Content-Based Document Image Retrieval using Sketch Queries

著者	Houssem Chatbri
内容記述	この博士論文は内容の要約のみの公開（または一部非公開）になっています
year	2016
その他のタイトル	スケッチ問い合わせを用いた文書画像内容検索
学位授与大学	筑波大学 (University of Tsukuba)
学位授与年度	2015
報告番号	12102甲第7704号
URL	http://hdl.handle.net/2241/00144197

博士(工学)論文概要

Content-Based Document Image Retrieval using Sketch Queries
(スケッチ問い合わせを用いた文書画像内容検索)

システム情報工学研究科 コンピュータサイエンス専攻

氏名　Houssem Chatbri

2016年　3月

1 Introduction

In recent years, more and more universities started to record their lectures and make them available in digital format. Such documents include lecture handouts and videos, and provide a growing amount of data that require specific systems for storage and retrieval. Databases of digitized lecture materials are often queried using text keywords (such as http://videolectures.net/ and https://www.youtube.com/edu). Text-based querying is convenient when users already know that they are looking for and when they can express it in words. However, this is not guaranteed in case of class beginners and when the query cannot be expressed in words (e.g. drawings, mathematical expressions). In this case, Sketch-based retrieval is a more adequate search paradigm.

In this thesis, a system for digitized visual lecture material retrieval using sketch queries is presented. The system allows users to submit sketch queries in order to retrieve digital documents that are indexed offline. The proposed system is modular and operates as follows (Fig. 1): 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.

As an initial application without loss of generality, the system is applied in content-based document image retrieval (CBDIR) of mathematical expression queries. Here, sketch images are binary handwritten images introduced by users using a sketching device, and document images are binarized since the color information is unneeded. Evaluation using a dataset of handwritten mathematical expressions and printed document images and a comparison with a state-of-the-art method demonstrate promising performances and possibility for further improvement and extension.

Contributions of this thesis include the study of contours and skeletons as compact image representations (Sec. 2), which findings are the basis for subsequent shape feature extraction algorithms. In addition, a shape descriptor is introduced and its main novelty is extracting keypoints from both the foreground and the background of binary images (Sec. 3). Furthermore, a theoretical model is introduced to rationalize query spotting in document images, and an algorithmic implementation is designed to adapt the model to noisy and fluctuated handwritten images (Sec. 4). A method for dataset indexing is introduced by using connected components clustering.

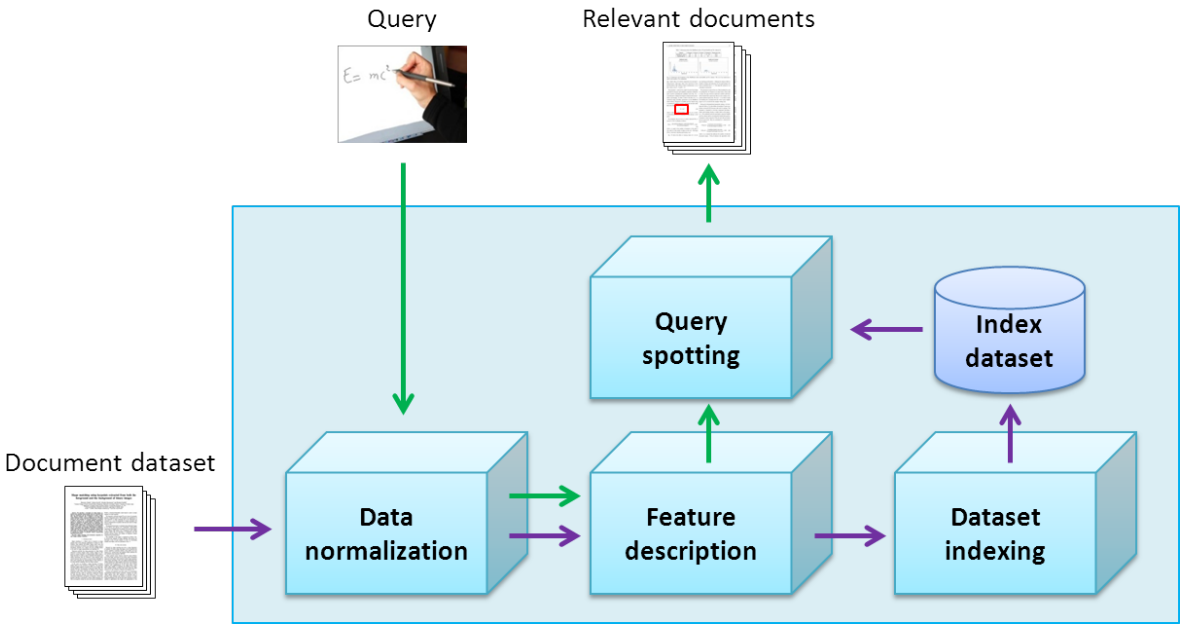


Fig. 1: Flowchart of the proposed system. Green and purple arrows indicate the online and offline processes respectively.

2 Image preprocessing and normalization

Choosing the adequate image representation is crucial to guarantee good retrieval performances. In case of sketches and binarized document images, contours and skeletons are intuitive representations due to their ability to encode the visual information of an object using a limited set of points.

In this section, contours and skeletons are compared to determine the best representation for the proposed system. Comparison is conducted using datasets of thick, elongated, and nearly-thin objects, that are exposed to different image variations (i.e. noise, blurring, size reduction).

The complete study and related references have been reported in details in [1]. In the following, we report essentially results related to contours and skeletons of noisy objects.

2.1 Comparison platform

The experimental platform is generic and consists of extracting shape features from an intermediate image representation which is either a contour or a skeleton (Sec. 2.1.1), and then use them as input to a shape matching algorithm (Sec. 2.1.2). The algorithm's performance is evaluated using objective metrics (Sec. 2.1.3).

2.1.1 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 [2]. 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 illustrates the image representations.

2.1.2 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's centroid (Fig. 3).

The similarity between two images I_1 and I_2 is equal to their histogram intersections, S, computed from their corresponding histograms. S is calculated as follows:

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

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 [3].

2.1.3 Shape matching algorithms and metrics

We 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:

$$Classification\ Accuracy = 100 \times \frac{1}{N} \sum_{k=1}^{N} score_k \tag{2}$$

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

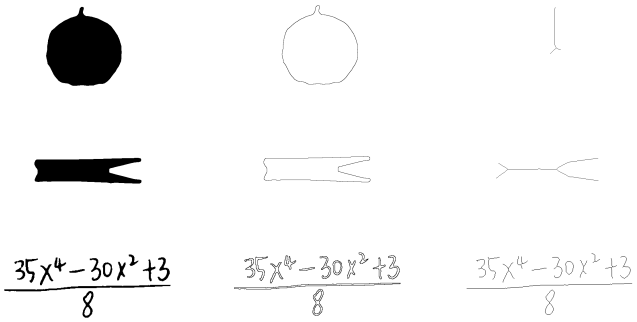


Fig. 2: Datasets and different image representations. Rows from top to bottom: Examples from Dataset 1 (thick objects), Dataset 2 (elongated objects), and Dataset 3 (nearly thin objects). Columns from left to right: Original images, contour images, skeleton images.

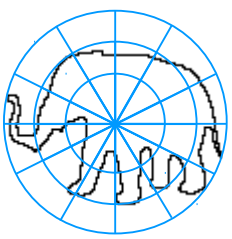


Fig. 3: Feature extraction layout.

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-Measure* metric, that is expressed as a percentage and calculated as follows:

$$F\text{-}Measure = 100 \times \frac{1}{N} \sum_{k=1}^{N} \frac{2 \times precision_k \times recall_k}{precision_k + recall_k} \tag{3}$$

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

$$precision_k = \frac{number\ of\ retrieved\ images\ of\ class\ k}{N_R} \tag{4}$$

$$recall_k = \frac{number\ of\ retrieved\ images\ of\ class\ k}{|class_k|} \tag{5}$$

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 retrieved image set.

2.2 Experimental results

2.2.1 Image datasets

The image datasets include 3 datasets: Dataset 1 which contains 197 images of *thick* objects organized in 20 classes, Dataset 2 which contains 174 images of *elongated* objects organized in 20 classes, and Dataset 3 which contains 200 images of *nearly thin* objects organized in 20 classes. Images of Dataset 1 and Dataset 2 have been collected from the ALOI dataset [4], while images of Dataset 3 were chosen from Zanibbi and Yu's dataset [5]. Fig. 2 shows examples of the images and their derived representations.

2.2.2 Image variations

We analyze the effect of contour noise, blurring, and size reduction. Contour noise is a common challenge for skeletonization algorithms and a biologically plausible image variation. In our 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. This synthetically simulates the binary images affected by border noise that are often produced by scanners or sketching devices.

Image blurring is a widely used image processing procedure and has been used to as a remedy to contour noise [6]. In this experiment, we 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 [7].

Size is a biologically plausible property that is frequently changed. Size change affects the level of details existing in an image; Large images contain more local details, while smaller images have less 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 [8]. This range insures a study of the performances on image that vary from large to small.

2.2.3 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 [9]. The parameter N_R used in *F-Measure* takes the same value as K. The feature histogram dimensions are set to $M = 5$ and $N = 12$, respectively.

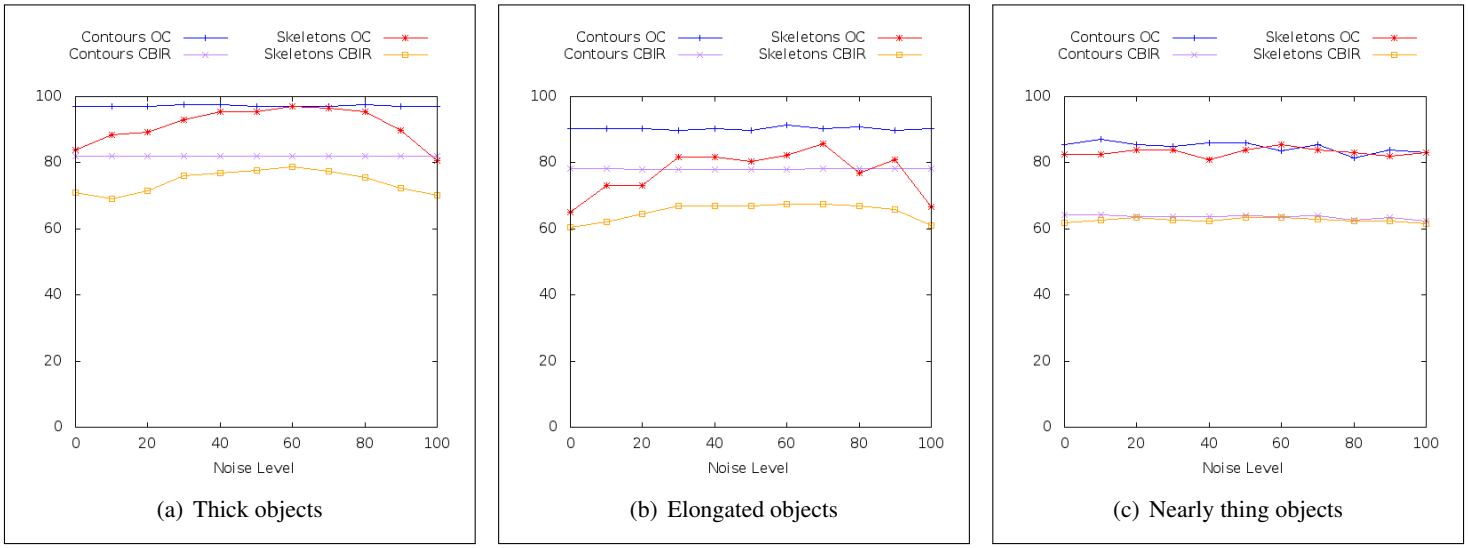
Figure 4 shows the results of OC and CBIR using the two representations of images exposed to noise. Contours outperformed skeletons regardless of the image category. The performances expressed in OC are higher than in CBIR due to the intrinsic difference between the corresponding metrics *Classification 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 we 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 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 [1].

Skeletons noticeably improve in the presence of noise. The improvement is particularly significant in case of thick and elongated images. Skeletons are more sensitive than contours and the performances by the two representations evolve differently. Noise does not affect the contour representation significantly.

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. This shows that additive contour noise results in skeletons that recover more feature points from the original object, and hence improve the uniqueness property of skeletons.

According to our findings, we rely on contours in feature extraction (Sec. 3) and query spotting (Sec. 4).



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

Fig. 4: Performances in presence of noise.

3 Feature extraction and 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. In case of binary images, shape features are essential because of the absence of other types of features (e.g. color).

In this section, a shape descriptor for binary images is introduced. The descriptor extracts keypoints from binary images, hence it is called binary image keypoints (BIK). It operates as follows: First, keypoints are extracted (Sec. 3.1.1). Then, the most important keypoints are selected and the others are filtered out (Sec. 3.1.2). Finally, a feature vector is calculated for each keypoint (Sec. 3.1.3).

An earlier version of this descriptor has been reported [10]. The current version introduces an improved keypoint selection procedure.

3.1 The proposed descriptor

3.1.1 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 neighborhoods are taken as keypoints.

In practice, keypoints are extracted as follows: First, the original image (Fig. 5(a)) is normalized by applying contour detection. Then, the bounding box of the contours is detected and used to generate a frame to contain the object translated towards the center (Fig. 5(b)). The width W_F and height H_F of the frame are as follows:

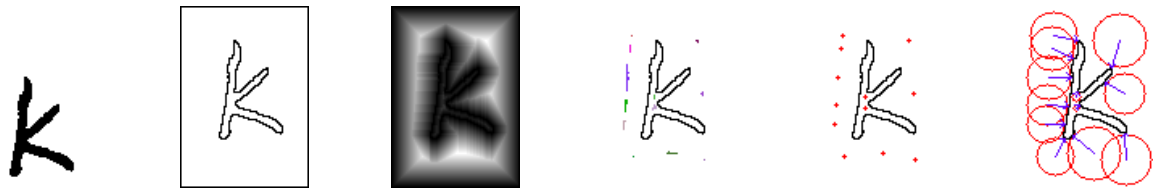
$$W_F = (1 + a)\, W_{BB}, \quad H_F = (1 + a)\, H_{BB} \tag{6}$$

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.

Afterwards, background information is added using the distance transform (DT) [11]. DT is applied to generate a grayscale image where the intensity of each pixel corresponds to its distance from the nearest contour pixel (Fig. 5(c)). Here, the distance between pixels is equal to their Manhattan distance as commonly used in DT implementations [12]. Before applying DT, 1-pixel-width foreground borders are added to the normalized image in order to delimit the object so DT does not systematically generate maxima at the borders.

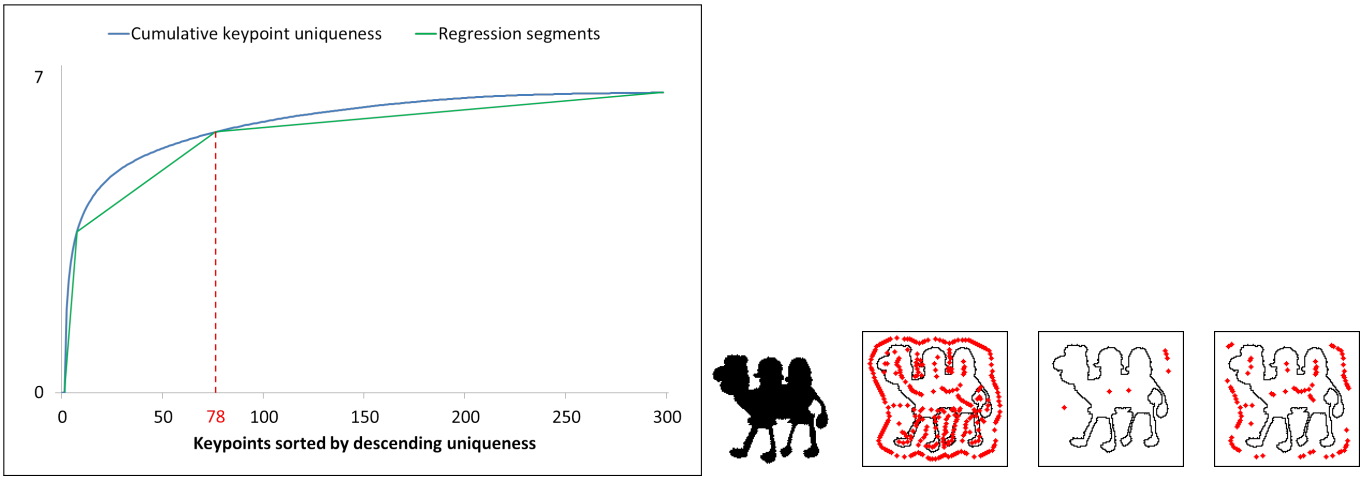
Next, regions of equal maximal intensity are detected on the DT image using a $k \times k$ square window (Fig. 5(d)). Finally, the detected regions are represented using their centers of masses which are taken as keypoints (Fig. 5(e)).

Regions of equal maximal intensity detection is done 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.



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

Fig. 5: 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) (b) (c) (d) (e)

Fig. 6: Keypoint selection: (a) Curve approximation by three segments applied on image (b), (c) 298 keypoints are extracted in total, (d) 7 keypoints extracted at the first segment end, (e) 78 keypoints finally selected.

3.1.2 Keypoint selection

The extracted keypoints might not all be necessary due to keypoint redundancy, and some keypoints are erroneously chosen by noise and contour perturbations. In this step, extracted keypoints are ranked according to a measure of their importance. Then, a number of keypoints is automatically selected.

A measure of keypoint *uniqueness* is introduced for keypoint ranking and selection. We define a *unique* keypoint as a one that has few keypoints in its neighborhood in order to reduce redundancy, and that is not very close to the shape contours or frame borders in order to reduce sensitivity to local details perturbations. Formally, the uniqueness $\gamma(i)$ of a keypoint K_i is calculated as follows:

$$\gamma(i) = \frac{d_i}{1 + n_i} \tag{7}$$

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 b_j to K_i.

We observe that the range of keypoint uniqueness indicates three types of keypoints. The first type corresponds to few keypoints with extreme uniqueness 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.

Keypoints are ranked in their descending uniqueness measures. Then, the *cumulative keypoint uniqueness* $\Gamma(i)$ is calculated as follows:

$$\Gamma(i) = ln\,(\sum_{j=1}^{i} \gamma(j)\,) \tag{8}$$

$\Gamma(i)$ expresses the cumulative uniqueness of a number i of ranked keypoints. Fig. 6(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 reduce the number of keypoints without compromising the descriptor's distinctiveness, we filter out the keypoints of the third type, which are redundant and too close to pixels of the contours and frame. For this purpose, we approximate the curve of Γ by three segments that minimize the area between the segments and the curve using a two dimensional search of the two points that connect the segments. Then, keypoints corresponding to the first and second types are selected. Fig. 6 illustrates this procedure.

3.1.3 Feature represenation and matching

The last step is to calculate a feature vector to each keypoint K_i. For this purpose, we 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:

$$r_i = \alpha \times d_i \tag{9}$$

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:

$$\mathcal{X}^2(h_1, h_2) = \frac{1}{2} \sum_{j=0}^{N_B-1} \frac{[h_1(j) - h_2(j)]^2}{h_1(j) + h_2(j)} \tag{10}$$

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:

$$d(I_1, I_2) = \frac{1}{N_1} \sum_{i=0}^{N_1-1} \min_{0 \le j < N_2} \{\mathcal{X}^2(h_i^1, h_j^2)\} \tag{11}$$

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, we express the distance between two images I_1 and I_2 as follows:

$$D(I_1, I_2) = \frac{d(I_1, I_2) + d(I_2, I_1)}{2} \qquad (D \in [0, 1]) \tag{12}$$

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. 5(f)).

3.2 Evaluation

BIK's evaluation is done using the following public datasets (Fig. 7): The Kimia 216 dataset [13] and the MPEG-7 dataset [14] contain silhouette images that are neat and which contain single component objects. Zanibbi and Yu's dataset [5] contains handwritten mathematical expressions which exhibit handwriting fluctuations and component displacement, which also appear in Liang et al.'s dataset [15] of hand-drawn diagram sketches. The Tobacco 800 dataset [16] contains logo images that are taken from scanned documents and they are the noisiest compared to the other datasets.

3.2.1 Parameter setting

The parameters are set as follows: The parameter for setting the normalization frame's dimensions is set $a = 0.25$ to insure a scale-invariant frame with space between its borders and the object contours. A keypoint K_j is considered close to a keypoint K_i if the distance between them is less than $\frac{d_i}{4}$, where d_i is the distance between keypoint K_i and its closest contour or frame point. The radial and angular numbers of bins in the shape descriptor are set in a way to avoid extreme cases. A small number of bins compromise the descriptor's distinctiveness, while a larger number of bins cause sensitivity to noise and fluctuations. Based on these considerations, we set their values to be 4 distance bins and 8 angle bins. As for the constant for setting the keypoint-dependent feature layout radius, we set it heuristically $\alpha = 1.5$ in order to insure that the feature extraction layout's bins of one unit distance can reach a sufficient number of pixels, which are the ones located at a close distance to the keypoint relative to the distance to its closest contour point. Empirically, we set the size of the keypoint sampling window to $k = 3$ to recover the maximum number of keypoints before subsequent selection.

3.2.2 Results

The proposed descriptor is evaluated using different metrics according to each dataset. In case of Kimia's dataset, we 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 [17]. In case of the Zanibbi and Yu's dataset, Tobacco 800 logos, and MPEG-7 dataset, the *precision at n* metric [17] is used. This metric is denoted $P@n$, and it is calculated as follows:

$$P@n = \frac{|\{n \textit{ retrieved images}\} \cap \{\textit{relevant images}\}|}{|\{n \textit{ retrievd images}\}|} \tag{13}$$

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

BIK yields high results on all datasets. According to each dataset corresponding metric, BIK gives 93.83% on Kimia dataset, 75.48% on MPEG-7 dataset, 82.74% on Zanibbi and Yu's dataset, 83.83% on Sketch Magik dataset, and 81.65% on Tobacco logos dataset.

Fig. 7: Samples of the dataset images. Rows from top to bottom: Kimia's dataset (216 images) [13], MPEG-7 dataset (1400 images) [14], Zanibbi and Yu's dataset (200 images) [5], Sketch Magik dataset (1086 images) [15], and Tobacco logo dataset (412 images) [16].

4 Query spotting in document images

In order to retrieve document images with a user query, the query needs to be localized, or *spotted*, inside the document image. In this section, a method for content-based document image retrieval (CBDIR) that is both segmentation and recognition-free is introduced. Avoiding segmentation and recognition is intended to allow the generality of the method and its applicability in problem domains other than text queries and documents. The proposed method has been reported in [18].

4.1 Theoretical model

Our 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 $\boldsymbol{x}$. By introducing the likelihood $p(\boldsymbol{x}|A^R)$, the posterior probability $P(A^R|\boldsymbol{x})$ can be evaluated.

4.1.1 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}, we 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 $A = \{C_{\phi(i)}^{DOC}\}_{i=1}^M$, 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 the majority of classes $\{\omega_i\}_{i=1}^M$. A^R denotes the event that set A is a relevant candidate. Assuming that $\{P(\omega_i|C_{\phi(i)}^{DOC})\}_{i=1}^M$ are independent, we take as the initial prior probability of A^R as follows:

$$P(A^R) = \prod_{i=1}^{M} P(\omega_i|C_{\phi(i)}^{DOC}) \tag{14}$$

4.1.2 Observation

Eq. 14 does not take into account the locations of components relative to each other inside a candidate A. Therefore, multiple (K) observations $\boldsymbol{x} = [x_1 \dots x_K]$ concerning the global resemblance and suitability of A are introduced by way of a likelihood function $p(\boldsymbol{x}|A^R)$.

4.1.3 Inference

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

$$P(A^R|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|A^R) \times P(A^R)}{p(\boldsymbol{x})} \tag{15}$$

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

4.1.4 Decision function

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

$$D(A) = ln(P(A^R|\boldsymbol{x})) = ln(p(\boldsymbol{x}|A^R)) + \sum_{i=1}^{M} ln(P(\omega_i|C^{DOC}_{\phi(i)})) \tag{16}$$

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

4.2 Algorithmic implementation

Our algorithm proceeds as follows: First, features are extracted from the components of I_Q and I_{DOC} (Sec. 4.2.1) and used to detect candidate occurrences of I_Q in I_{DOC} (Sec. 4.2.2). Next, a score is calculated for each candidate to express its relevance to the query (Sec. 4.2.3).

4.2.1 Component feature extraction and matching

A feature vector $\boldsymbol{V}$ 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. 3). 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:

$$S(C_i, C_j) = \sum_{r=0}^{R-1} \sum_{\theta=0}^{\Theta-1} min(\boldsymbol{V}^i_{r,\theta}, \boldsymbol{V}^j_{r,\theta}) \tag{17}$$

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 $P(\omega_i \mid C^{DOC}_j)$ defined in Sec. 4.1.1.

4.2.2 Detection of query occurrence candidates

One component of the query, that we 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 $B = \{C^{DOC}_j \mid S(\hat{Q}, C_j) \geq \alpha : 1 \leq j \leq N\}$. 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. 8). The bounding box's dimensions are calculated as follows:

$$(W, H) = (W_Q, H_Q) \times \frac{size\ of\ \hat{G}}{size\ of\ \hat{Q}} \times \beta \tag{18}$$

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.

$$\frac{35x^4 - 30x^2 + 3}{8}$$

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

Fig. 8: Illustration of the bounding box-based spotting procedure. Left: An example of a handwritten query with the *main component* $\hat{Q}$ highlighted in green. Right: 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).

4.2.3 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, $p(\boldsymbol{x}|A^R)$ is estimated as a multidimensional observation $\boldsymbol{x} = [a\ b\ c]$, 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. 4.2.1). 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 $\boldsymbol{x}$ are independent, the combined probability $p(\boldsymbol{x}|A^R) = p(a|A^R)\ p(b|A^R)\ p(c|A^R)$ is integrated in Eq. 16, which gives:

$$D(A) = ln(P(a|A^R)) + ln(P(b|A^R)) + ln(P(c|A^R)) + \sum_{i=1}^{M} ln(P(\omega_i|C^{DOC}_{\phi(i)})) \tag{19}$$

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

$$score(A, \gamma) = ln(1 + a) + ln(1 + b) + ln(1 + c) + \frac{\gamma}{N_{\hat{Q}}} \sum_{u=1}^{N_A} ln(1 + \max_{1 \le i \le M} S(C_i, C_u)) \tag{20}$$

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. 16). When component disconnectedness and merging in a candidate A are significant, the quantity of the score $\sum_{u=1}^{N_A} ln(1 + \max_{1 \le i \le M} S(C_i, C_u))$ 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.

4.3 Experimental Results

Comparative evaluation is done with Zanibbi and Yu's method [5]. This method is adequate for comparison as it is based on segmentation using horizontal and vertical cutting and tree-based feature representation, which are often used in CBDIR of mathematical expressions. Their dataset contains 200 printed 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. The images are subjected to binarization [7] followed by contour detection.

4.3.1 Parameter setting

The radial and angular numbers of sections in the shape descriptor (Sec. 4.2.1) 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. Based on these considerations, we set their values to be $R = 5$ and $\Theta = 10$.

The rest of the parameters are set as follows: The CCs similarity threshold is set $\alpha = 0.7$. The parameter to control the size of the spotting bounding box is set $\beta = 1.1$ to account for component displacement. The parameter to mitigate the effect of disconnectedness and merging of CCs is set $\gamma = 2$. The number of main components is set depending on the number of query components $N_{\hat{Q}} = \frac{3}{4}M$, and a group is discarded if the number of its components is different to the number of query components by $\frac{M}{4}$, where M is the number of query components.

4.3.2 Comparative evaluation

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 document image dataset is indexed to allow a reasonable retrieval time [19]. A maximum of 100 clusters are selected from the index, and a maximum of 2000 groups are selected for each query component. These restrictions are introduced for the sake of efficiency.

Comparative evaluation is done using the *P-Recall* and *A-Recall* metrics that are used in [5]. They are calculated for n retrieved images as follows:

$$P\text{-}Recall = \frac{\#\ relevant\ retrieved\ images}{\#\ relevant\ images\ in\ dataset} \times 100, \qquad A\text{-}Recall = \frac{Candidate\ bound.\ box\ area}{Ground\ truth\ bound.\ box\ area} \times 100 \tag{21}$$

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.

Results of the proposed method are presented when the candidate score (Sec. 4.2.3) is used, and when BIK (Sec. 3) is used for candidate re-ranking after initial ranking of 20 candidate groups by the candidate score. Table 1 compares the proposed method with Zanibbi and Yu's algorithm. The proposed method gives better results both when the candidate score and BIK re-ranking are used. BIK improves significantly retrieval performances especially in case of handwritten queries. For printed queries, results are slightly better when the candidate score is used.

4.4 Conclusion

In this section, a CBDIR method that is segmentation and recognition-free has been presented. It is conform with a theoretical model and it has been designed to overcome noises and fluctuations of handwritten images. Comparative evaluation using mathematical expression queries and document images demonstrated that the proposed method is competitively effective.

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

Queries		Printed		Handwritten	
	n	*P-Recall*	*A-Recall*	*P-Recall*	*A-Recall*
Our method	1	100.0%	94.28%	40.0%	27.83%
	5	100.0%	96.78%	63.5%	51.15%
	10	100.0%	96.78%	73.5%	57.92%
Our method + BIK	1	92.5%	89.29%	54.0%	47.84%
	5	100.0%	96.29%	70.0%	59.89%
	10	**100.0%**	**96.78%**	**75.0%**	**62.43%**
Zanibbi and Yu [5]	1	.	90%	38.6%	26.7%
	5	.	90%	54.9%	39.8%
	10	.	90%	63.2%	43.3%

References

[1] H. Chatbri, K. Kameyama, and P. Kwan, "A comparative study using contours and skeletons as shape representations for binary image matching," *Pattern Recognition Letters (In Press)*, 2015.

[2] F. Zhang *et al.*, "An improved parallel thinning algorithm with two subiterations," *Optoelectronics Letters*, vol. 4, no. 1, pp. 69–71, 2008.

[3] D. Navon, "Forest before trees: The precedence of global features in visual perception," *Cognitive psychology*, 1977.

[4] J. M. Geusebroek, G. J. Burghouts, and A. W. M. Smeulders, "The amsterdam library of object images," *International Journal of Computer Vision*, vol. 61, no. 1, pp. 103–112, 2005.

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

[6] H. Chatbri and K. Kameyama, "Using scale space filtering to make thinning algorithms robust against noise in sketch images," *Pattern Recognition Letters*, vol. 42, pp. 1–10, 2014.

[7] N. Otsu, "A threshold selection method from gray-level histograms," *Automatica*, vol. 11, no. 285-296, pp. 23–27, 1975.

[8] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*. " O'Reilly Media, Inc.", 2008.

[9] M. Maier, M. Hein, and U. von Luxburg, "Optimal construction of k-nearest-neighbor graphs for identifying noisy clusters," *Theoretical Computer Science*, vol. 410, no. 19, pp. 1749–1764, 2009.

[10] H. 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, 2012.

[11] A. Rosenfeld and J. L. Pfaltz, "Sequential operations in digital picture processing," *Journal of the ACM (JACM)*, 1966.

[12] D. G. Bailey, "An efficient euclidean distance transform," in *Combinatorial Image Analysis*, pp. 394–408, Springer, 2005.

[13] T. Sebastian, P. Klein, and B. Kimia, "Recognition of shapes by editing shock graphs," in *International Conference on Computer Vision*, vol. 1, pp. 755–755, IEEE Computer Society, 2001.

[14] M. Bober, "MPEG-7 visual shape descriptors," *IEEE Transactions on circuits and systems for video technology*, 2001.

[15] S. Liang, J. Luo, L. Wenyin, and Y. Wei, "Sketch matching on topology product graph," *IEEE Trans. PAMI*, 2015.

[16] G. Zhu and D. Doermann, "Automatic document logo detection," in *International Conference on Document Analysis and Recognition (ICDAR)*, pp. 864–868, 2007.

[17] Baeza-Yates *et al.*, *Modern information retrieval*, vol. 463. ACM press New York, 1999.

[18] H. Chatbri, K. Kameyama, and P. Kwan, "Towards a segmentation and recognition-free approach for content-based document image retrieval of handwritten queries," in *Asian Conference on Pattern Recognition (ACPR, to appear)*, IAPR, 2015.

[19] H. Chatbri and K. Kameyama, "Document image dataset indexing and compression using connected components clustering," in *IAPR International Conference on Machine Vision Applications (MVA)*, pp. 267–270, IEEE, 2015.