Mobile Retriever - Finding Document with a Snapshot

Xu Liu
Institute for Advanced Computer Studies
University of Maryland
liuxu@cs.umd.edu

David Doermann
Institute for Advanced Computer Studies
University of Maryland
doermann@umd.edu

Abstract

In this paper we describe a camera based document image retrieval system which is targeted toward camera phones. Our goal is to enable the device to identify which of a known set of documents it is "looking at". This paper provides two key contributions 1) a local context descriptor that effectively rules out irrelevant documents using only a small patch of the document image and 2) a layout verification approach that boosts the accuracy of retrieval even under severe degradation such as warping or crinkling. We have implemented the mobile retriever client on an iMate Jamin camera phone and tested it with a document database of 12742 pages. Experiments show that our verification approach clearly separates successful retrievals from unsuccessful retrievals.

1. Introduction

1.1. Motivation

The research in this paper is motivated by two facts. First, the trends toward a paperless office is leading to large quantities of documents existing in both electronic form (being born digital or being digitalized) and in hard copy (newspaper, magazine, etc.). Most of documents have digital version. Second, the number of camera phone users has increased tremendously in recent years. According to Garner's report[4] 48% of cellular phones had cameras in 2006, and is projected to increase to 81% by 2010. The camera phone is an ideal platform for content based retrieval systems [5] since it is easy to carry, it has the computational power of image processing and is linked to the wireless network. In this paper, we provide a way to enable camera phones (or other camera enabled devices) to serve as the input device for visually querying a document image database. More specifically, our document retrieval is based on a partial snapshot (Fig. 1 (b)) of the page from an unconstrained viewing angle (Fig. 1 (a)) , with the goal of finding

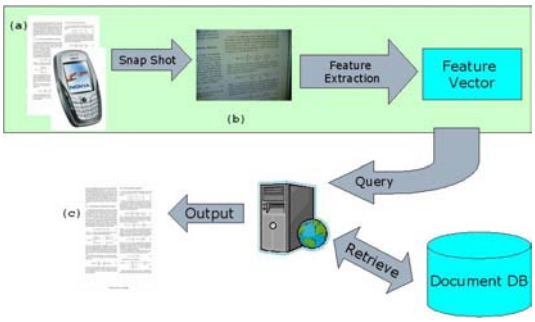

Figure 1. System

the original page (Fig. 1 (c)) in the database. Finding the page will enable many interesting applications for mobile access.

1.2. Use Scenario

Researchers often store a large number of digital documents on their computers. Often he has one printed paper and hopes to find the original PDF file, the simplest way is to identify the title of the paper, type it into a local/web search engine and look for the correct record from the results. On the desktop or laptop this approach is reasonable but for mobile devices it is cumbersome. Our approach simplifies this procedure into one snapshot. The original electronic document can be retrieved immediately when the camera phone "sees" the paper, even if the image quality is not good enough to "read" it.

Publishers will know what their reader reads if the reader is willing to take a snapshot of the article. The readers can comment, annotate and send feedback immediately.

The visually impaired can listen to the audio version of the article if its snapshot can be retrieved. The assumption is that the document is pre-stored in our database and the audio version is ready or can be synthesized using text to speech.

Watermarking usually requires special modification to document texts and can not be stably read by a camera. Our

approach can be used as a natural watermark for every document. It is unique and can be read by a camera phone.

1.3. Related work

Various approaches has been explored for image based document retrieval. In [6], Hull proposed a series of distortion-invariant descriptors allowing robust retrieval against re-formatting, re-imaging and geometric distortion. In [1], Cullen et al. use texture cues to retrieve documents from a database. In [8], Tan et al. measure document similarity by matching partial word images. In [7] Kameshiro et al. describe the use of an outline of the character shape, that tolerates recognition and segmentation errors for document image retrieval. Our approach is most closely related to the system proposed by Nakai and Kise et al. in [9] and [10]. In their approach, combinations of local invariants are hashed into a large hash table. Retrieval is accomplished by voting from this hash table and they are able to obtain an accuracy of 98% over 10000 documents. However the combinations of local invariants result in a very large feature vector. Furthermore, the query image must cover a large portion of the page which is sometimes hard to enforce especially with a camera phone. Camera phones are usually equipped with lower-end CMOS cameras; when capturing the whole page, the resolution might be too low to have the words separated. In our approach we have loosened the requirements of capturing, requiring only about 1/8 of a page. This makes the problem harder because the captured image could be from anywhere of the page, and a stable distribution of features cannot be expected because a large portion of the page may be absent.

Since we want our retrieval to be robust against perspective distortion, occlusion, uneven lighting, and even crinkled pages, we cannot use global configurations of feature points which might be partially missing or changed by degradations. Like Kise [9], we use local features which we call the "layout context". The rest part of this paper is organized as follows. A brief description of how we process the camera phone captured image in Section 2 is followed by a detailed explanation of layout context is contained in Section 3. After gathering a large number of layout contexts we cluster them to build a lexicon (Section 4) to index and re-rank the pages in database. A global layout verification step is introduced in Section 5, followed by experiments in Section 6. We discuss the shortcomings and future work in Section 7.

2. Image Processing

Before feature extraction, we need to separate the foreground contents from the background of the page, i.e. we binarize the image to identify the text. Camera phone images may be captured from various angles, and under various lighting conditions. Therefore, a global binarization will typically not work. We employ Niblack's[11] adaptive binarization method and then extract connected components from the image. A typical binarized image is shown in Fig. 2 and is sufficient for identifying words and components.

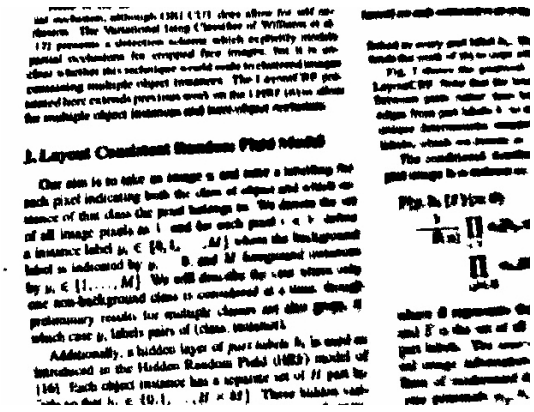

Figure 2. Adaptive binarized snapshot

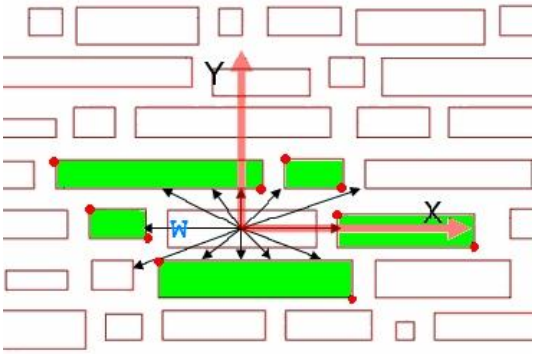

Figure 3. An Example of Layout Context

Although it will be very difficult, if not impossible, to extract stable and distinctive features from a single connected component (word), the relationships between components speak their own language - a language of contexts. Our goal is to extract the lexicon of this language,and index and retrieval documents using this lexicon. The first task is to define a "word" in this lexicon, i.e. the layout context.

3. Layout Context

We begin with an ideal image with no perspective distortion. In Fig. 3 each rectangle shows a bounding box of a word. To extract the layout context of a word w in Fig. 3, suppose we begin at the center of the word and look for the most visible n neighbors. Fig. 3 shows, for $n = 5$, using 5 green rectangles. The visibility is defined by the angle

of the view and the top n can be extracted using an efficient computational geometry algorithm with complexity linearly bounded by the total number of nearest m neighbors (it is safe to choose $m = 10n$). The top n visible neighbors are invariant to rotation and the percentage of view angles that a neighbor word occupies will not be effected by rotation. We put the coordinate system origin at the center of *w* with the X-axis parallel to the baseline of *w* and define the unit metric using the width of *w*. Under this coordinate system, the coordinates of the n most visible neighbors are invariant to similarity transformations.

- Translation: the original point always stays at the center of word *w*.
- Rotation: the X-axis always falls along the direction of the text lines.
- Scale: distortions are normalized by the width of word *w*. To use width of *w* as a stable factor of normalization,*w* must have an aspect ratio greater than a threshold (3 for example). This condition is satisfied by a large portion of words that have more than 3 characters.

With $n = 5$, a layout context is a vector that occupies $5 \times 2 \times 2 = 20$ bytes data (5 context words, 2 corners per word, 2 bytes per corner). When a document image is captured using a camera phone, it undergoes perspective transform, but locally can still be approximated by a similarity transform. For a similarity transform, scale, translation and rotation have to be normalized. We detect the baseline of text lines by finding the lower boundary of every connected component and the locally rotate text lines into the horizontal direction. After this, the scaling normalization is the same as for a perfect image.

4. Building the lexicon

As stated above, these low resolution documents speak a language of contexts. To understand this language, we must first build its lexicon, i.e. the dictionary of "visual words" [12] . We define the lexicon to be a set of representative layout contexts extracted from a training set of layout contexts. For example, we used 2000 pages randomly selected from the proceedings of CVPR04, 05, 06. From these 2000 pages we collect approximately 600 layout contexts from each page, for a total of 120548 layout contexts. For two reasons we cannot directly use these 120548 layout contexts as the lexicon. First, such a large lexicon will make the indexing procedure slow since we need to index each layout context by its nearest neighbors. Second, such a lexicon has layout contexts which are very similar; the nearest neighbor search could result in misclassification. In order to reduce the dimension, we run a mean-shift clustering on the layout contexts that results in a lexicon of containing 10042 clusters $<10\%$ of the original size.

5. Verification

The layout contexts extracted from the camera captured image may not be exactly the same as the one stored in the database for the following reasons:

- The word segmentation can be erroneous because of uneven lighting or inaccurate binarization, neighbor words could be touching and long words could be segmented. Segmentation inconsistency also occurs in the presence of non-text elements such as formulas and figures.
- On the boarder area of the camera captured image, the layout context may be different from the layout context stored in the database because some neighbor words are missing in the camera captured image.
- The document page might not be as perfectly flat as its digital version, warping and crinkling might destroy the planar invariants.

After sorting the documents by the coverage of layout contexts, therefore, a verification step is required and this step is our key contribution. To verify point set matches, RANSAC[3] is a classical model based algorithm. But RANSAC suffers most from non-rigid degradation since it is based on a model of transform, i.e. for plane-to-plane matching, this model is a projective transform (or homography). The assumption is that, all the inliers of matches must fit in this model exactly. But when a paper is warped or crinkled, which is very common, the model collapses since a homography can no longer be used to map from one point set to another. To allow a non-rigid transform, a more elastic method such as soft assign[2] might be used. However, soft assign is an iterative method which could take a long time to converge.

We propose a triplet based point matching algorithm which is robust against projective transforms, deformations and occlusions. Consider three points (A, B, C) on a 2D planar surface with homogeneous coordinates $(X_A, Y_A, 1)$, $(X_B, Y_B, 1)$ and $(X_C, Y_C, 1)$, their orientation is defined as

$$Sign(\begin{vmatrix} X_A & Y_A & 1 \\ X_B & Y_B & 1 \\ X_C & Y_C & 1 \end{vmatrix}) \quad (1)$$

where

$$Sign(X) = \begin{cases} 1 \cdots X \geq 0 \\ -1 \cdots X < 0 \end{cases}$$

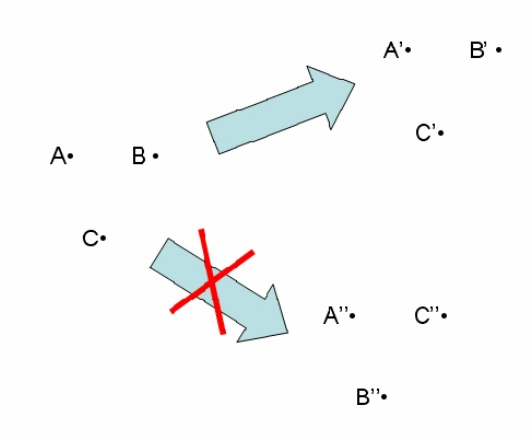

Figure 4. Possible Triplet Matches

When this surface is bent or viewed from another view angle, these three points appears as A', B' and C' and we have

$$Sign(\begin{vmatrix} X_A & Y_A & 1 \\ X_B & Y_B & 1 \\ X_C & Y_C & 1 \end{vmatrix}) \times Sign(\begin{vmatrix} X'_A & Y'_A & 1 \\ X'_B & Y'_B & 1 \\ X'_C & Y'_C & 1 \end{vmatrix}) = 1 \tag{2}$$

which means the orientation of (A, B, C) is consistent with (A', B', C'). On the contrary, (A, B, C) is inconsistent with (A', B', C') when

$$Sign(\begin{vmatrix} X_A & Y_A & 1 \\ X_B & Y_B & 1 \\ X_C & Y_C & 1 \end{vmatrix}) \times Sign(\begin{vmatrix} X'_A & Y'_A & 1 \\ X'_B & Y'_B & 1 \\ X'_C & Y'_C & 1 \end{vmatrix}) = -1 \tag{3}$$

When a point set S is matched to another point S', we define the score of this match as

$$\sum_{A,B,C \in S} (Sign(\begin{vmatrix} X_A & Y_A & 1 \\ X_B & Y_B & 1 \\ X_C & Y_C & 1 \end{vmatrix}) \times Sign(\begin{vmatrix} X'_A & Y'_A & 1 \\ X'_B & Y'_B & 1 \\ X'_C & Y'_C & 1 \end{vmatrix})) \tag{4}$$

An ideal match from a one point set to another n-point set has a score of $\binom{n}{3}$ when every triplet is consistent with its match. The worst match score is $-\binom{n}{3}$ (mirrored).

In order to obtain the best match between two sets, a maximum flow or Hungarian algorithm can be used, but such an algorithm has a complexity of $O(v^3)$, where v is the number of vertices of the bi-partition graph (often greater than 600 for a single page). Since we will apply this verification step to a list of candidate pages, it consumes most of the runtime and must be efficient. We use a greedy algorithm to find an approximate match instead. Consider the two point sets as a bipartite graph and the value of each edge is the Euclidian distance between the layout contexts of its two vertices. We find the edge with the smallest value, match its two vertices, remove this edge together with its vertices, and repeat this procedure m times to find m pairs of point matches. The score of these m matches is between $-\binom{m}{3}$ and $\binom{m}{3}$.

(a)

Successful retrieval, highest score = 1538

(b)

Failed to retrieve, highest score = 332

Figure 5. Example matches with point correspondences, m=30

6. Implementation and Experiment

We have collected the proceedings of CVPR04, 05, 06 and ICCV05, 12742 pages in total. Table 1 shows the number of pages from each proceedings. From every page we extract layout contexts and each layout context is indexed by its nearest neighbor from the lexicon. Every page is complied into a bag of indexes with their coordinates. The coordinates are for the verification step.

When retrieving, the camera captured image is also complied into a bag of indexes with coordinates after rotation normalization. The pages in the document database are sorted by their coverage of the bag of query indexes. No-

tice that we are only interested in the top part of this sorted list of pages, most of the pages are eliminated immediately. We verify the top 1000 pages from this sorted list and the page that gets the highest score is the result of our retrieval. Fig. 5 shows a successfully retrieval (a) and an unsuccessful (page not in databse) retrieval (b). From the point-to-point correspondence we can see that the successful retrieval has almost all lines "parallel" i.e. they intersect at a vanish point and has a high score (a), while the unsuccessful matches point in arbitrary directions and have a low score (b).

Our mobile retriever is implemented on an iMate Jamin (Windows Mobile 5.0) phone using a camera with 1024×1280 resolution. From each captured image we extract 100 layout contexts, each of which takes about 24 bytes together with its coordinates, and in total approximately 2.4KB is required per query. For simplification and logging purpose, in our current implementation we upload the image to the server and extract features on the server side. In the future, the image capturing, processing and feature extraction (green box in Fig. 1) can all be done on the mobile device and the communication from device to server will take less than one second via GPRS/EDGE/CDMA network.

To test the performance of our system, we randomly select 50 pages from the database and 50 pages that are not in the database and capture pictures of these pages as queries for retrieval. Among the first 50 captures, 45 were successfully retrieved; among the second 50, as expected, all are rejected. We show a scatter plot of successful and unsuccessful retrieves in Fig. 6 with their scores and ranks. We can see a clear gap between successful retrieval and rejection. Therefore when a page has a score greater than 800, we have a high confidence that it is a correct retrieval. When a page has a score less than 400 it must not be a correct retrieval. By setting a proper threshold, we can achieve an 100% accurate retrieval. However, a page with high score may not have a high rank since some of the layout contexts can be mismatched and this mismatch will only be corrected during verification step. Fig. 7 shows two successful retrievals and two failed retrievals. Our approach is robust under crinkled and warped degradations. Since it relies on the layout of words, it fails when there is a small amount of text present in the captured image. We also approximate projective transform locally using similarity transform, so it may fail when perspective distortion is too strong.

7. Conclusion and future work

In this paper we present an end-to-end system that retrieves original documents from a camera phone captured sample. We use a distinctive local "layout context" descriptor to represent features of the document image and we verify the retrievals using triplets orientation which is robust to page or imaging distortion. This verification draws a clear

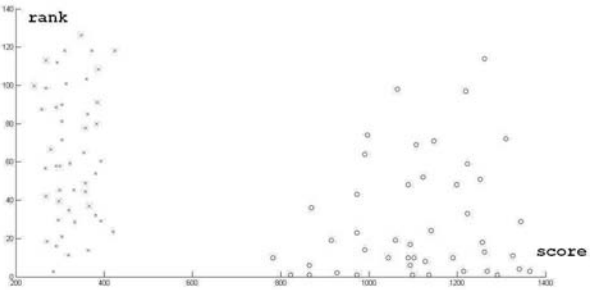

Figure 6. Score and rank of retrieval, dot: success, cross: rejection

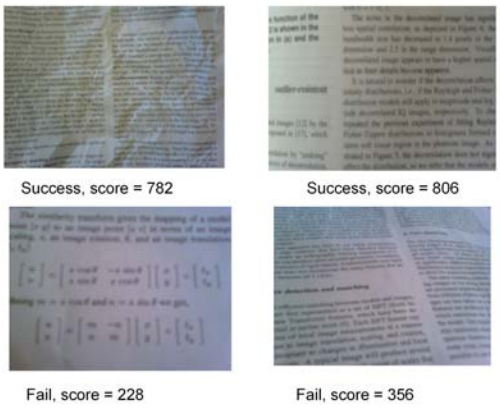

Figure 7. Successful and unsuccessful retrievals

Table 1. Data Collection

CVPR04	3397 pages
CVPR05	3501 pages
CVPR06	4001 pages
ICCV05	1843 pages
TOTAL	12742 pages

gap between successful and unsuccessful retrievals. A draw back of this verification is that it has to be applied to every page candidate and takes most of the runtime. On a Pentium 4, 2GHz CPU, a retrieval might take up to 20 seconds in going through 200 candidates. In future work, this verification may be replaced by a hashing of triplets which can accelerate speed. Another limitation with our approach and with most of the existing approaches is that, they are based on word and therefore targeted for Latin languages. For Asian languages such as Chinese, Japanese and Korean a new local feature descriptor has to be designed but the verification can still be applied.

References

[1] J. Cullen, J. Hull, and P. Hart. Document image database retrieval and browsing using texture analysis. *Proceedings of the 4th International Conference on Document Analysis and Recognition*, pages 718–721, 1997.

[2] P. David, D. DeMenthon, R. Duraiswami, and H. Samet. SoftPOSIT: Simultaneous Pose and Correspondence Determination. *International Journal of Computer Vision*, 59(3):259–284, 2004.

[3] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[4] Gartner. Nearly 50 percent of worldwide mobile phones will have a camera in 2006 and 81 percent by 2010. *Gartner*.

[5] J. Hare and P. Lewis. Content-based image retrieval using a mobile device as a novel interface. *Storage and Retrieval Methods and Applications for Multimedia 2005. Proceedings of the SPIE,*, 5682:64–75, 2004.

[6] J. Hull. Document image matching and retrieval with multiple distortion-invariant descriptors. *Document Analysis Systems*, pages 379–396, 1995.

[7] T. Kameshiro, T. Hirano, Y. Okada, and F. Yoda. A document image retrieval method tolerating recognition andsegmentation errors of OCR using shape-feature and multiple candidates. *Document Analysis and Recognition, 1999. ICDAR'99. Proceedings of the Fifth International Conference on*, pages 681–684, 1999.

[8] Y. Lu and C. L. Tan. Information retrieval in document image databases. *IEEE Transactions on Knowledge and Data Engineering*, 16(11):1398–1410, 2004.

[9] T. Nakai, K. Kise, and M. Iwamura. Hashing with Local Combinations of Feature Points and Its Application to Camera-Based Document Image Retrieval. *Proc. CBDAR05*, pages 87–94, 2005.

[10] T. Nakai, K. Kise, and M. Iwamura. Use of affine invariants in locally likely arrangement hashing for camera-based document image retrieval. *DAS06*, pages 541–552, 2006.

[11] W. Niblack. *An introduction to digital image processing*. Strandberg Publishing Company, Birkeroed, Denmark, Denmark, 1985.

[12] J. Sivic and A. Zisserman. Video Google: a text retrieval approach to object matching in videos. *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1470–1477, 2003.