE International Conference on Digital Image Computing: Techniques and Applications (DICTA), pp.1-8, 26-28 Nov. 2013.

9. Houssem Chatbri and Keisuke Kameyama, "Sketch-based image retrieval by shape points description in support regions," In Proc. IEEE International Conference on Systems, Signals and Image Processing (IWSSIP), pp.19-22, 7-9 July 2013.

10. Houssem Chatbri and Keisuke Kameyama, "Towards making thinning algorithms robust against noise in sketch images," In Proc. IEEE International Conference Pattern Recognition (ICPR), pp.3030-3033, 11-15 Nov. 2012.

Other publications

11. Houssem Chatbri and Keisuke Kameyama, "A comparative study of keypoint extraction methods in binary image matching." Tunisia-Japan Symposium on Society, Science and Technology (TJASSST), Accepted, 23-25 Feb. 2016.

12. Houssem Chatbri and Keisuke Kameyama, "Thinning Noisy Sketch Images Using Scale Space Filtering." AEARU Workshop on Computer Science and Web Technology, 2015.

13. Keisuke Kameyama, Houssem Chatbri and Wataru Matsumoto, "Content-Based Image Retrieval Using Adaptive Thinning and Illumination Invariant Color Features." In Proc. Tunisia-Japan Symposium on Society, Science and Technology (TJASSST), 2013.

14. Koya Ando, Houssem Chatbri and Keisuke Kameyama, "Optical Character Recognition robust to the difference in font using Support Region Descriptor." IEICE General Meet, D-11-24, 2015.

15. Houssem Chatbri and Keisuke Kameyama, "An adaptive thinning algorithm for sketch images based on Gaussian Scale Space", IEICE technical report, Image engineering, pp. 33-38, 2011.

In preparation

16. Houssem Chatbri and Keisuke Kameyama, "Shape matching using keypoints extracted from both the foreground and the background of binary images and a measure of keypoint uniqueness." Pattern Recognition Letters (Under Review), 2015.

Bibliography

[1] Jayant Kumar, Peng Ye, and David Doermann. "Structural similarity for document image classification and retrieval". In: *Pattern Recognition Letters* 43 (2014), pp. 119–126.

[2] George Nagy. "Twenty years of document image analysis in PAMI". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.1 (2000), pp. 38–62.

[3] Arnold WM Smeulders et al. "Content-based image retrieval at the end of the early years". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.12 (2000), pp. 1349–1380.

[4] Fred W Mast and Stephen M Kosslyn. "Visual mental images can be ambiguous: Insights from individual differences in spatial transformation abilities". In: *Cognition* 86.1 (2002), pp. 57–70.

[5] Valentina Daelli, Nicola J van Rijsbergen, and Alessandro Treves. "How recent experience affects the perception of ambiguous objects". In: *Brain research* 1322 (2010), pp. 81–91.

[6] Edwin G Boring. "A new ambiguous figure." In: *The American Journal of Psychology* (1930).

[7] T. Kato et al. "A sketch retrieval method for full color image database-query by visual example". In: *IAPR International Conference on Pattern Recognition. Vol. I. Conference A: Computer Vision and Applications*. IEEE. 1992, pp. 530–533.

[8] Mathias Eitz et al. "Sketch-based image retrieval: Benchmark and bag-of-features descriptors". In: *IEEE Transactions on Visualization and Computer Graphics* 17.11 (2011), pp. 1624–1636.

[9] Brendan Klare and Anil K Jain. "Sketch-to-photo matching: a feature-based approach". In: *SPIE Defense, Security, and Sensing*. International Society for Optics and Photonics. 2010, pp. 766702–766702.

[10] Richard Zanibbi and Li Yu. "Math spotting: Retrieving math in technical documents using handwritten query images". In: *International Conference on Document Analysis and Recognition (ICDAR)*. IEEE. 2011, pp. 446–451.

[11] Nicolas Mangano et al. "How Software Designers Interact with Sketches at the Whiteboard". In: *IEEE Transactions on Software Engineering* 41.2 (2015), pp. 135–156.

[12] Haojin Yang et al. "Automatic lecture video indexing using video OCR technology". In: *IEEE International Symposium on Multimedia (ISM)*. IEEE. 2011, pp. 111–116.

[13] Haojin Yang, Christoph Oehlke, and Christoph Meinel. "An automated analysis and indexing framework for lecture video portal". In: *Advances in Web-Based Learning-ICWL*. Springer, 2012, pp. 285–294.

[14] Guangyu Zhu and David Doermann. "Automatic Document Logo Detection". In: *International Conference on Document Analysis and Recognition (ICDAR)*. 2007, pp. 864–868.

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

[16] Rui Hu, Stuart James, and John Collomosse. *Annotated free-hand sketches for video retrieval using object semantics and motion*. Vol. 7131. Lecture Notes in Computer Science. Springer, 2012, pp. 473–484.

[17] Kenny Davila et al. "Accessmath: Indexing and retrieving video segments containing math expressions based on visual similarity". In: *IEEE Western New York Image Processing Workshop (WNYIPW)*. IEEE. 2013, pp. 14–17.

[18] Houssem Chatbri, Keisuke Kameyama, and Paul Kwan. "Towards a segmentation and recognition-free approach for content-based document image retrieval of handwritten queries". In: *Asian Conference on Pattern Recognition (ACPR)*. IAPR. 2015.

[19] Houssem Chatbri et al. "Shape matching using keypoints extracted from both the foreground and the background of binary images". In: *International Conference on Image Processing: Theories and Applications (IPTA), to appear*. IEEE. 2015.

[20] Houssem Chatbri, Keisuke Kameyama, and Paul Kwan. "A comparative study using contours and skeletons as shape representations for binary image matching". In: *Pattern Recognition Letters (in press)* (2015).

[21] Houssem Chatbri and Keisuke Kameyama. "Document Image Dataset Indexing and Compression Using Connected Components Clustering". In: *Machine Vision and its Appliactions (to appear)*. IAPR. 2015.

[22] Milan Sonka, Vaclav Hlavac, and Roger Boyle. *Image processing, analysis, and machine vision*. Cengage Learning, 2014.

[23] Kaleem Siddiqi and Stephen M Pizer. *Medial representations: mathematics, algorithms and applications*. Vol. 37. Springer, 2008.

[24] Dengsheng Zhang and Guojun Lu. "Review of shape representation and description techniques". In: *Pattern recognition* 37.1 (2004), pp. 1–19.

[25] Harry Blum. "Biological shape and visual science (Part I)". In: *Journal of theoretical Biology* 38.2 (1973), pp. 205–287.

[26] E Roy Davies. *Machine vision: theory, algorithms, practicalities*. Elsevier, 2004.

[27] GabriellaSanniti di Baja. "Skeletonization of Digital Objects". English. In: *Progress in Pattern Recognition, Image Analysis and Applications*. Vol. 4225. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2006, pp. 1–13. ISBN: 978-3-540-46556-0.

[28] L. Lam, S.W. Lee, and C.Y. Suen. "Thinning methodologies-a comprehensive survey". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14.9 (1992), pp. 869–885.

[29] Ju Jia Zou, Hung-Hsin Chang, and Hong Yan. "Shape skeletonization by identifying discrete local symmetries". In: *Pattern Recognition* 34.10 (2001), pp. 1895–1905.

[30] Azriel Rosenfeld. "Axial representations of shape". In: *Computer Vision, Graphics, and Image Processing* 33.2 (1986), pp. 156–173.

[31] Raymond W Smith. "Computer processing of line images: A survey". In: *Pattern Recognition* 20.1 (1987), pp. 7–15.

[32] Yukako Yamane et al. "Representation of the spatial relationship among object parts by neurons in macaque inferotemporal cortex". In: *Journal of neurophysiology* 96.6 (2006), pp. 3147–3156.

[33] Irvin Sobel. "Neighborhood coding of binary images for fast contour following and general binary array processing". In: *Computer graphics and image processing* 8.1 (1978), pp. 127–135.

[34] Satoshi Suzuki et al. "Topological structural analysis of digitized binary images by border following". In: *Computer Vision, Graphics, and Image Processing* 30.1 (1985), pp. 32–46.

[35] B-K Jang and Roland T. Chin. "Analysis of thinning algorithms using mathematical morphology". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 12.6 (1990), pp. 541–551.

[36] Carlo Arcelli, Di Baja, and Gabriella Sanniti. "A width-independent fast thinning algorithm". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 4 (1985), pp. 463–474.

[37] Niranjan Mayya and VT Rajan. "An efficient shape representation scheme using Voronoi skeletons". In: *Pattern Recognition Letters* 16.2 (1995), pp. 147–160.

[38] Hongzhi Liu et al. "On the generation and pruning of skeletons using generalized Voronoi diagrams". In: *Pattern Recognition Letters* 33.16 (2012), pp. 2113–2119.

[39] Carlo Arcelli. "Pattern thinning by contour tracing". In: *Computer Graphics and Image Processing* 17.2 (1981), pp. 130–144.

[40] Punam K Saha, Bhabatosh Chanda, and D Dutta Majumder. "A single scan boundary removal thinning algorithm for 2-D binary object". In: *Pattern Recognition Letters* 14.3 (1993), pp. 173–179.

[41] Péter Kardos and Kálmán Palágyi. "Topology-preserving hexagonal thinning". In: *International Journal of Computer Mathematics* 90.8 (2013), pp. 1607–1617.

[42] Gábor Németh and Kálmán Palágyi. "Topology preserving parallel thinning algorithms". In: *International Journal of Imaging Systems and Technology* 21.1 (2011), pp. 37–44.

[43] J. Cai. "Robust Filtering-Based Thinning Algorithm for Pattern Recognition". In: *The Computer Journal* 55.7 (2012), pp. 887–896.

[44] Houssem Chatbri and Keisuke Kameyama. "Using scale space filtering to make thinning algorithms robust against noise in sketch images". In: *Pattern Recognition Letters* 42 (2014), pp. 1–10.

[45] Xiang Bai, Longin Jan Latecki, and Wen-Yu Liu. "Skeleton pruning by contour partitioning with discrete curve evolution". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29.3 (2007), pp. 449–462.

[46] Muhammed Melhi, Stanley S. Ipson, and William Booth. "A novel triangulation procedure for thinning hand-written text". In: *Pattern Recognition Letters* 22.10 (2001), pp. 1059–1071.

[47] André Sobiecki, Andrei Jalba, and Alexandru Telea. "Comparison of curve and surface skeletonization methods for voxel shapes". In: *Pattern Recognition Letters* 47 (2014), pp. 147–156.

[48] Robert L Ogniewicz and Olaf Kübler. "Hierarchic voronoi skeletons". In: *Pattern recognition* 28.3 (1995), pp. 343–359.

[49] Theodosios Pavlidis. "A review of algorithms for shape analysis". In: *Computer graphics and image processing* 7.2 (1978), pp. 243–258.

[50] Sven Loncaric. "A survey of shape analysis techniques". In: *Pattern Recognition* 31.8 (1998), pp. 983–1001.

[51] Shuang Liang and Zhengxing Sun. "Sketch retrieval and relevance feedback with biased SVM classification". In: *Pattern Recognition Letters* 29.12 (2008), pp. 1733–1741.

[52] Lambert Schomaker, Edward de Leau, and Louis Vuurpijl. "Using pen-based outlines for object-based annotation and image-based queries". In: *Visual Information and Information Systems*. Springer. 1999, pp. 585–592.

[53] Remco C Veltkamp and Michiel Hagedoorn. *State of the art in shape matching*. Springer, 2001.

[54] Louisa Lam and Ching Y. Suen. "An evaluation of parallel thinning algorithms for character recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17.9 (1995), pp. 914–919.

[55] Houssem Chatbri, Paul Kwan, and Keisuke Kameyama. "An application-independent and segmentation-free approach for spotting queries in document images". In: *International Conference on Pattern Recognition (ICPR)*. IEEE. 2014, pp. 2891–2896.

[56] Yuliang He et al. "Image enhancement and minutiae matching in fingerprint verification". In: *Pattern Recognition Letters* 24.9 (2003), pp. 1349–1360.

[57] Dakai Jin and Punam K Saha. "A new fuzzy skeletonization algorithm and its applications to medical imaging". In: *Image Analysis and Processing (ICIAP)*. Springer, 2013, pp. 662–671.

[58] Rajesh Kumar, JD Sharma, and Bhabatosh Chanda. "Writer-independent off-line signature verification using surroundedness feature". In: *Pattern Recognition Letters* 33.3 (2012), pp. 301–308.

[59] Houssem Chatbri, Keisuke Kameyama, and Paul Kwan. "Sketch-Based Image Retrieval By Size-Adaptive and Noise-Robust Feature Description". In: *International Conference on Digital Image Computing: Techniques and Applications (DICTA)*. IEEE. 2013, pp. 1–8.

[60] David W Paglieroni and Anil K Jain. "Fast classification of discrete shape contours". In: *Pattern Recognition* 20.6 (1987), pp. 583–598.

[61] Serge Belongie, Jitendra Malik, and Jan Puzicha. "Shape matching and object recognition using shape contexts". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.4 (2002), pp. 509–522.

[62] Farzin Mokhtarian and Riku Suomela. "Robust image corner detection through curvature scale space". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20.12 (1998), pp. 1376–1381.

[63] Xiaoqian Xu et al. "A spine X-ray image retrieval system using partial shape matching". In: *IEEE Transactions on Information Technology in Biomedicine* 12.1 (2008), pp. 100–108.

[64] Wing Ho Leung and Tsuhan Chen. "Trademark retrieval using contour-skeleton stroke classification". In: *International Conference on Multimedia and Expo (ICME)*. Vol. 2. IEEE. 2002, pp. 517–520.

[65] A. Chalechale, G. Naghdy, and A. Mertins. "Sketch-based image matching using angular partitioning". In: *IEEE Transactions on Systems, Man and Cybernetics* 35.1 (2005), pp. 28–41.

[66] Tai Sing Lee et al. "The role of the primary visual cortex in higher level vision". In: *Vision research* 38.15 (1998), pp. 2429–2454.

[67] Yukako Yamane et al. "A neural code for three-dimensional object shape in macaque inferotemporal cortex". In: *Nature neuroscience* 11.11 (2008), pp. 1352–1360.

[68] Chia-Chun Hung, Eric T Carlson, and Charles E Connor. "Medial axis shape coding in macaque inferotemporal cortex". In: *Neuron* 74.6 (2012), pp. 1099–1113.

[69] Yasuhiro Hatori and Ko Sakai. "Early representation of shape by onset synchronization of border-ownership-selective cells in the V1-V2 network". In: *JOSA A* 31.4 (2014), pp. 716–729.

[70] Wei Qiu, Yasuhiro Hatori, and Ko Sakai. "Neural Construction of 3D Medial Axis from the Binocular Fusion of 2D MAs". In: *Neurocomputing* (2014).

[71] F. Zhang et al. "An improved parallel thinning algorithm with two subiterations". In: *Optoelectronics Letters* 4.1 (2008), pp. 69–71.

[72] David Navon. "Forest before trees: The precedence of global features in visual perception". In: *Cognitive psychology* 9.3 (1977), pp. 353–383.

[73] J. M. Geusebroek, G. J. Burghouts, and A. W. M. Smeulders. "The Amsterdam Library of Object Images". In: *International Journal of Computer Vision* 61.1 (2005), pp. 103–112.

[74] Houssem Chatbri and Keisuke Kameyama. "Towards making thinning algorithms robust against noise in sketch images". In: *International Conference on Pattern Recognition (ICPR)*. IEEE. 2012, pp. 3030–3033.

[75] Mark E Hoffman and Edward K Wong. "Scale-space approach to image thinning using the most prominent ridge line in the image pyramid data structure". In: *Photonics West'98 Electronic Imaging*. International Society for Optics and Photonics. 1998, pp. 242–252.

[76] Nobuyuki Otsu. "A threshold selection method from gray-level histograms". In: *Automatica* 11.285-296 (1975), pp. 23–27.

[77] John C Russ and Roger P Woods. "The image processing handbook". In: *Journal of Computer Assisted Tomography* 19.6 (1995), pp. 979–981.

[78] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer vision with the OpenCV library.* " O'Reilly Media, Inc.", 2008.

[79] Markus Maier, Matthias Hein, and Ulrike von Luxburg. "Optimal construction of k-nearest-neighbor graphs for identifying noisy clusters". In: *Theoretical Computer Science* 410.19 (2009), pp. 1749–1764.

[80] Sameer K Antani et al. "Partial Shape Matching for CBIR of spine X-ray images". In: *Electronic Imaging 2004*. International Society for Optics and Photonics. 2003, pp. 1–8.

[81] Sameer Antani et al. "Evaluation of shape similarity measurement methods for spine X-ray images". In: *Journal of Visual Communication and Image Representation* 15.3 (2004), pp. 285–302.

[82] Sameer Antani, L Rodney Long, and George R Thoma. "A Biomedical Information System for Combined Content-Based Retrieval of Spine X-Ray Images, Associated Text Information." In: *ICVGIP*. 2002.

[83] Michael Breuß et al. *Innovations for shape analysis: models and algorithms*. Springer Science & Business Media, 2013.

[84] Mingqiang Yang, Kidiyo Kpalma, and Joseph Ronsin. "A survey of shape feature extraction techniques". In: *Pattern recognition* (2008), pp. 43–90.

[85] Houssem Chatbri, Keisuke Kameyama, and Paul Kwan. "A comparative study using contours and skeletons as shape representations for binary image matching". In: *Pattern Recognition Letters* (2015).

[86] Farzin Mokhtarian, Sadegh Abbasi, and Josef Kittler. "Robust and Efficient Shape Indexing through Curvature Scale Space". In: *British Machine and Vision Conference (BMVC)*. Vol. 96. 1996.

[87] Edgar Roman-Rangel and Stephane Marchand-Maillet. "HOOSC128: A More Robust Local Shape Descriptor". In: *Pattern Recognition*. Vol. 8495. Lecture Notes in Computer Science. Springer, 2014, pp. 172–181.

[88] Xiang Bai and Longin Jan Latecki. "Path similarity skeleton graph matching". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30.7 (2008), pp. 1282–1292.

[89] R da S Torres and Alexandre X Falcao. "Contour salience descriptors for effective image retrieval and analysis". In: *Image and Vision Computing* 25.1 (2007), pp. 3–13.

[90] Fernanda A Andaló et al. "Shape feature extraction and description based on tensor scale". In: *Pattern Recognition* 43.1 (2010), pp. 26–36.

[91] Glauco V Pedrosa, Marcos A Batista, and Celia AZ Barcelos. "Image feature descriptor based on shape salience points". In: *Neurocomputing* (2013).

[92] Shuang Liang, Rong-Hua Li, and George Baciu. "A graph modeling and matching method for sketch-based garment panel design". In: *International Conference on Cognitive Informatics & Cognitive Computing (ICCI CC)*. IEEE. 2011, pp. 340–347.

[93] M Fatih Demirci, Reinier H van Leuken, and Remco C Veltkamp. "Indexing through laplacian spectra". In: *Computer Vision and Image Understanding* 110.3 (2008), pp. 312–325.

[94] Thomas Sebastian, Philip Klein, and Benjamin Kimia. "Recognition of shapes by editing shock graphs". In: *International Conference on Computer Vision*. Vol. 1. IEEE Computer Society. 2001, pp. 755–755.

[95] Miroslaw Bober. "MPEG-7 visual shape descriptors". In: *IEEE Transactions on circuits and systems for video technology* 11.6 (2001), pp. 716–719.

[96] Shunlin Liang et al. "Sketch Matching on Topology Product Graph". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.8 (2015), pp. 1723–1729.

[97] D. Zhang and G. Lu. "A comparative study of Fourier descriptors for shape representation and retrieval". In: *Asian Conferernce on Computer Vision (ACCV)*. 2002, pp. 646–651.

[98] H. Bunke and K. Riesen. "Towards the unification of structural and statistical pattern recognition". In: *Pattern Recognition Letters* 33.7 (2012), pp. 811–825.

[99] Punam K Saha, Gunilla Borgefors, and Gabriella Sanniti di Baja. "A survey on skeletonization algorithms and their applications". In: *Pattern Recognition Letters* (2015).

[100] M. Donoser, H. Riemenschneider, and H. Bischof. "Efficient partial shape matching of outer contours". In: (2010), pp. 281–292.

[101] Xin Shu and Xiao-Jun Wu. "A novel contour descriptor for 2D shape matching and its application to image retrieval". In: *Image and vision Computing* 29.4 (2011), pp. 286–294.

[102] Andrew P Witkin. "Scale-space filtering: A new approach to multi-scale description". In: *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Vol. 9. IEEE. 1984, pp. 150–153.

[103] David G Lowe. "Distinctive image features from scale-invariant keypoints". In: *International journal of computer vision* (2004).

[104] Edgar Roman-Rangel and Stephane Marchand-Maillet. "Shape-based detection of Maya hieroglyphs using weighted bag representations". In: *Pattern Recognition* 48.4 (2015), pp. 1161–1173.

[105] Azriel Rosenfeld and John L Pfaltz. "Sequential operations in digital picture processing". In: *Journal of the ACM (JACM)* 13.4 (1966), pp. 471–494.

[106] Arnold Meijster, Jos BTM Roerdink, and Wim H Hesselink. "A general algorithm for computing distance transforms in linear time". In: *Mathematical Morphology and its applications to image and signal processing*. Springer, 2000, pp. 331–340.

[107] Xingwei Yang, Hairong Liu, and Longin Jan Latecki. "Contour-based object detection as dominant set computation". In: *Pattern Recognition* 45.5 (2012), pp. 1927–1936.

[108] Christopher W Tyler. *Human symmetry perception and its computational analysis*. Psychology Press, 2002.

[109] Seungkyu Lee. "Symmetry-driven shape description for image retrieval". In: *Image and Vision Computing* 31.4 (2013), pp. 357–363.

[110] Tevfik Metin Sezgin and Randall Davis. "Scale-space based feature point detection for digital ink". In: *ACM SIGGRAPH 2007 courses*. ACM. 2007, p. 36.

[111] Ricardo Baeza-Yates, Berthier Ribeiro-Neto, et al. *Modern information retrieval*. Vol. 463. ACM press New York, 1999.

[112] Andrew Kae and Erik Learned-Miller. "Learning on the fly: font-free approaches to difficult OCR problems". In: *International Conference on Document Analysis and Recognition (ICDAR)*. IEEE. 2009, pp. 571–575.

[113] Muhammad Ismail Shah and Ching Y Suen. "Word Spotting Techniques in Document Analysis and Retrieval-A Comprehensive Survey". In: *Handbook of Pattern Recognition and Computer Vision* 4 (2010), pp. 353–376.

[114] Jorge Moraleda. "Large scalability in document image matching using text retrieval". In: *Pattern Recognition Letters* 33.7 (2012), pp. 863–871.

[115] Raid Saabni, Abedelkadir Asi, and Jihad El-Sana. "Text line extraction for historical document images". In: *Pattern Recognition Letters* 35 (2014), pp. 23–33.

[116] Richard Zanibbi and Dorothea Blostein. "Recognition and retrieval of mathematical expressions". In: *International Journal on Document Analysis and Recognition (IJDAR)* 15.4 (2012), pp. 331–357.

[117] Ahmad-Montaser Awal, Harold Mouchère, and Christian Viard-Gaudin. "A global learning approach for an online handwritten mathematical expression recognition system". In: *Pattern Recognition Letters* 35 (2014), pp. 68–77.

[118] Nina ST Hirata and Frank D Julca-Aguilar. "Matching based ground-truth annotation for online handwritten mathematical expressions". In: *Pattern Recognition* 48.3 (2015), pp. 837–848.

[119] Harold Mouchere et al. "ICDAR 2013 CROHME: Third international competition on recognition of online handwritten mathematical expressions". In: *International Conference on Document Analysis and Recognition (ICDAR)*. IEEE. 2013, pp. 1428–1432.

[120] Katsumi Marukawa et al. "Document retrieval tolerating character recognition errors–evaluation and application". In: *Pattern Recognition* 30.8 (1997), pp. 1361–1371.

[121] R Manmatha, Chengfeng Han, and Edward M Riseman. "Word spotting: A new approach to indexing handwriting". In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. 1996, pp. 631–637.

[122] Tony M Rath and Rudrapatna Manmatha. "Word spotting for historical documents". In: *International Journal of Document Analysis and Recognition (IJDAR)* 9.2-4 (2007), pp. 139–152.

[123] Shijian Lu and Chew Lim Tan. "Retrieval of machine-printed Latin documents through Word Shape Coding". In: *Pattern Recognition* 41.5 (2008), pp. 1799–1809.

[124] Yue Lu and Chew Lim Tan. "Word spotting in Chinese document images without layout analysis". In: *International Conference on Pattern Recognition (ICPR)*. Vol. 3. IEEE. 2002, pp. 57–60.

[125] Toufik Sari, Abderrahmane Kefali, et al. "A search engine for Arabic documents". In: *Dixième Colloque International Francophone sur l'Ecrit et le Document*. 2008, pp. 97–102.

[126] Duk-Ryong Lee, Wonju Hong, and Il-Seok Oh. "Segmentation-free word spotting using SIFT". In: *IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*. IEEE. 2012, pp. 65–68.

[127] Guangyu Zhu et al. "Signature detection and matching for document image retrieval". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31.11 (2009), pp. 2015–2031.

[128] Haralick Robert M Ha Jaekyu and Ihsin T Phillips. "Recursive XY cut using bounding boxes of connected components". In: *International Conference on Document Analysis and Recognition (ICDAR)*. Vol. 2. 1995, pp. 952–955.

[129] Meinard Müller. "Dynamic time warping". In: *Information retrieval for music and motion* (2007), pp. 69–84.

[130] Houssem Chatbri, Paul Kwan, and Keisuke Kameyama. "A modular approach for query spotting in document images and its optimization using genetic algorithms". In: *IEEE Congress on Evolutionary Computation (CEC)*. 2014, pp. 2085–2092.

[131] George Labahn et al. "Mathbrush: A system for doing math on pen-based devices". In: *IAPR International Workshop on Document Analysis Systems (DAS)*. IEEE. 2008, pp. 599–606.

[132] Simone Marinai, Beatrice Miotti, and Giovanni Soda. "Digital libraries and document image retrieval techniques: A survey". In: *Learning Structure and Schemas from Documents*. Springer, 2011, pp. 181–204.

[133] David Doermann. "The indexing and retrieval of document images: A survey". In: *Computer Vision and Image Understanding* 70.3 (1998), pp. 287–298.

[134] Dmitriy Karpman et al. "Lidar depth image compression using clustering, re-indexing, and JPEG2000". In: *SPIE Defense, Security, and Sensing*. International Society for Optics and Photonics. 2011, 80370G–80370G.

[135] Patrick Haffner et al. “High quality document image compression with DjVu”. In: *Journal of Electronic Imaging* 7.3 (1998), pp. 410–425.

[136] Patrick Haffner et al. “DjVu: Analyzing and compressing scanned documents for Internet distribution”. In: *IEEE ICDAR*. 1999, pp. 625–628.

[137] Hajime Imura and Yuzuru Tanaka. “Compression and string matching method for printed document images”. In: *IEEE ICDAR*. 2009, pp. 291–295.

[138] Chwan-Yi Shiah and Yun-Sheng Yen. “Compression of Chinese Document Images by Complex Shape Matching”. In: *The Computer Journal* 56.11 (2013), pp. 1292–1304.

[139] David Salomon. *Data compression: the complete reference*. Springer, 2004.

[140] Kenny Davila, Stephanie Ludi, and Richard Zanibbi. “Using off-line features and synthetic data for on-line handwritten math symbol recognition”. In: *International Conference on Frontiers in Handwriting Recognition (ICFHR)*. IEEE. 2014, pp. 323–328.