

Keyword Spotting and Retrieval of Document Images Captured by a Digital Camera

Shijian Lu, Chew Lim Tan

Department of Computer Science, School of Computing
National University of Singapore, Kent Ridge, 117543, Singapore
{lusj, tancl@comp.nus.edu.sg}

Abstract

This paper presents a keyword spotting technique that locates keywords within document images captured by a digital camera. In the proposed technique, the shape of word images in perspective view is captured by using three perspective invariants, namely, holes, water reservoirs, and character ascenders and descenders. Given a camera image of document, text line and word images are first segmented through the connected component analysis. The three perspective invariants are then detected through two rounds of scanning process, which transliterate each character image into a character shape code of dimension six and so convert each word image into a word shape code. Keywords within camera images of documents are finally located through a partial matching process. Experiments show some promising results.

1. Introduction

As the sensor resolution rises, an increasing number of documents are being captured by using a digital camera. Compared with document scanners, digital cameras have a few advantages including the faster capturing process, the more capturing flexibility, and the better portability. However, camera images of documents often suffer from the perspective and geometric distortion due to the camera capturing mechanism and the document curvature as illustrated by the two camera image of documents shown in Figure 1. As a result, they cannot be recognized by most generic optical character recognition (OCR) systems properly. Under such circumstance, a perspective-tolerant keyword spotting technique will help for the location of word images and the pre-screening of document images.

Some works [1, 4] have been reported to locate keywords and retrieve document images directly without OCR. One approach is through the character shape coding [3, 5, 6, 8,

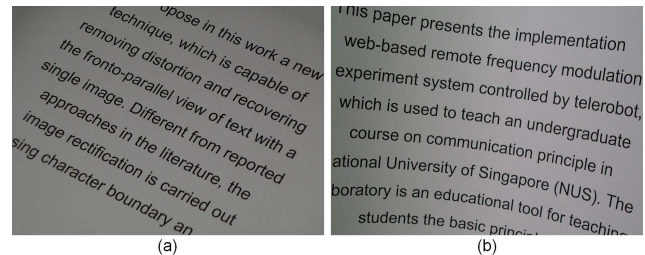


Figure 1. Two camera images of document with one lying over a planar surface in (a) and the other lying a curved surface in (b).

9, 10, 11], which converts imaged characters into a set of pre-defined codes. The generated character shape codes are then grouped into higher-level word shape tokens. Finally, keywords are located and document images are retrieved by using the produced word shape tokens. For example, Nakayama classifies characters into seven categories and then uses the converted word shape tokens for content word detection [5] and document image categorization [6]. Similarly, Spitz *et al.* take a character shape coding approach for keyword spotting [9], document image categorization [10], and document image retrieval [8]. In addition, Tan *et al.* also propose a few word shape coding schemes for keyword spotting [3] and document image retrieval [11].

Though several character shape coding schemes have been reported, few of them can deal with camera images of documents that are degraded by the perspective distortion. In this paper, we present a novel keyword spotting technique that is capable of locating keywords within document images captured by a digital camera. Given a camera image of document, vertical and horizontal text directions are first detected. The shape of word images is then captured by using three perspective invariants including holes, water reservoirs, and character ascenders and descenders.

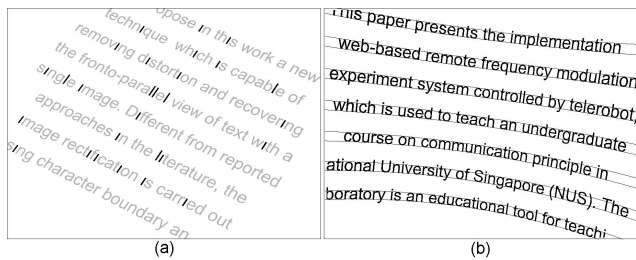


Figure 2. (a) The estimation of vertical text direction by using characters “i” and “l”; (b) the estimation of horizontal text direction by using the x line and base line of text.

The three invariant features transliterate each character into a digit sequence of dimension six and so convert each word image into a word shape code. Word images are accordingly located through a partial matching process.

2. Proposed Methods

This section presents the proposed keyword spotting and document image retrieval technique. In particular, we will divide this section into four subsections, which deal with the text direction detection, character shape analysis, keyword spotting, and discussions, respectively.

2.1 Text Direction Detection

Horizontal and vertical text directions are required for the analysis of the character shapes in perspective view. We determine vertical and horizontal text directions by using characters “i” and “l” and the x line and base line of text. Before that, a series of document preprocessing operations are required as detailed below.

Firstly, noise of small size such as salt & pepper is suppressed by using a mean filter. Filtered camera images of documents are then segmented from the background by using Sauvola’s adaptive thresholding technique [7]. After that, imaged characters are labeled through the connected component analysis. Lastly, character components of small size such as the top part of characters “i” and “j” are removed by an image filtering process as follows:

$$T = k \cdot Size_{mdn} \quad (1)$$

where $Size_{mdn}$ refers to the median size of the labeled character components. For Roman document image, parameter k lies between 0.2-0.4 in most cases. We set it at 0.3 in our implemented system.

We estimate the vertical text direction by using characters i (the top part has been filtered out) and l because both

characters normally indicate the vertical direction of document text in perspective view. In particular, we detect the two characters based on the character linearity that is evaluated as follows:

$$dist = \frac{1}{n} \sum_{i=1}^n d(p_i, l) \quad (2)$$

where n denotes the number of character pixels and l refers to the straight line determined through the least square fitting of character pixels. Function d gives the distance between the i^{th} character pixel p_i and the l . Obviously, the linearity of characters “i” and “l” evaluated in Equation (2) is much smaller than that of other characters. For the document in Figure 1(a), Figure 2(a) shows the detected characters “i” and “l” (highlighted by the black color).

Horizontal text direction is then determined by using the x -line and base line of text. For documents lying over a planar or smoothly curved surface, the shape of text lines can be modeled by using a cubic polynomial curve in most cases. We fit x lines and base lines through the classification of character extremum points, which refer to the highest and lowest character pixels lying around the x line and base line positions. We adopt the point tracing technique [2] to classify character extremum points to different text lines. The horizontal character direction can accordingly be determined as the tangent of the related x -line or base line at the character centroid position. The u axis and v axis in Figure 3 illustrate the horizontal and vertical text directions. For the document in Figure 1(b), Figure 2(b) shows the fitted x lines and base lines of text.

2.2 Character Shape Analysis

In the proposed technique, word images within camera documents are coded by using three perspective invariants, namely, character ascenders and descenders, holes, and water reservoirs. For camera images of documents, character ascenders and descenders remain invariant to the perspective distortion because they always lie above or below the x lines or base line of text in perspective view. They can be detected by using the extremum points described in the last subsection. In particular, the extremum points of characters with the ascender and descender always lie a bit higher or lower than the x line and base line of text.

The last two perspective invariants refer to the hole and water reservoir illustrated in Figure 3, which can be detected by using the white runs information. Scanning horizontally (or vertically) from left to right (or from top to bottom), a white run can be located by a beginning pixel BP and an ending pixel EP illustrated in Figure 4 where black pixels (0) refer to the foreground text and white pixels (1) denote the blank background. As Figure 4 shows, the beginning pixel of a white run can be typically detected at

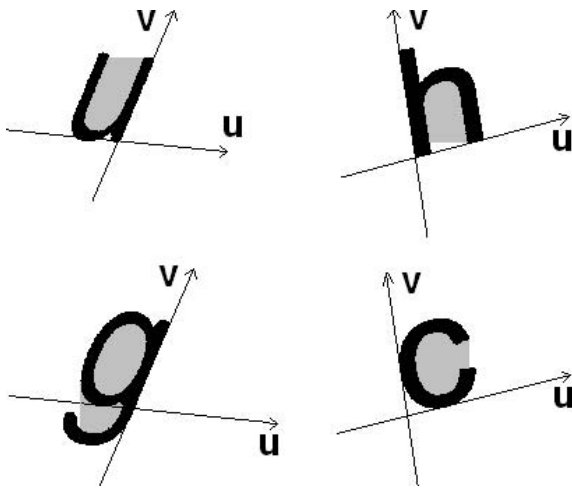


Figure 3. The hole feature in character “g” and the water reservoirs of characters (“g”, “c”, “u”, and “h”) facing left, right, up, and down, respectively.

the second pixel of the pixel sequence “01” and the ending pixel detected at the first pixel of the pixel sequence “10” instead. Each white run can accordingly be represented by its beginning pixel and ending pixel.

Character holes and character water reservoirs are detected through scanning the character horizontally row by row and vertically column by column. Take the horizontal scanning process as an example. Once a horizontal white run is detected, it will be set as the initial run if no white run in the last row is detected to be connected with it. On the other hand, the detected white run will be set as the last run if no white run in the next row is detected to be connected with it. In particular, the two white runs at two adjacent rows are determined as connected if they satisfy the following constraints:

$$BP_c < EP_l \ \& \ EP_c > BP_l \quad (3)$$

where BP_c and EP_c refer to the beginning and ending pixel of the white run detected in the current row. BP_l and EP_l instead denote the beginning and ending pixel of the white run detected in the last row.

After the scanning process described above, a character hole is detected if all pixels above the initial white run and those below the last white run of a set of connected white runs correspond to black text pixels illustrated by the character “g” in Figure 3. On the other hand, if only pixels above the initial white run or those below the last white run correspond to black text pixels, a downwards or an upwards water reservoir is detected, illustrated by characters “u” and “h” in Figure 3. Otherwise, a passage between adjacent charac-

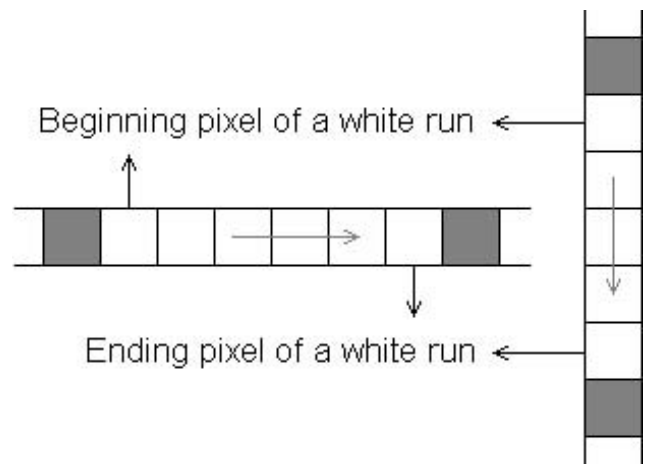


Figure 4. The beginning and ending pixel of a horizontal a vertical white run.

Table 1. Word shape codes of 26 lowercase Roman letters.

Characters	Codes	Characters	Codes
a	110010	bd	100001
c	001000	e	101100
ft	001001	g	11010-1
h	000011	ir	000000
j	00000-1	k	001111
l	000001	m	000020
n	000010	o	100000
pq	10000-1	s	011110
uv	000100	w	000210
x	011110	y	01010-1
z	011000		

ter is detected. Furthermore, while scanning the character component vertically, the water reservoirs facing left and right illustrated by characters “g” and “c” in Figure 3 can be detected in the similar way.

Each character image can thus be represented by using the three character shape features. We convert each character image into a character shape code of dimension six. In particular, the first code element records the number of holes within the character component under study. The second to the fifth elements record the numbers of water reservoirs facing left, right, up, and down, respectively. The last code element instead encodes character ascender and descender information, which is assigned with 1, -1, and 0, corresponding to the character that has an ascender, descender, and none of them, respectively.

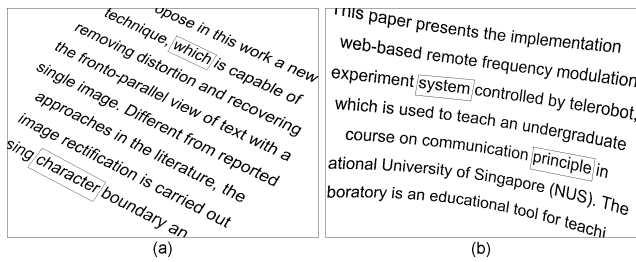


Figure 5. A few located word images within the two documents in Figure 1.

Table 1 shows the character shape code of the 26 lowercase Roman letters. As Table 1 shows, the three character shape features classify the 26 letters into 20 categories. A word image can thus be converted into a word shape code by concatenating the codes of the spelled characters from left to right. Take the most frequent word “the” as an example. Its word shape code becomes a digit sequence “001001000011101000” where the three subsequences “001001”, “000011”, and “101000” are converted from the three characters “t”, “h”, and “e”, respectively.

2.3 Keyword Spotting

Based on the proposed word shape coding scheme described in the last subsection, word images can be located from camera images of document through a word shape code matching process. Given a query word, the corresponding word shape code can be determined through the transliteration of the spelled characters according to the coding scheme in Table 1. The similarity between the query word shape code and word shape codes of word images within the studied document can accordingly be evaluated by using the the Hamming distance as follows:

$$sim(WSC_q, WSC_i) = 1 - H(WSC_q, WSC_i)/L \quad (4)$$

WSC_q refers to the query word shape code and L refers to the word shape code length. WSC_i refers to the code of the i^{th} word image of the same length as the query word. The function $H()$ gives the Hamming distance.

2.4 Discussions

Similar to most character shape coding based techniques [5, 6, 8, 9, 10], the proposed word shape coding scheme is sensitive to the character segmentation error. As described above, the proposed technique is character based, which convert each character into a digit sequence of dimension six. Therefore, the coding will be wrong if one or more spelled characters are touching or broken.

Table 2. The coding collision rates with relation to the word length (L: word length).

Coding scheme	L=6	L=7	L=8	L=9
Spitz’s	0.4822	0.4736	0.4621	0.4437
Tan’s	0.2918	0.2880	0.2674	0.2491
Ours	0.0621	0.0525	0.0413	0.0282

The sensitivity to the character segmentation error can be relieved according to the character shape codes in Table 1. For example, when the two characters “t” and “h” are touched at the base line position, the corresponding character shape code becomes [0 1 1 1 0 1] because there exist three water reservoir facing right, upwards, and downwards and a character ascender. Clearly, a character segmentation error is encountered because Roman letters in Table 1 have no such code pattern. Therefore, either the upward water reservoir or the downward should be removed. Under such circumstance, if we break the downwards water reservoir of character “h”, the component on the left will be coded as [0 0 1 1 0 1], which still has no match in Table 1. Therefore, the upward water reservoir is broken and the character shape code is divided into two, namely, [0 0 1 0 0 1] and [0 0 0 0 1 1], which are exactly the same as the codes of characters “t” and “h”.

3. Experiments

This section presents experimental results. The proposed word shape coding scheme is first evaluated by using a large number of character-coded words of different length. The keyword spotting performance is then tested by using 30 prepared documents captured by a digital camera.

3.1 Word Shape Coding Evaluation

This section presents the performance of the proposed word shape coding scheme. Besides, we also compare our coding scheme with the other two, namely, Spitz’s [9] and Tan’s [3]. In particular, we evaluate the coding performance by the collision rate, which is defined as the ratio between the number of words sharing word shape codes with others and the number of words coded. The three coding schemes are evaluated by every 500 words at each specific word length shown in Table 2. Experiments show that the collision rate of the proposed coding scheme is lower than 3%, which is much lower than Spitz’s and Tan’s. The lower collision rate can be explained by the better distinguishability of the proposed coding scheme, which classifies 26 Roman letters into 21 categories.

Table 3. Precision and recall of the proposed keyword spotting scheme.

<i>sim</i>	0.6	0.7	0.8	0.9	1
P%	0.3217	0.5386	0.7114	0.8953	0.9870
R%	0.9190	0.8435	0.7680	0.7215	0.5329
F%	0.4766	0.6574	0.7386	0.7991	0.6921

Besides, the coding scheme must be stable across the fonts and perspective distortion. We test the stability of the proposed coding scheme by using the 30 test documents where texts are printed in different fonts including Arial, Roman, Verdana, and so on. The 30 test documents are printed and then put to some planar and smoothing curved surfaces and captured by digital cameras from different viewpoints. Experiments shows that character ascenders and descenders and holes can be detected properly in most cases. However, character water reservoir in four directions may not be detected accurately due to the perspective distortion and the failure in the character segmentation.

3.2 Keyword Spotting Evaluation

The performance of the proposed keyword spotting technique is then evaluated by using the 30 camera image of document described in the last subsection. For each camera image of document, a few word image are picked and tested as queries. For the two documents in Figure 1, Figure 5 shows a few located word images. We evaluate the spotting performance by changing the code similarity in Equation (4). Besides the precision (P) and recall (R), we also use the composite measure F_1 defined as follows:

$$F_1 = \frac{2RP}{R + P} \quad (5)$$

The F_1 measure is derived from the recall and precision. It is a strict measurement because it does not only reflect the absolute value of the recall and precision but also reflect the degree of balance between the two.

It can be seen from Table 3 that the average recall and precision reach 75.70% and 69.08%. At the same time, it can be seen that setting a higher threshold gives a better precision but poorer recall. Therefore, if the emphasis is to locate word images accurately, then a threshold of 1 should be used. On the other hand, if the intent is to locate the keyword at least, then a threshold of 0.5 may be adopted. Furthermore, the parameter F_1 reaches the highest 79.91% for threshold of 0.9, which indicates that threshold 0.9 may be a good compromise.

4. Conclusion

This paper presents a keyword spotting technique that locates keywords within document images captured by a digital camera. The proposed technique captures word shapes by using three perspective invariants including holes, water reservoirs, and character ascenders and descenders, which converts each word image into a word shape code. Keywords are then located through a partial matching process. Experiments show that word images within camera images of document can be located properly.

5. Acknowledgement

This research is supported by the Agency for Science, Technology and Research (A*STAR), Singapore, under grant no. 0421010085.

References

- [1] D. Doermann. The indexing and retrieval of document images: A survey. *Computer Vision and Image Understanding*, 70(3):287–298, February 1998.
- [2] S. Lu, B. M. Chen, and C. C. Ko. A partition approach for the restoration of camera images of planar and curled document. *Image and Vision Computing*, pages 837–848, August 2006.
- [3] Y. Lu, L. Zhang, and C. L. Tan. A search engine for imaged documents in pdf files. In *27th Annual International ACM SIGIR Conference*, pages 536–537. ACM, July 2004.
- [4] M. Mitra and B. Chaudhuri. Information retrieval from documents: A survey. *Information Retrieval*, 2(2/3):141–163, August 2000.
- [5] T. Nakayama. Modeling content identification from document images. *Fourth conference on Applied natural language processing*, pages 22–27, October 1994.
- [6] T. Nakayama. Content-oriented categorization of document images. *International Conference On Computational Linguistics*, pages 818–823, August 1996.
- [7] J. Sauvola and M. Pietikainen. Adaptive document image binarization. *Pattern Recognition*, 33(2):225–236, January 2000.
- [8] A. F. Smeaton and A. L. Spitz. Using character shape coding for information retrieval. *4th International Conference on Document Analysis and Recognition*, pages 974–978, August 1997.
- [9] A. L. Spitz. Using character shape codes for word spotting in document images. *Shape, Structure and Pattern Recognition*, pages 382–389, 1995.
- [10] A. L. Spitz and A. Maghbouleh. Text categorization using character shape codes. *SPIE Symposium on Electronic Imaging Science and Technology*, pages 174–181, December 1999.
- [11] C. L. Tan, W. Huang, Z. Yu, and Y. Xu. Image document text retrieval without ocr. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 24(6):838–844, June 2002.