Document Image Retrieval Based on Density Distribution Feature and Key Block Feature

Hong Liu [a], Suoqian Feng [a], Hongbin Zha [a], Xueping Liu [b]
[a] National Lab on Machine Perception, Peking University, China
[b] Ricoh Co., Japan
{liuhong, sqFeng, zha}@cis.pku.edu.cn

Abstract

Document image retrieval is an important part of many document image processing systems such as paperless office systems, digital libraries and so on. Its task is to help users find out the most similar document images from a document image database. For developing a system of document image retrieval among different resolutions, different formats document images with hybrid characters of multiple languages, a new retrieval method based on document image density distribution features and key block features is proposed in this paper. Firstly, the density distribution and key block features of a document image are defined and extracted based on documents' print-core. Secondly, the candidate document images are attained based on the density distribution features. Thirdly, to improve reliability of the retrieval results, a confirmation procedure using key block features is applied to those candidates. Experimental results on a large scale document image database, which contains 10385 document images, show that the proposed method is efficient and robust to retrieve different kinds of document images in real time.

1. Introduction

Document image retrieval is an important part of the high performance document image processing systems, which are generally demanded in digital libraries, office automation, etc. Document image matching is the kernel technology in document image retrieval systems, whose aim is to attain the most similar document images in a document database for any input document image. In recent years, a number of document image retrieval approaches have been proposed. They can be included in two kinds of methods based on the features they used. The first class is retrieving document image based on document concrete content implemented by using OCR technology [1], [2]. But high time-consuming and the requirement of different OCR systems to deal with different languages characters are their drawbacks. The other is retrieving image based on image level information directly, and many approaches have been proposed.

P. Herrmann et al. proposed a document image retrieval method based on document page layout features [3]. D. Doermann et al. proposed a duplicate image detection algorithm based on extraction of a signature from a representative line of text in a document image by using character shape coding technique [4]. C. L. Tan et al. proposed a document image retrieval method based on horizontal traverse density and vertical traverse density of the character objects [5]. H. Peng et al. proposed a document image matching method based on sizes and positions of component block list in document image [6]. C. Wang et al. proposed a document image retrieval method based on the proportion of the black pixel area in character bounding box areas [7].

Parts of the above methods are based on the character feature or other local features in the document image. Because of distortion, noises, and low scan quality of the document images, it is generally quite difficult to extract these local features accurately. The other methods focus on page layout information or other global features. Obviously, successful document image retrieval algorithm should combine both local features and global features to achieve a more outstanding performance. For retrieving document image among different resolutions, different formats and hybrid characters document images in a large scale document image database quickly, we proposed a retrieval method based on density distribution feature as document image local feature, and key block feature as global feature.

This paper is organized as follows: In section 2, extraction methods of the features for document image retrieval are explained. In section 3, a document image retrieval method based on density distribution features and key block features is proposed. In section 4, performances of our proposed method is presented and experimental results with a large scale document image database are analyzed. Finally, conclusions are drawn in section 5.

Proceedings of the 2005 Eight International Conference on Document Analysis and Recognition (ICDAR'05)
1520-5263/05 $20.00 © 2005 IEEE

2. Document Image Features Extraction

How to describe a document image is quite important, which largely affects performances of a retrieval algorithm. As far as it goes, there exist a lot of methods to define and extract document image features. Characters, line segments, text blocks, tables, images and some more abstract features such as projection graphs are commonly used. Here, we use the foreground pixel distribution feature and the relative size and position among the key blocks, which are the most outstanding regions in the document image, to describe a document image. These two kinds of features are both robust to image degradation. What is more, the time requirement of the former is quite low and the latter is robust to image distortion.

2.1 Preprocessing

Because raw document images may be of different formats, contain unknown skew angles and scanned noises, it should be firstly binarized, deskewed and removed unknown noises.

In our system, we retrieve document image based on its foreground pixel distribution features, therefore binarization is an important step. It affects the performance of the system remarkably. Here, a method combining Otsu Algorithm [8] with region-based method is used to extract the foreground of a document image from the background. Skew will cause serious problems in following stages such as page layout analysis and so on [9]. Here, we detect the skew angle by selecting the longest borderlines of the foreground in an image and using a least square method to calculate the skew angle. Last, a 3×3 template is used to remove noises.

2.2 Feature Extraction

2.2.1 Print-core localization

Print-core is the region where we extract the document image features, and provide a uniform reference frame for different kinds of images. Here, The print-core of a document image is located as follows:

1) A raw print-core is firstly extracted based on the projection profile of a document image. At the same time, the main direction of the document and the minimal space between lines is attained.

2) The document image is processed with the RLSA (Run Length Smoothing Algorithm) to enlarge the differences between the image foreground and noises.

3) A print-core is located based on the projection profile of the document image and the size restriction. The connected block at the boundary of the raw print-core will be ignored, when its width or height is smaller than the thresholds.

2.2.2 Density Distribution Feature

Density distribution feature is defined as a matrix whose components are referred as the relative density of the foreground pixel in each small region produced by separating the print-core of the document image equally. The following steps are included to get density distribution features automatically.

1) Region division: The print-core region of a document image is separated into $m\times n$ small regions equally along the horizontal and vertical directions as Fig. 1.

2) Feature computation: The total number s_i (i=0, ..., $m\times n-1$) of the foreground pixels in each small region is computed.

3) Feature normalization: The features of each small region are normalized as follows:

a) Finding out the maximum S,

$$S = \max_{i=0,\ldots,m\times n-1}(s_i) \tag{1}$$

b) Computing the ratio r_i,

$$r_i = s_i / s \qquad i = 0, \ldots, m\times n-1 \tag{2}$$

c) Computing the difference dr_i among r_i of connected small regions according to the main direction of the document.

If the direction of the document is horizontal,

$$dr_{j\times n+l} = \begin{cases} r_{j\times n+l} - r_{j\times n+l-1} & ,l = n-1 \\ r_{j\times n+l} - r_{j\times n+l+1} & , \ l = 0 \\ 2\times r_{j\times n+l} - r_{j\times n+l-1} - r_{j\times n+l+1} & , others \end{cases} \tag{3}$$

If the direction of the document is vertical,

$$dr_{j\times n+l} = \begin{cases} r_{j\times n+l} - r_{(j-1)\times n+l} & ,j = m-1 \\ r_{j\times n+l} - r_{(j+1)\times n+l} & , \ j = 0 \\ 2\times r_{j\times n+l} - r_{(j-1)\times n+l} - r_{(j+1)\times n+l} & , \ others \end{cases} \tag{4}$$

d) Organizing the density distribution feature *DDF* as follows:

$$DDF(r_0, \ldots, r_{m\times n-1}; dr_0, \ldots, dr_{m\times n-1})$$

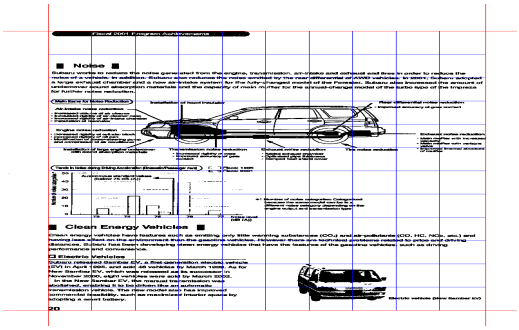

Fig. 1. Separating an image's print-core

Proceedings of the 2005 Eight International Conference on Document Analysis and Recognition (ICDAR'05)
1520-5263/05 $20.00 © 2005 IEEE

2.2.3 Key Block Feature

Key block feature is defined as a vector whose components are referred as the relative size and position of Top 3 key blocks of the homogeneous region of a document image. The following steps are included to produce a series of key blocks of a document image automatically.

1) Processing a document image with the RLSA (Run Length Smoothing Algorithm) to enhance differences between foreground and background of the image;

2) Estimating the periodicity of each region along the horizontal and vertical directions of the interested region of the document image based on the differences between the widths of the border upon lines or the border upon line spaces;

3) If the region is not periodical, find the position for splitting where the difference is the largest;

4) Repeating step 2) and step 3) until the region is homogeneous;

5) Choosing the Top 3 key blocks based on the average width of all the lines in the homogeneous region. c_i represents the centroid of the i^{th} key block, and bw_i represents the average width of all the lines in the i^{th} key block;

6) Organizing the features and relative positions of each key block into a vector *KBF* according to the order of the average width of all the lines in the key block:

$$l_1 = \|c_1 - c_2\|, \quad l_2 = \|c_2 - c_3\|, \quad l_3 = \|c_3 - c_1\| \tag{5}$$

$$l_0 = l_3 \tag{6}$$

$$L = \max(l_1,\ l_2,\ l_3) \tag{7}$$

$$kb_i = bw_i / L, \qquad i=1, \ldots, 3 \tag{8}$$

$$tl = 0.5 \times (l_1 + l_2 + l_3) \tag{9}$$

$$A = ((tl-l_1)\times(tl-l_2)\times(tl-l_3)\times tl)^{1/2} \tag{10}$$

$$\alpha_i = 2 \times A / l_i / l_{i-1} \tag{11}$$

then KBF is organized as:

$$KBF(kb_1,\ kb_2,\ kb_3,\ \alpha_1,\ \alpha_2,\ \alpha_3)$$

Where α_i represents sin(θ_i) and θ_i is the angle at the center of the i^{th} key block illustrated in Fig 2.

Based on above definition, a document image is represented by density distribution features and key block features. The advantages of density distribution feature are its simple, efficient extraction and low matching computing consumption. However, its performance largely depends on the accuracy of the print-core location. Hence, other features are needed to serve as the supplementary features. Key block feature is an reasonable option. It represents the relative size and position of the Top 3 key blocks whose centroids compose the vertex of a triangle. The triangle is robust to image translation, image scaling with factor s_x equaling to factor s_y, image rotation and the print-core location.

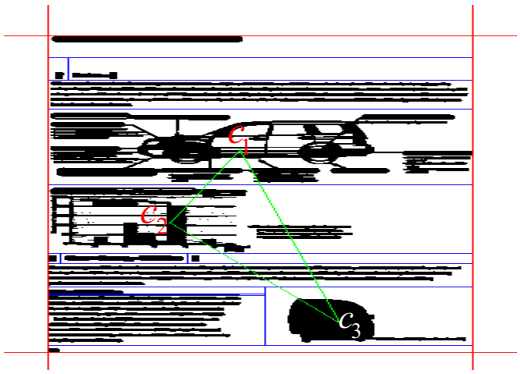

Fig. 2. The relative angle between the key block

3. Document Image Retrieval Algorithm

The density distribution feature of the k^{th} document image in a database is noted as DDF(k), and the key block feature of the k^{th} document image in a database as KBF(k). Given a candidate document image database, the problem of document image retrieval equals to finding out the most similar images. Here, we propose a method which gets the Top 10 candidate target images with density distribution features firstly and then the targeted images are confirmed with key block features.

3.1 DDF Matching Algorithm

This algorithm can be described with the following pseudocode:

Procedure DDF matching algorithm;
{Input: DDF(q) for a input document image, a handle to a image database of DDF }
{Output: the Top 10 most similar targeted image with the minimum distance $Ddis_k$ & $Ddif_k$ to the Input image}
begin
  for $k=1,\ldots,K$, **do begin** compute $Ddis_k$ & $Ddif_k$, which are the distances between *DDF(q)* and the k^{th} DDF(k);
  end;
  select the Top 10 most similar targeted image based on the distance $Ddis_k$ & $Ddif_k$
end

The distance $Ddis_k$ is computed as follow:

1) Computing the square of the distance between part of the corresponding component of the DDF(q) and DDF(k):

$$ds_i^k = (r_i^{ddf(q)} - r_i^{ddf(k)})^2 \quad (12)$$

2) Sorting the ds_i^k into ds_j^k according to the order:

$$ds_0^k < ... < ds_j^k < ds_{j+1}^k < ... < ds_{m\times n-1}^k$$

3) To tolerate a bit false of binarization and improve the robustness of our method, $Ddis_k$ is computed as follow:

$$Ddis_k = \sum_j ds_j^k, \quad j < m \times n - 1 \quad (13)$$

The distance $Ddif_k$ is computed as follow:

1) Computing the square of the distance between part of the corresponding component of the DDF(q) and DDF(k),

$$dd_i^k = (dr_i^{ddf(q)} - dr_i^{ddf(k)})^2 \quad (14)$$

2) Computing $Ddif_k$,

$$Ddif_k = \sum_{i=0,...,m\times n-1} dd_i^k \quad (15)$$

The Top 10 most similar targeted images are selected as follows:

1) Selecting the Top 5 most similar target images cti_i^{dis} based on the distance $Ddis_k$.

2) Selecting the Top 5 most similar target images cti_i^{dif} based on the distance $Ddif_k$.

3) Assembling the Top 10 candidate images as follows:

$$(cti_0^{dis}, ..., cti_4^{dis}, cti_0^{dif}, ..., cti_4^{dif})$$

3.2 KBF Confirming Algorithm

The Top 10 candidate images are obtained by the DDF matching algorithm. However, To tolerate a bit false of binarization and improve the robustness of our matching method, we admit some larger different parts among the inputted document image and images in the database. That will lead to retrieval failure, Sometimes. We can not be confident with the results only based on density distribution features. At the same time, density distribution features of those images only full of printed characters are similar. Hence, other features are needed to distinguish those images. Therefore, we confirm the candidate images by key block features. The algorithm can be described with the following pseudocode:

Procedure KBF confirming algorithm;
{Input: KBF(q) for a input document image, a handle to the Top 10 candidate target images of KBF }
{Output: the Top 5 most similar target images}
begin
for *i=0, ..., 9*, **do begin** compute the reliability $Rkbf_i$, which are the distances between KBF(q) and the i^{th} candidate image's KBF(*i*);
end
select the Top 5 most similar image for the inputted image;
end

The reliability $Rkbf_i$ is computed as follow:

1) Computing the reliability between the input image and the candidate output image with the key block foreground average size .

$$sc_i = \sum_{j=1}^{3} [sw_j \times (kb_j^{kbf(q)} - kb_j^{kbf(i)})^2] \quad (16)$$

2) Computing the reliability between the input image and the candidate output image with the relative position between the key blocks.

$$pc_i = \sum_{j=1}^{3} [pw_j \times (\alpha_j^{kbf(q)} - \alpha_j^{kbf(i)})^2] \quad (17)$$

3) Computing reliability $Rkbf_i$:

$$Rkbf_i = 1/(1 + sc_i \times pc_i) \quad (18)$$

Output images selecting can be described with the following pseudocode*:*

Begin
for *i = 0, ..., 9*, **do begin** sort the cti_i^{dif} and cti_i^{dis} based on the $Rkbf_i$;
end
select the Top 5 most similar target images based on the order based on the $Rkbf_i$;
end

4. Experiment Results and Analysis

4.1 Database

For developing a document image retrieval system, a large-scale document image database is established, which contains 10385 document images with different resolutions (100 dpi, 200 dpi, 400 dpi.), different formats (binary, gray, color), and hybrid different languages (Chinese, Japanese, English) characters. There are 3668 images containing only printed characters, 2910 images containing printed characters and pictures, 2153 images containing printed characters and tables and 1654 images containing printed characters, tables and pictures. The features of all the document images in the database are extracted and sorted in the database as density distribution features and key block features.

4.2 Performances

We design the following experiments to test our proposed method with 10385 different document images with different skew angles and different resolutions scanned from three scanners. Experiment 1 is to retrieve the document images by density distribution features only.

Results of the experiment 1 is showed in Table 1. Experiment 2 is to retrieve the document images by key block features. The results of the experiment 2 is demonstrated in Table 2. Experiment 3 is to retrieve the candidate documents images with density distribution features, and then the results are confirmed by key block features. The results is demonstrated in Table 3.

From Table 1 and Table 2, we can find out that the accuracy of Top 1 is about 90.1% and 21.8% when the image is retrieved only based on the density distribution features or key block features respectively. The main reason for the poor performance of experiment 2 is that many images in the database have almost the same key block features. Therefore, the key block features are weak to retrieve document image itself. The accuracy of Top 1 is improved to 92.3% while we combine the density distribution features with key block features.

Table 1. Retrieval results: query 10385 images based on density distribution features

	Top 1	Top 2	Top 3	Top 4	Top 5
Number	9360	9600	9667	9697	9719
Proportion	90.1	92.4	93.1	93.4	93.6

Table 2. Retrieval results: query 10385 images based on key block features

	Top 1	Top 2	Top 3	Top 4	Top 5
Number	2261	2845	3189	3467	3677
Proportion	21.8	27.4	30.7	33.4	35.4

Table 3. Retrieval results: query 10385 images based on density distribution features and key block features

	Top 1	Top 2	Top 3	Top 4	Top 5
Number	9587	9827	9859	9878	9966
Proportion	92.3	94.6	94.9	95.1	96.0

Above experiments are implemented on a PC with Pentium 1.7 GHZ and 256 MB memory. The average time requirements are about 1.8s for image processing and 1.6s for querying and ranking retrieval results from the database containing 10385 document images. The retrieval time grows slightly with the size of the database while the image processing time remains constant.

4.3 Analysis

We can find that our method is efficient in retrieving different kinds of document images from a large scale database. It produces the coarse candidate images based on a kind of local features, density distribution features. Then a confirming procedure using the global feature, key block features, is applied to improve the performance of the system. It not only focus on the global information of document images, but also cares the local prominent features, which ensure the accuracy and reliability of the system. Nevertheless, the results of the binarization affect the performance of our system evidently.

Although our method is efficient for most document images, it is weak for those images which are very similar to each other in distribution and block features. Novels with full of the uniform character are the typical examples. A more detail local features should be imported to improve its performances.

5. Conclusions

In this paper, we presented a method of document image retrieval based on density distribution features and key block features of document images. Key block features are applied to confirm the reliability of the raw candidate images so as to improve the retrieval performance. Experiments show that our method is efficient for retrieving color and gray document images from a large-scale hybrid different languages document image database in real time.

Acknowledgement

This research was supported by the jointed project of 'Document Image Retrieval' between Ricoh Co., Japan, and Peking University, China.

References

[1] Y. He, Z. Jiang, B. Liu, H. Zhao, "Content-based indexing and retrieval method of chinese document images", In Proc. 5th ICDAR, pages 685 - 688, 1999.

[2] A. Takasu, "Document filtering for fast approximate string matching of erroneous text", In Proc. 6th ICDAR, pages 916 – 920, 2001.

[3] P. Herrmann, G. Schlageter, "Retrieval of document images using layout knowledge", In Proc. 2nd ICDAR, pages 537 – 540, 1993.

[4] D. Doermann, H. Li, O. Kia, "The detection of duplicates in document image databases", In Image and Vision Computing, pages 907 – 920, 1998.

[5] C. L. Tan, W. Huang, Z. Yu, Y. Xu, "Imaged document text retrieval without OCR". In IEEE Trans. PAMI, pages 838 – 844, 2002.

[6] H. Peng, F. Long, Z. Chi, W. Siu, "Document image template matching based on component block list", In Pattern Recognition letters, pages 1033 – 1042, 2001.

[7] C. Wang, T. Chen, Y. Chan, R. Hwang, W. Huang, "Chinese document image retrieval system based on proportion of black pixel area in a character image", In 6th ICACT, pages 25 – 29, 2004.

[8] N. Otsu, "A threshold selection method from gray-level histogram", In IEEE Trans. SMC-9(1), pages 62-66, 1979.

[9] M. Chen, X. Ding , "A robust skew detection algorithm for grayscale document image", In Proc. 5th ICDAR, pages 617 - 620, 1999.