2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

A Modular Approach for Query Spotting in Document Images and Its Optimization Using Genetic Algorithms

Houssem Chatbri, Paul Kwan, and Keisuke Kameyama

***Abstract*— Query spotting in document images is a subclass of Content-Based Image Retrieval (CBIR) algorithms concerned with detecting occurrences of a query in a document image. Due to noise and complexity of document images, spotting can be a challenging task and easily prone to false positives and partially incorrect matches, thereby reducing the overall precision of the algorithm. A robust and accurate spotting algorithm is essential to our current research on sketch-based retrieval of digitized lecture materials. We have recently proposed a modular spotting algorithm in [1]. Compared to existing methods, our algorithm is both application-independent and segmentation-free. However, it faces the same challenges of noise and complexity of images. In this paper, inspired by our earlier research on optimizing parameter settings for CBIR using an evolutionary algorithm [2][3], we introduce a Genetic Algorithm-based optimization step in our spotting algorithm to improve each spotting result. Experiments using an image dataset of journal pages reveal promising performance, in that the precision is significantly improved but without compromising the recall of the overall spotting result.**

I. Introduction

Content-Based Image Retrieval (CBIR) is the area of research concerned with designing image search systems that use the image visual content instead of text keywords [4]. The exponential growth in the scale of image databases and the progress achieved in Pattern Recognition and Image Processing paved the way for many applications of CBIR such as in medical imaging [5][6], copyright protection [7], cultural heritage analysis [8][9], trademark retrieval [10][11], Sketch-Based Image Retrieval [12][13], query spotting in document images [14], etc.

In order to improve CBIR systems performances, Evolutionary Algorithms (EA) have been utilized. Particularly, Genetic Algorithms (GA) [15] and Particle Swarm Optimization (PSO) [16] have been explored to make CBIR systems adaptive to the class of images they use [2] and to allow a highly semantic retrieval [17].

We focus in this work on the area of query spotting, that is finding occurrences of a query, in a document image. This area of research has gained attention as early as digital libraries started to become popular [14][18]. In case of printed documents with standard fonts and high resolutions, spotting can be implemented using Optical Character Recognition (OCR) by *recognize-then-retrieve* approaches [19][20]. However, when documents are old, handwritten, or multilingual, more sophisticated and *recognition-free* approaches become needed [21][22]. Such methods usually extract features from the query and match them in the document image. It is also common that spotting methods deal with particular classes of queries such as words [23] or mathematical expressions [24]. In these cases, regions of interest that are identical to the query class are usually extracted to facilitate the spotting process.

Due to noise and complexity of document images, spotting systems are prone to errors [25][26]. Particularly, in case of recognition and segmentation free approaches, false positive or partially incorrect matches containing *residuals* (Sec. III-B, Fig. 5) (i.e. irrelevant parts of the query detected occurrences that have been matched positively) can be detected. This issue affects the precision of the system and it is a challenge for subsequent applications after spotting. Therefore, a post-processing that reduces the number of residuals in spotting results is critically important.

In this work, we introduce a spotting approach that operates in two stages: First, a modular framework based on pruning and voting detects candidate occurrences of the query. Second, a Genetic Algorithm (GA) is used to optimize each spotting result. Compared to the state-of-the-art, our approach is recognition and segmentation free. In addition, no existing method has tried involving GA in spotting queries in document images.

The remainder of this paper is organized as follows: Sec. II overviews the state-of-the-art of using EAs in optimizing CBIR systems, and references on query spotting in document images. The proposed approach for spotting and result optimization using GA is explained in detail in Sec. III. Experimental results are presented and discussed in Sec. IV. Sec. V announces our conclusions and future directions.

II. Related work

A. Use of EA in CBIR

EA have been used to optimize CBIR systems in various aspects. For the purpose of optimizing feature extraction, Torres et al. used GA to determine an optimal combination function of multiple feature descriptors and proved that it outperforms conventional weighting combination [27]. In [28], Kiranyaz et al. presented an approach for feature distinctiveness optimization using PSO.

Houssem Chatbri is with the Graduate School of Systems and Information Engineering, Department of Computer Science, University of Tsukuba, Tsukuba City, Japan (email: chatbri@adapt.cs.tsukuba.ac.jp).

Paul Kwan is with the School of Science and Technology, University of New England, Armidale NSW 2351, Australia (email: paul.kwan@une.edu.au).

Keisuke Kameyama is with the Faculty of Engineering, Information and Systems, University of Tsukuba, Japan (email: keisuke.kameyama@cs.tsukuba.ac.jp).

978-1-4799-1488-3/14/$31.00 ©2014 IEEE

In [29], Jadhav and Patil aimed to optimize feature matching, and used a GA to find query similar images using low level features such as color, texture, and shape. They proved that GA can be used as a matching mechanism in CBIR.

In our previous work [2], we used PSO to improve image similarity evaluation by automatically optimizing the CBIR system parameters using the suitability of the result. First, image similarity is calculated using Relaxation Matching [30]. Then, a PSO-based optimization scheme uses a measure for query-result-discrepancy to optimize the system parameters. In another similar work, we used PSO to tune the system using *relevance feedback* provided by users [3].

EA have been used to optimize CBIR systems in other aspects such as large database indexing [31] and user intention inference [32].

B. Recognition free spotting methods

As stated above, methods for spotting can be categorized as OCR-based or recognition-free. In this section, we review references of recognition-free methods, as our contribution is of this category. We refer the reader to references [18, 32-34] for information on OCR-based methods.

In order to account for noise and complexity of document images, spotting methods usually rely on a priori knowledge to perform image segmentation, regions of interest extraction, and to use domain-specific features.

In [18], Manmatha et al. introduced an early work on spotting that starts by segmenting the document image into words using a priori knowledge about the distance between characters and words. After segmentation, word query spotting is done by applying one of two algorithms which estimate the shift between the query and the words in the document image using the Euclidean distance and Scott and Longuet Higgins' algorithm.

Another segmentation-based method has been presented by Rath and Manmatha [33]. The authors tackled the problem of word spotting in historical documents. After segmenting the document image into words, the ink pixel distribution is used for word feature extraction. Matching is done using Dynamic Time Warping [34].

In [24], Zanibbi and Yu introduced an approach for mathematical expression spotting using handwritten queries, that works as follows: Recursive X-Y cutting [35] is used to segment the query and document image and index them by X-Y trees, 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.

Other word spotting methods use a priori knowledge about the document's language. For instance, Lu and Tan presented a method for word spotting in Chinese documents where they use a modified Hausdorff distance tuned for Chinese characters [25]. Likewise, Sari and Kefali presented a method for word spotting in Arabic documents, based on specific features of Arabic characters (e.g. diacritics, loops, etc.) [36].

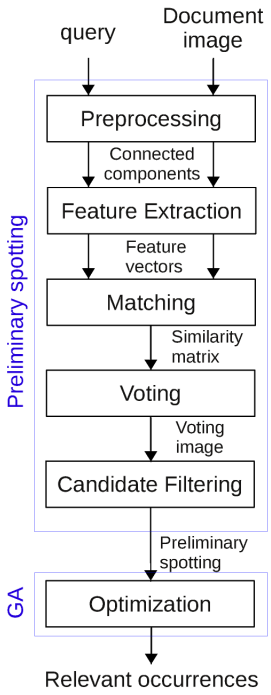


Fig. 1. Overview of the proposed approach.

C. Direction of this research

Our earlier work on using EA in optimizing CBIR systems [2][3], and the above state-of-the-art review on spotting methods lead us to make the following observations:

- EA can facilitate the optimization of CBIR systems in different aspects.
- Most existing spotting methods rely on segmentation in order to extract regions of interest from the document image, and use a priori knowledge such as the query class or document language.

In this paper, we aim to develop an application-independent spotting approach that is recognition and segmentation free. As spotting is a subclass of CBIR, the successful use of EA in optimizing CBIR systems motivated us to use EA in optimizing the proposed approach's performances by further improving spotting precision.

III. The Proposed Approach

The proposed approach operates in two stages: preliminary spotting and spotting optimization. Fig 1 shows an overview of the proposed approach and Algorithm 1 summarizes the first stage. First, the query and document image are subjected to a Preprocessing step that is charged of noise reduction and *connected components* extraction (for the ease of description, we will refer to connected components simply as components).

Then, features are extracted from components and represented in *feature vectors*.

Afterwards, the feature vectors corresponding to the query components and the document image component are matched and the similarity scores are stored in a *similarity matrix*.

Algorithm 1 Preliminary spotting

(1)	$(\{C_i^Q\}_{i\leq M}, \{C_j^{DOC}\}_{j\leq N})$	$\leftarrow$	Preprocessing (I_Q, I_{DOC})
(2)	$(\{\overrightarrow{H}_i^Q\}_{i\leq M}, \{\overrightarrow{H}_j^{DOC}\}_{j\leq N})$	$\leftarrow$	Feature Extraction ($\{C_i^Q\}_{i\leq M}, \{C_j^{DOC}\}_{j\leq N}$)
(3)	$S_{M,N}$	$\leftarrow$	Matching ($\{\overrightarrow{H}_i^Q\}_{i\leq M}, \{\overrightarrow{H}_j^{DOC}\}_{j\leq N}$)
(4)	I_V	$\leftarrow$	Voting ($\{C_i^Q\}_{i\leq M}, \{C_j^{DOC}\}_{j\leq N}, S_{M,N}$)
(5)	$\{G_k\}_{k\leq K}$	$\leftarrow$	Candidate Filtering (I_V)

Next, *similarity matrix* is used to vote for locations of candidate occurrences of the query in the document image.

Then, groups of document image connected components are formed around each voting point.

Finally, a GA is used to optimize each spotting result by removing residuals in partially incorrect matches.

A. Preliminary spotting

Algorithm 1 summarizes the preliminary spotting steps. Throughout this section, each step will be explained in detail.

*1) **Preprocessing**:* Document images are usually prone to noise due to the quality and age of the document and imperfection of scanning devices. Therefore, a preprocessing step for noise reduction and input normalization is needed. For this purpose, we use our previously reported *Adaptive Thinning Framework (ATF)* [37][38]. ATF produces 1-pixel width representation of images, and it is robust against noise compared to conventional thinning algorithms.

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

*2) **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 [39]. 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 are 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 $\{C_j^{DOC}\}_{j\leq N}$.

*3) **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:

$$S(i,j) = \sum_{k=0}^{K-1}\sum_{l=0}^{L-1} min(H_{kl}^q, H_{kl}^d) \tag{1}$$

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 saving computations, 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 descriptor are prone to false positive. Therefore, before applying this method, we apply another pruning mechanism.

Our mechanism uses the hypothesis assuming that if a document image component C_j^{DOC} is visually similar to a query component C_i^Q, it should be nearly similar or dissimilar to the remaining components of the query $\{C_k^Q\}_{k\neq i}$, in the same way as C_i^Q.

We implement this hypothesis as follows:

1) The query auto-correlation matrix, $S_{M,M}^Q$, is calculated by matching the query's components against each others using the Histogram Intersection measure (Eq. 1). S^Q is symmetric with 1-values on the diagonal.
2) If a document image component $C_{j_0}^{DOC}$ is found to be similar to a query component $C_{i_0}^Q$, that is $S(i,j) > \alpha$, where α is a similarity threshold, the Euclidean distance $D(i_0, j_0)$ between $\{S^Q(i,i_0)\}_{i\leq M}$ and $\{S(i,j_0)\}_{i\leq M}$ is calculated as follows:

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

When $D(i_0,j_0) > \theta$, where θ is a dissimilarity threshold, it means that $C_{j_0}^{DOC}$ does not keep a similarity pattern to $\{C_k^Q\}_{k\neq i}$ in the same way as $C_{i_0}^Q$ does. In such case, $C_{j_0}^{DOC}$ is discarded.

The output of this step is *similarity matrix* S after pruning

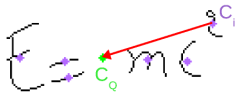


Fig. 2. Illustration of the component centroid c_i, the query centroid c_Q, and $\overrightarrow{c_i c_Q}$. The circles in purple highlight the components' centroids.

of false positive and small similarity scores.

4) ***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 Mat^i, 1-valued cells show candidate locations voted for by similar $\{C_j^{DOC}\}_{j \leq N}$, and 0-valued cells show absence of voting. A voting operation is determined using the following information:

- The centroid c_j of C_j^{DOC}.
- The displacement vector $\overrightarrow{c_i c_Q}$ connecting the centroid c_i of C_i^Q and the centroid c_Q of the query (Fig. 2).
- The similarity score $S(i,j)$ of matching C_i^Q and C_j^{DOC}.

Next, the *voting vector* $\overrightarrow{V_j}$, originating from c_j, parallel to $\overrightarrow{c_i c_Q}$ and having the norm calculated as follows:

$$|\overrightarrow{V_j}| = |\overrightarrow{c_i c_Q}| \times \gamma \tag{3}$$

where γ is a scale normalization factor calculated using the radius of the enclosing circles of C_i^Q and C_j^{DOC}. $\overrightarrow{V_j}$ points to the *voting point*, that is the center of the candidate occurrence.

Then, the cells of Mat^i located in a circular region around the voting point are made 1-valued. The voting is made in a circular region in order to account for components displacement, and the radius r of the circular region is calculated as follows:

$$r = r^Q \times \gamma \times S(i,j) \times \delta \tag{4}$$

where r^Q is the radius of the query's enclosing circle, and δ is a parameter to control the size of the voting region. Fig. 3 illustrates a voting operation in Mat^i superposed on the document image.

After generating a *voting matrix* Mat^i corresponding to each query component C_i^Q, the matrix Mat^{mean} holding the average of values of votes $\{Mat^i(x,y)\}_{i \leq M}$ is calculated. Then, the entries of Mat^{mean} are mapped into grayscale intensities and used to produce the voting image I_V.

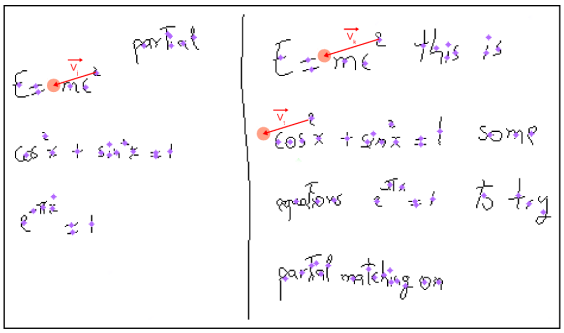


Fig. 3. The *voters* are C_j^{DOC} (the symbol '2' on the left), C_k^{DOC} (the symbol '2' on the right), and C_t^{DOC} (the symbol '2' on the center). Their *voting vectors* are $\overrightarrow{V_i}$, $\overrightarrow{V_k}$, and $\overrightarrow{V_t}$. The *voting vectors* differ in norms to adapt for the size change. The circles in purple highlight the components' centroids.

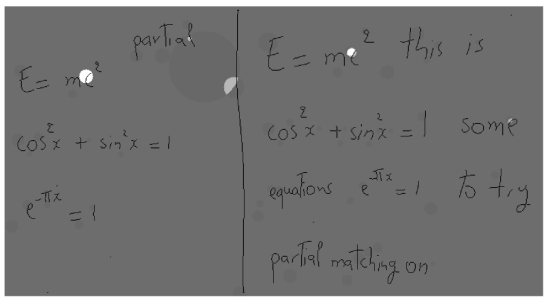


Fig. 4. The voting image I_V (superimposed on the document image for the sake of illustration). Bright spots show regions of high voting scores.

Fig. 4 shows an example of the voting image I_V superposed on the document image. I_V is the output of this step.

5) ***Candidate filtering****:* The voting image I_V has been produced using votes from groups of components C_j^{DOC}. In this step, *voting groups* are identified, extracted from the document image, and classified as relevant or irrelevant.

First, the centers of the voting regions $\{c_k^V\}_{k \leq K}$, where K is the total number of voting regions, are extracted from I_V by applying Distance Transform [40] and an intensity maxima detection algorithm.

Next, a *voting group* G_k is formed around each voting center by finding the components C_j^{DOC} which voting vectors (Eq. 3) point to a location inside the voting circular region which center is c_k^V and radius is calculated as in Eq. 4.

At this stage, K *voting groups* are extracted from the document image. A preliminary filtering of irrelevant groups is done by evaluating the scale consistency of the components of each group; The idea is that a relevant voting group should have components that hold nearly equal scale factor γ with their corresponding query components. This is insured by calculating the scale factor variance $\sigma(\gamma)$. If $\sigma(\gamma)$ is large, it means that the *voting group* is formed of components having inconsistent scales.

The output of this step is the detected voting groups after scale-consistency filtering.

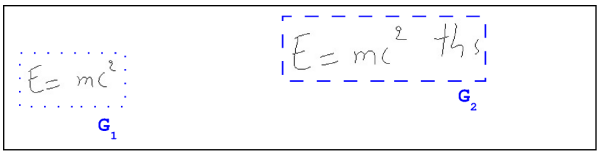


Fig. 5. Preliminary spotting result with 2 voting groups (Image's height reduced because of the space limit). *Voting Group* G_2 on the right contains 3 residuals which are the letters "t", "h", and "s".

B. Spotting optimization using Genetic Algorithms

Genetic Algorithms (GA) are optimization algorithms that model genetic evolution [41]. In a population of candidate solutions, the characteristics of each individual are expressed using *chromosomes*, and genetic operators such as *crossover* and *mutation* are used to evolve the population towards an optimal solution. A *Fitness function* is used to estimate the distance of an individual from the optimal solution.

Basically, *chromosomes* are encoded in binary strings. During the population evolution, chromosomes that fit best, i.e. calculate the best values of the *fitness function*, are selected to give *offspring* by using *crossover*. *Mutation* is used in order to insure wide exploration of the solution space and prevent local optima.

In the following, we go through our GA modelization in details. We use basic GA as described above. Other variants of GA have been introduced for particular purposes such as GA using real-valued representations [42], multi-areas Genetic Algorithms [43], Genetic Programming [44], etc.

1) ***GA modelization****:* The purpose of using GA in our approach is to optimize the spotting result by removing *residuals* (Fig. 5) which might exist in the output of the preliminary spotting stage (Sec. III-A).

For each voting group G_k formed by N_k components $\{C_v^{DOC}\}_{v \leq N_k}$, a GA operates in parallel on a *population* P^k to enhance the spotting result. The GA operates as follows: Initially, P^k contains a fixed number of *chromosomes* $\{Ch_u^k\}_{u \leq ||P||}$ initialized with random values (Sec. III-B.2). Then, a *fitness* $f(Ch_u^k)$ is calculated for each *chromosome* (Sec. III-B.3). Next, best fit *chromosomes* are selected and *crossover* is used to produce their *offspring* (Sec. III-B.4). In addition to *crossover*, *mutation* is used to insure wide exploration of the solution space (Sec. III-B.5).

The evolution terminates automatically when *population* P^k reaches a stable state. The stability is estimated by calculating the *fitness variance* of the population *chromosomes*. The best fit *chromosome* is then the output of the algorithm.

2) ***Population****:* For each *voting group* G_k corresponds a *population* P^k that contains *chromosomes* $\{Ch_u^k\}_{u \leq ||P^k||}$. A *chromosome* Ch_u^k has N_k genes b_{uv}^k, where $v \leq N_k$. The gene b_{uv}^k activates or deactivates a corresponding component C_v of *voting group* G_k. Fig. 6 shows an example of a *chromosome*.

3) ***Fitness function****:* The fitness function $f(Ch_u^k)$ is equal to the similarity score obtained by comparing the image constructed from the *chromosome* Ch_u^k and the query image I_Q using a shape descriptor. We use *Support Region Descriptor (SRD)* [12] for this purpose:

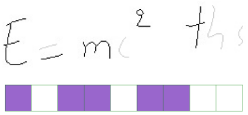


Fig. 6. Illustration of a *chromosome* generated from G_2 (Fig. 5). Colored boxes refer to active *genes* and empty boxes refer to inactive *genes*. In this illustration, the order of components is assumed to be from left to right.

$$f(Ch_u^k) = S_{SRD}(I_{Ch_u^k}, I_Q) \tag{5}$$

where the similarity measure $S_{SRD} \in [0, 1]$ expresses visual similarity in case of large values, and dissimilarity in case of small values. SRD extracts a 2-dimensional histogram for each image, and uses the Histogram Intersection measure to express the similarity between two images.

4) ***Crossover****:* In each iteration of the algorithm, the chromosomes are sorted in decreasing *fitness*. Then, the $\frac{N_k}{2}$ least fit chromosomes are replaced by the *offspring* of the $\frac{N_k}{2}$ fittest chromosomes. *Genes* of the *offspring* are produced using a voting procedure involving 3 parents; Each new *gene* $b_v^{offspring}$ (indexes modified for simplicity) is determined as follows:

$$b_v^{offspring} = \begin{cases} active, & \text{if } \sum_{l=1}^{3} \phi(b_v^{parent(l)}) \geq 2 \\ inactive, & \text{otherwise} \end{cases}$$

where $\phi(b_v^{parent(l)})$ is a mapping function that returns 1 if the *gene* of parent l is active, and 0 if it is inactive.

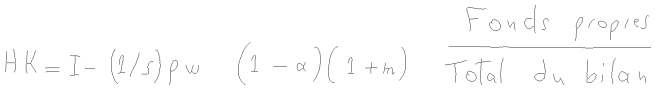
5) ***Mutation****:* It has been demonstrated that an initial large mutation rate that decreases exponentially as a function of the number of algorithm iterations improves convergence speed and accuracy [41]. Based on this finding, we implement *mutation* as described in Algorithm 2: For each iteration, a parameter $\rho(t)$ corresponding to iteration t is calculated to control the mutation probability. Initially, $\rho(0) = 100$. A constant β is used to adjust the decreasing speed of ρ. Then, a random number $0 \leq rand \leq 100$ is generated. If $rand < \rho(t)$, then a *gene* $b_{uv_0}^k$ is selected and mutated.

Algorithm 2 Mutation procedure

$\rho(t) \leftarrow \rho(t-1) - \beta \times 10$
$rand = generateRandom(min = 0, max = 100)$
if $rand < \rho(t)$ **then**
  $v_0 = generateRandom(min = 1, max = N_k)$
  $b_{uv_0}^k = \overline{b_{uv_0}^k}$
end if

IV. Experimental results

In this section, we present our preliminary experimental results. We aimed to evaluate the approach's performance when spotting handwritten queries in document images with challenging quality, and the effectiveness of the GA in optimizing the spotting.



(a) Query 1 (b) Query 2 (c) Query 3

Fig. 7. Thinned queries used in the experiment.

TABLE I

VALUES OF *Precision* AND *Recall* CORRESPONDING TO 3 QUERIES, PRIOR TO USING GA OPTIMIZATION.

Query	Precision	Recall
Query 1	62.9%	63.8%
Query 2	63.9%	83.94%
Query 3	67.8%	73.54%

Evaluation procedure

We prepared an image dataset by converting pages of the journal *Annales de l'insée (Numéro 40, Oct-Dec 1980)* [45] into document images in 200×200 dpi. The dataset contains 104 images that include text and mathematical calculations. The image resolution was 1110×1162.

Throughout the experiment, the parameters were set empirically as follows: The similarity threshold $\alpha = 0.35$, the dissimilarity threshold $\theta = 0.03$, and the voting region size parameter $\delta = 0.35$. The size of the GA population is fixed according to the number of components in the *voting group* G_k as follows: $||P^k|| = 3 \times N_k$, and the mutation probability parameter $\beta = 1$.

The evaluation was done using 3 queries that were scanned in 300×300 dpi and thinned using ATF (Fig. 7). Statistical analysis of the results was done by calculating *Precision* and *Recall* as follows:

$$Precision = \frac{Number\ of\ Relevant\ Pixels \times 100}{Number\ of\ Retrieved\ Pixels} \quad (6)$$

$$Recall = \frac{Number\ of\ Relevant\ Pixels \times 100}{Number\ of\ Total\ Relevant\ Pixels} \quad (7)$$

where $Number\ of\ Total\ Relevant\ Pixels$ is known from the ground truth, $Number\ of\ Relevant\ Pixels$ and $Number\ of\ Retrieved\ Pixels$ are calculated after the spotting.

Precision expresses the ability of the approach to find relevant occurrences, while *Recall* expresses the ability to find all correct results.

Results and discussion

Table I shows the values of *Precision* and *Recall* for the 3 queries in Fig. 7, prior to using GA optimization. The low values of *Precision* are the caused by *residuals*. The values of *Recall* are affected by the complexity of the query.

Fig. 8 shows the effect of using the GA optimization on the values of *Precision* and *Recall* calculated for the 3 queries. From the early iterations, *Precision* improved significantly and *Recall* decreased then started to improve again. The

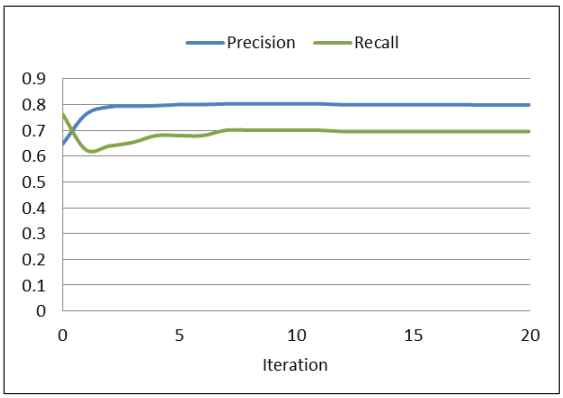


Fig. 8. Curves of *Precision* and *Recall* per algorithm iteration.

algorithm reached an optimal stage after 8 iterations, where $Precision = 80.4\%$ and $Recall = 70.0\%$.

The pruning step using the query auto-correlation matrix removes around 50% of the document image content in average. Instead of direct comparison between components, the advantage of this pruning method lies in comparing between patterns of similarity and dissimilarity between a document image component and a set of query components. By doing so, it attenuates the effect of handwriting and standard font variations.

The preliminary results indicate promising performances of the modular spotting approach and effectiveness of using GA to optimize the algorithm precision. The modular spotting approach is effective in removing most of irrelevant patterns from the document image. Then, the GA improves significantly the algorithm precision without compromising its recall. Fig. 9 illustrates the results of steps of our approach on an image from the dataset.

Our preliminary results are based on the choice of parameters α, θ, and δ. Currently, the parameters are set empirically. In our future work, we aim to make the parameters' setting automatic and adaptive to the query and document image.

V. CONCLUSION AND FUTURE WORK

We reported our ongoing research on a modular approach for spotting queries in document images, and its optimization using GA. The modular approach finds candidate occurrences of a query in a document image by removing irrelevant pixels using normalization, feature matching, pruning, and voting. Then, optimization of the spotting result is done using GA in order to remove *residuals*.

Preliminary experimental results show promising performances and possibility of further improvement. Comparing to existing methods, the proposed approach is application-independent and segmentation-free. In addition, it involves EA in spotting which is, to the best of our knowledge, a first attempt in this issue.

Our next direction is to make the algorithm's parameters adjusting automatic and adaptive to the query and document image. We also intend to carry large scale experiments in order to evaluate the GA effectiveness and the algorithm's performance compared with other methods.

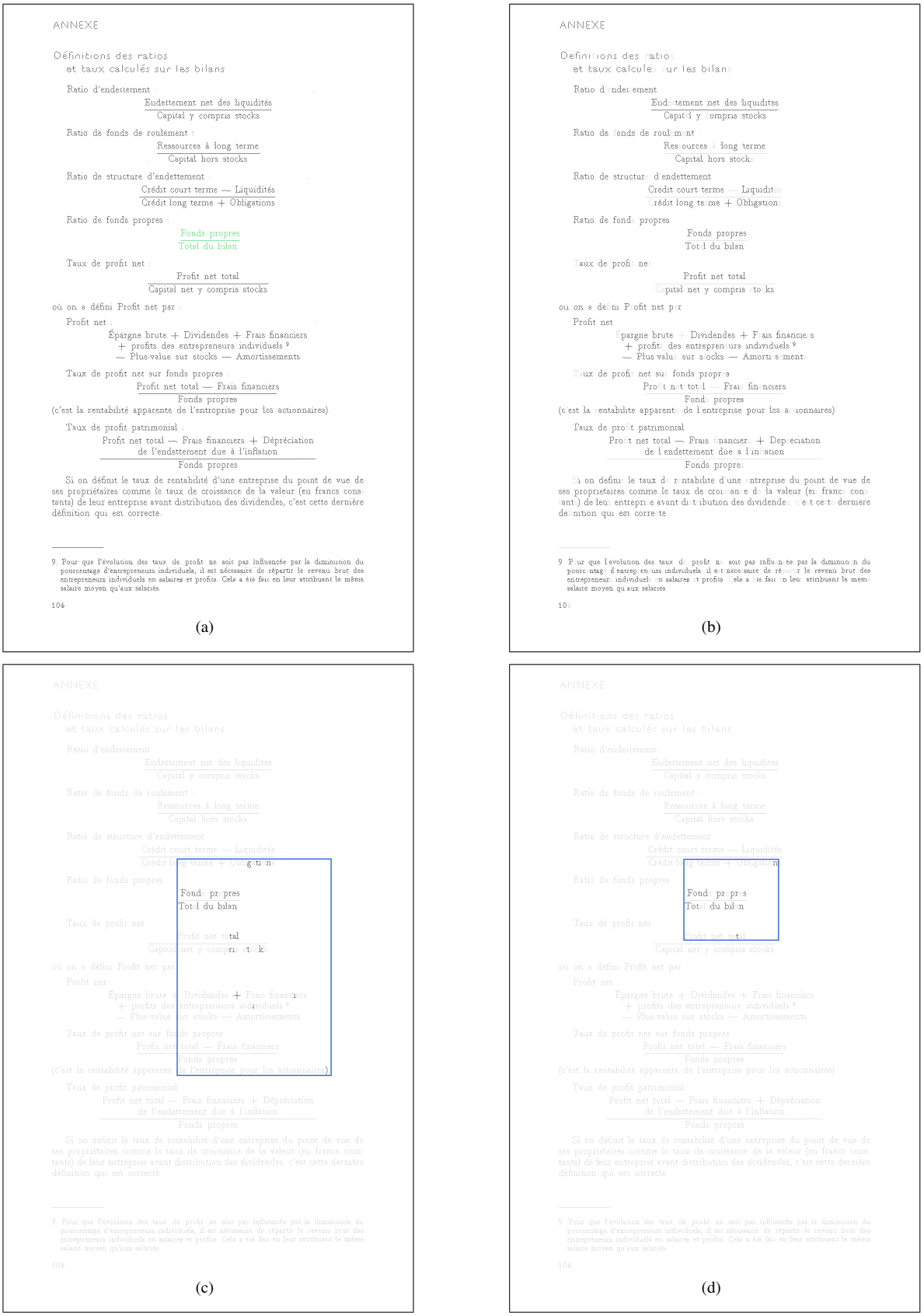


Fig. 9. Illustration of steps of our spotting approach (suppressed pixels from the original image are kept in bright gray color for convenience of illustration): (a) Original document image. The query occurrence is highlighted in green. (b) Image after pruning. Around 15% of the pixels have been pruned. (c) Result of preliminary spotting illustrated by a blue box. (d) Result of spotting optimization using GA illustrated by a blue box. 86.7% of the residuals have been removed.

REFERENCES

[1] H. Chatbri, K. Kameyama, and P. Kwan, "An application-independent and segmentation-free approach for spotting queries in document images," in *International Conference on Pattern Recognition (ICPR'14) (to appear)*. IEEE, 2014.

[2] K. Kameyama, N. Oka, and K. Toraichi, "Optimal parameter selection in image similarity evaluation algorithms using particle swarm optimization," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2006, pp. 1079–1086.

[3] M. Okayama, N. Oka, and K. Kameyama, "Relevance optimization in image database using feature space preference mapping and particle swarm optimization," in *Neural Information Processing*. Springer, 2008, pp. 608–617.

[4] A. W. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 12, pp. 1349–1380, 2000.

[5] H. Müller, N. Michoux, D. Bandon, and A. Geissbuhler, "A review of content-based image retrieval systems in medical applicationsclinical benefits and future directions," *International journal of medical informatics*, vol. 73, no. 1, pp. 1–23, 2004.

[6] A. Gharbi and K. Marzouki, "A novel approach of content based medical images indexing system based on spatial distribution of vector descriptors," in *International Multi-Conference on Systems, Signals & Devices (SSD)*. IEEE, 2013, pp. 1–4.

[7] S. Berrani, L. Amsaleg, and P. Gros, "Robust content-based image searches for copyright protection," in *Proceedings of the 1st ACM international workshop on Multimedia databases*. ACM, 2003.

[8] K. Kameyama, S.-N. Kim, M. Suzuki, K. Toraichi, and T. Yamamoto, "Content-based image retrieval of kaou images by relaxation matching of region features," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 14, no. 04, pp. 509–523, 2006.

[9] P. W. Kwan, K. Kameyama, J. Gao, and K. Toraichi, "Content-based image retrieval of cultural heritage symbols by interaction of visual perspective," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 25, no. 05, pp. 643–673, 2011.

[10] P. W. Kwan, K. Kameyama, and K. Toraichi, "Trademark retrieval by relaxation matching on fluency function approximated image contours," in *IEEE Pacific Rim Conference on Communications, Computers and signal Processing (PACRIM)*, vol. 1. IEEE, 2001.

[11] P. W. Kwan, K. Kameyama, and K. Toraichi, "On a relaxation-labeling algorithm for real-time contour-based image similarity retrieval," *Image and Vision Computing*, vol. 21, no. 3, pp. 285–294, 2003.

[12] H. Chatbri, K. Kameyama, and P. 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.

[13] H. Chatbri and K. Kameyama, "Sketch-based image retrieval by shape points description in support regions," in *International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2013, pp. 19–22.

[14] A. Murugappan, B. Ramachandran, and P. Dhavachelvan, "A survey of keyword spotting techniques for printed document images," *Artificial Intelligence Review*, vol. 35, no. 2, pp. 119–136, 2011.

[15] J. H. Holland, *Adaptation in Natural and Artificial Systems*. MIT Press, 1992.

[16] K. Kameyama, "Particle swarm optimization-a survey," *IEICE transactions on information and systems*, vol. 92, no. 7, 2009.

[17] Y. Liu, D. Zhang, G. Lu, and W.-Y. Ma, "A survey of content-based image retrieval with high-level semantics," *Pattern Recognition*, vol. 40, no. 1, pp. 262–282, 2007.

[18] R. Manmatha, C. Han, and E. M. Riseman, "Word spotting: A new approach to indexing handwriting," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'96)*. IEEE, 1996, pp. 631–637.

[19] M. Bokser, "Omnidocument technologies," *Proceedings of the IEEE*, vol. 80, no. 7, pp. 1066–1078, 1992.

[20] A. Kae and E. G. Learned-Miller, "Learning on the fly: Font free approaches to difficult ocr problems," in *International Conference on Document Analysis and Recognition (ICDAR'09)*, 2009.

[21] K. Tamura, T. Yoshikawa, and T. Furuhashi, "A study on document retrieval system based on visualization to manage ocr documents," in *Human-Computer Interaction. Interaction Modalities and Techniques*. Springer, 2013, pp. 740–749.

[22] D.-R. Lee, W. Hong, and I.-S. Oh, "Segmentation-free word spotting using SIFT," in *IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*. IEEE, 2012, pp. 65–68.

[23] M. I. Shah and C. Y. Suen, "Word spotting techniques in document analysis and retrieval-a comprehensive survey," *Handbook of Pattern Recognition and Computer Vision*, vol. 4, pp. 353–376, 2010.

[24] 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'11)*, 2011.

[25] Y. Lu and C. L. Tan, "Word spotting in chinese document images without layout analysis," in *International Conference on Pattern Recognition (ICPR'02)*, vol. 3. IEEE, 2002, pp. 57–60.

[26] S. Bai, L. Li, and C. L. Tan, "Keyword spotting in document images through word shape coding," in *International Conference on Document Analysis and Recognition (ICDAR'09)*. IEEE, 2009, pp. 331–335.

[27] R. d. S. Torres, A. X. Falcão, M. A. Gonçalves, J. P. Papa, B. Zhang, W. Fan, and E. A. Fox, "A genetic programming framework for content-based image retrieval," *Pattern Recognition*, vol. 42, no. 2, pp. 283–292, 2009.

[28] S. Kiranyaz, J. Pulkkinen, T. Ince, and M. Gabbouj, "Multi-dimensional evolutionary feature synthesis for content-based image retrieval," in *International Conference on Image Processing (ICIP)*. IEEE, 2011, pp. 3645–3648.

[29] S. M. Jadhav and V. Patil, "An effective content based image retrieval (CBIR) system based on evolutionary programming (EP)," in *Advanced Communication Control and Computing Technologies (ICACCCT), 2012 IEEE International Conference on*. IEEE, 2012, pp. 310–315.

[30] K. Kameyama, K. Toraichi, and Y. Kosugi, "Constructive relaxation matching involving dynamical model switching and its application to shape matching," *International Journal of Image and Graphics*, vol. 2, no. 04, pp. 655–667, 2002.

[31] M. Saadatmand-Tarzjan and H. A. Moghaddam, "A novel evolutionary approach for optimizing content-based image indexing algorithms," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 37, no. 1, pp. 139–153, 2007.

[32] C. G. Johnson, "Search-based evolutionary operators for extensionally-defined search spaces: Applications to image search," in *IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2012, pp. 1–7.

[33] T. M. Rath and R. Manmatha, "Word spotting for historical documents," *International Journal of Document Analysis and Recognition (IJDAR)*, vol. 9, no. 2-4, pp. 139–152, 2007.

[34] D. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in *KDD workshop*, vol. 10, no. 16, 1994, pp. 359–370.

[35] H. R. M. Ha, Jaekyu and I. T. Phillips, "Recursive XY cut using bounding boxes of connected components," in *International Conference on Document Analysis and Recognition (ICDAR'95)*, vol. 2, 1995.

[36] T. Sari, A. Kefali *et al.*, "A search engine for arabic documents," in *Dixième Colloque International Francophone sur l'Ecrit et le Document*, 2008, pp. 97–102.

[37] H. Chatbri and K. Kameyama, "Towards making thinning algorithms robust against noise in sketch images," in *International Conference on Pattern Recognition (ICPR'12)*. IEEE, 2012, pp. 3030–3033.

[38] H. Chatbri and K. Kameyama, "Using scale space filtering to make thinning algorithms robust against noise in sketch images," *Pattern Recognition Letters*, vol. 42, no. 0, pp. 1–10, 2014.

[39] X. Shu and X.-J. Wu, "A novel contour descriptor for 2D shape matching and its application to image retrieval," *Image and vision Computing*, vol. 29, no. 4, pp. 286–294, 2011.

[40] G. Borgefors, "Applications using distance transforms," *Aspects of Visual Form Processing*, pp. 83–108, 1994.

[41] A. P. Engelbrecht, *Computational intelligence: an introduction*. John Wiley & Sons, 2007.

[42] L. Davis, "Hybridization and numerical representation," *The Handbook of Genetic Algorithms*, pp. 61–71, 1991.

[43] L. Cui, J. Poon, S. Poon, K. Fan, H. Chen, P. Kwan, J. Gao, and Z. Ling, "Parallel model of independent component analysis constrained by reference curves for HPLC-DAD and its solution by multi-areas genetic algorithm," in *International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE, 2013, pp. 27–28.

[44] J. R. Koza, *Genetic Programming: vol. 1, On the programming of computers by means of natural selection*. MIT press, 1992, vol. 1.

[45] Annales de l'insée (numéro 40, Oct-Dec 1980). [Online]. Available: http://www.jstor.org (Accessed: December 2013)