Using Corner Feature Correspondences to Rank Word Images by Similarity

Jamie L. Rothfeder, Shaolei Feng and Toni M. Rath[*]
Multi-Media Indexing and Retrieval Group
Center for Intelligent Information Retrieval
University of Massachusetts
Amherst, MA 01002

Abstract

Libraries contain enormous amounts of handwritten historical documents which cannot be made available on-line because they do not have a searchable index. The wordspotting idea has previously been proposed as a solution to creating indexes for such documents and collections by matching word images. In this paper we present an algorithm which compares whole word-images based on their appearance. This algorithm recovers correspondences of points of interest in two images, and then uses these correspondences to construct a similarity measure. This similarity measure can then be used to rank word-images in order of their closeness to a querying image. We achieved an average precision of 62.57% on a set of 2372 images of reasonable quality and an average precision of 15.49% on a set of 3262 images from documents of poor quality that are even hard to read for humans.

1 Introduction

Traditional libraries contain an enormous amount of hand-written historical documents. These collections cannot be made available electronically on the Internet or on digital media if they do not include some sort of index. One possibility for adding an index would be to manually annotate each of the pages, and then to construct and index from this annotation. Manual transcription is, however, expensive and automation is a desirable alternative.

Using an automatic handwriting recognition approach seems like an obvious solution. However, the Optical Character Recognition (OCR) techniques which are used in such applications cannot be applied to index historical documents, since their success is mostly limited to applications with limited lexicons (e.g. automatic check processing) and *on-line* handwriting recognition, where pen position, velocity, etc. are recorded during writing. Furthermore, OCR is usually performed on newly scanned images of good quality, while historical documents suffer from degradation due to faded ink, stained paper, and other factors.

Word-spotting is an alternative method for indexing handwritten manuscripts. For this approach it is required that all documents in the collection were written by a single author (e.g. we use George Washington's manuscripts from the Library of Congress). Due to this constraint, instances of the same word will have a similar appearance. This enables us to cluster words based on their visual similarity, after they have been segmented from their respective documents. Clusters of frequent words (less *stop* words, such as 'the') are annotated, so that an index can be built for them. This is a matching and clustering approach for whole words, rather than a recognition solution, which would have to deal with such difficult issues as the well-known character segmentation problem.

In this paper, we present an algorithm that matches word-images by recovering correspondences between image corners, which have been identified by the Harris detector [2]. The similarity between two images is measured by the accumulated displacement of corresponding image locations.

Section 1.1 discusses past experiments done on word-matching with the George Washington data set. Section 2 explains the matching algorithm used. In section 3 we present experimental results that show the performance of the proposed algorithm in a retrieval task on the George Washington data set.

1.1. Related Work

Previous work on similar data (Thomas Jefferson's manuscripts) has shown that current character recognizers are not suitable for historic manuscripts: in [11] a recognizer was used to align a transcription with a manuscript by extracting lexicons for every word hypothesis. Even

[*]This work was supported in part by the Center for Intelligent Information Retrieval and in part by the National Science Foundation under grant number IIS-9909073. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

for very small lexicons of at most 11 words per segmented word, only 83% of the words could be correctly aligned with the transcription.

Word-spotting as a solution to indexing handwritten historical documents was initially introduced in [5]. The current scale-space approach to the word segmentation problem for noisy historical manuscripts is described in [6].

A number of authors propose different approaches to matching word/character images. They fall into two categories:

- *image-domain* matches: this category contains techniques such as straightforward counting of pixel-by-pixel differences (XOR), *Euclidean Distance Mapping* (EDM), which weighs difference pixels more heavily if they occur clumped together and *Sum of Squared Differences* (SSD) distance measures, which employs a non-linear penalty for pixel mismatches. All of these techniques were examined in [3] for their suitability as word matching algorithms.

- *feature-based* matches: here correspondences between image features are recovered and then used to compare the features. This category contains the work in [10], where an affine mapping between feature points is recovered (also implemented in [3] for word outline sample points), the shape context descriptor [1], which was highly successful for handwritten digit recognition and *Dynamic Time Warping* (DTW) based approaches such as in [4] and [7], where time series of image column features are aligned and compared. Of all these approaches, the DTW technique in [7] yielded the best matching performance (see comparison in that work).

The point-of-interest correspondence approach to matching, which we present in this work, is inspired by the research in stereo vision and mosaicking. Applications in these domains require robust determination of image correspondences, which is often supported by the identification of *points of interest*, such as corners. A number of point of interest operators were proposed in the literature (see [9] for a comparative study), from which we selected the *Harris* detector [2].

2. Matching

Matching words is critical to the success of word spotting. We examine the similarity of whole words in order to cluster them based on their appearance.

The first step when when matching two gray-scale images is to choose n points of interest in both of the images. Correspondences between these points are established by comparing local context windows with the Sum of Squared Differences (SSD) measure. The similarity is then judged

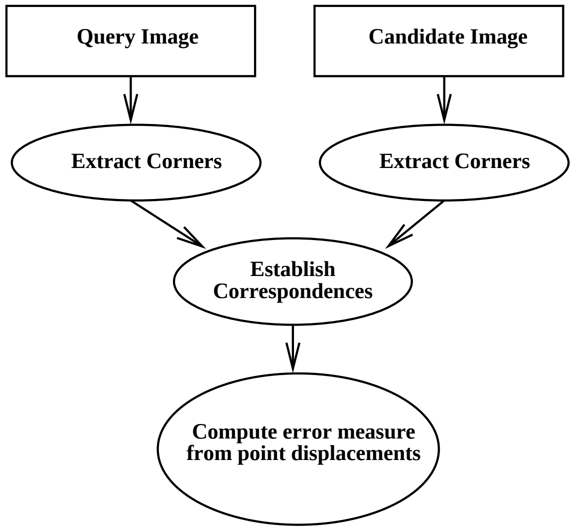

Query Image
Candidate Image
Extract Corners
Extract Corners
Establish Correspondences
Compute error measure from point displacements

Figure 1. Illustration of the procedure.

by the accumulated displacement of corresponding interest points in the two images. This calculation requires the selection of consistent origins in the images, which we define as the centroids of the detected interest points. See Figure 1 for an illustration of the matching procedure.

In the next sections we will explain the matching procedure in more detail after briefly outlining how word-images are obtained from the original documents.

2.1. Segmenting Words From the Document

Before the word-images can be compared, they must be segmented from their respective documents. We used an automatic word segmentation algorithm [6] which computes a projection profile in the horizontal direction for the entire document, in order to identify line boundaries. It then examines each line at different scales to determine when entire words appear as connected “blobs” (words break up into smaller units at higher scales, and inter-word spaces are missed at lower scales). Once the correct scale is determined, the words can easily be segmented from the document. Due to the cursive writing style, such segmented words often contain descenders (lower parts of ’g’) and ascenders (upper parts of ’f’) from lines below and above the segmented word.

2.2. Harris Corner Detector

To compare all of the points in the image would be an expensive procedure since our dataset includes thousands of word-images. We used the Harris corner detector to detect points of interest because of their repeatability, invariant to viewpoint changes and invariant to illumination changes [2].

This detector operates on the matrix

$$\mathbf{M} = \begin{pmatrix} (\frac{\partial I}{\partial x})^2 & (\frac{\partial I}{\partial x})(\frac{\partial I}{\partial y}) \\ (\frac{\partial I}{\partial x})(\frac{\partial I}{\partial y}) & (\frac{\partial I}{\partial y})^2 \end{pmatrix}$$

where I is the gray level intensity image. A corner is indicated when the 2 eigenvalues of M are large since this indicates grayscale variations in both the x and y directions. The number of corners detected in images is variable. Extra corners did not, however, complicate our evaluation method since we do not penalize for non-corresponding points. Furthermore, we assume that the Harris detector will find about the same number of extraneous corners in images of similar size and content.

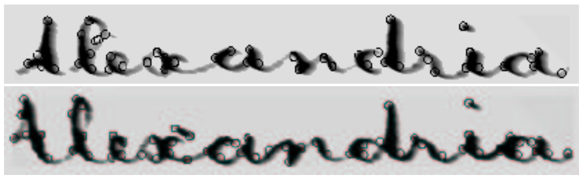


Figure 2. Corners detected with the Harris corner detector on two gray level images.

Figure 2 shows the results of the Harris corner detector on two images. We see that most of the points found (denoted with circles) in the two images have correspondences.

2.3. Recovering Corner Correspondences

The focus in this section is on determining correspondences between two sets of corner points, that were detected with the Harris operator. Most correspondence methods compare the characteristics of the local regions around feature points, and then select the most similar pairs as correspondences. The characteristics of local regions can be represented by either a feature vector, (e.g. see [8]) or by windows of gray-level intensities.

We use the Sum of Squared Differences (SSD) error measure to compare gray-level intensity windows, which are centered around detected corner locations. The reason for selecting the SSD measure is its simplicity and the low number of operations needed to calculate it; an important consideration when comparing a large number of image pairs. Given the windows M and N of size $n \times n$, centered at one feature point in the query image and one in a database image respectively, the SSD error measure is calculated as follows:

$$SSD(M,N) = \sum_{i=1}^{n} \sum_{j=1}^{n} (M(i,j) - N(i,j))^2 \qquad (1)$$

Due to its simplicity, there are some limitations inherent to the SSD error as a similarity measurement for feature points. The following are some factors that can cause false matches. We try to alleviate them with constraints.

- the size of the query word may be different from that of the database word image, that is, they have different resolutions. Assuming tight bounding boxes for all words, we resize all candidate images to the size of the query image.

- For a given feature point, there might be several feature points in a database image, which result in small SSD errors. This happens when distinct strokes in the database images have very similar local regions. To decrease the probability of false matches, we add a local constraint when matching feature points: corresponding feature points in the database images must appear in a neighborhood of the corner point in the template image. This constraint is valid in most cases, because the characters are arranged in the same order for the same words. Furthermore, the constant allows some feature points to have no correspondence in the database image. This makes sense, because of the possibility of partial visibility of strokes or because the template and the database image do not match. The proportion of features with correspondences and the total number of detected features in the query image will be used as a factor when calculating the distance[1] between a database image and the template in the next step.

The whole matching process is implemented on images, which are half the size of the originals. This is “equivalent” to enlarging the SSD window (parameter n) to twice the size we are using, without increasing the computations. This also reduces the number detection of false correspondence pairs, by increasing the context that is taken into consideration when determining correspondences.

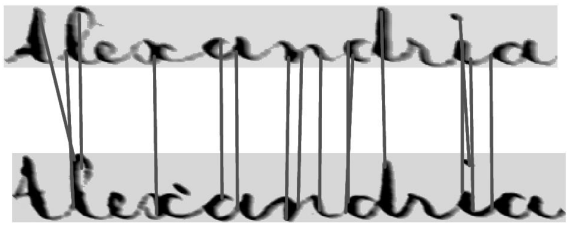


Figure 3. Recovered correspondences in two word images.

Preliminary experiments showed that adding these constraints greatly improved the matching accuracy (see Figure

[1]This distance does not necessarily satisfy all metric axioms.

3 for an example of recovered correspondences).

2.4. Distance Measure Calculation

The correspondence between pairs of feature points captures the similarity between local regions of two images. Our goal is now to combine this local information into a global measure of similarity.

Various approaches were investigated: they include the total sum of SSD errors for all corresponding pairs, counting the number of recovered correspondences and calculating the Euclidean distance between correspondences.

Using the Sum of SSD errors of the correspondences is sensitive to global brightness variations. The measure tends to assign very low ranks to images of the same word if their backgrounds have very different gray levels or when the writing strokes have different ink properties.

Measuring the similarity by counting the number of recovered correspondences falls in the voting method category: each pair of feature points is counted as a vote, if the similarity is larger than a given threshold. In the end, the images are ranked according to the number of votes they received. One problem with this method is that the sparseness of feature points detected in some template images makes it difficult for limited vote numbers to determine the similarities of numerous database images.

In our method, we used Euclidean distances of correspondences as the distance measurement between two images. Unlike the above two measurements, the Euclidean distance measurement to some extent takes the structure of two words into account, because it assumes the feature points for same words should have similar spatial distribution in different images. Our distance measurement is formulated as:

$$D(A,B) = \frac{\sum_i \sqrt{(x_{bi}-x_{ai})^2 + (y_{bi}-y_{ai})^2}}{\#\text{correspondences}} \cdot \frac{\#\text{feature points in A}}{\#\text{correspondences}} \quad (2)$$

where A is the query image, B a database image; (x_{ai}, y_{ai}) and (x_{bi}, y_{bi}) are the coordinates of a pair of corresponding feature points, in the query image and the database image respectively. Here, we multiply the Euclidean distance with a factor indicating the proportion of correspondences with the feature points in the template image. Thus, the fewer corresponding feature points are found in the database image, the larger the distance is between it and the template image.

3. Experimental Results

3.1. Data Sets and Processing

We performed experiments on the same two test sets of different quality images that were used in the evaluation in [7]. Both of these sets were 10 pages in size. Figure 4(a) is an example from the first set, which is of acceptable quality. Figure 4(b) is an example of the second set which is so degraded that it is difficult even for people to read. We used this second set to test how poorly the algorithms would perform. Several variations of the algorithms were tested and the results presented are the best that we achieved. The four sets were constructed as follows:

A: 15 images in test set 1.

B: entire test set 1 (2381 images total, 9 without word content[2]).

C: 32 images in test set 2, analyzed in [3] (for comparison purposes). 13 of these only occur once (i.e. the image itself) in the collection.

D: entire test set 2 (3370 images total, 108 without word content[2]).

The subsets A and C allow us to test algorithms which would otherwise take too long to run on the entire dataset.

3.2. Evaluation Method

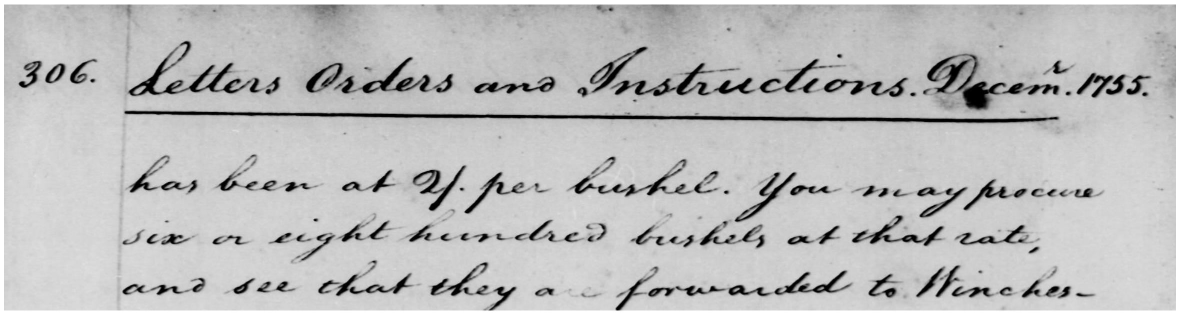
Images in the data sets were tagged manually with their ASCII equivalent. Partial images were annotated with what letters were visible. We used our algorithm to compare every image against every other image in the data set [3] and create lists of images ranked by similarity. Two images were considered relevant if their ASCII tags matched. `trec_eval` was then used to produce the average precision scores in table 1.

3.3. Results

Table 1 contains the results from experiments performed on the George Washington collection using various word-matching algorithms:

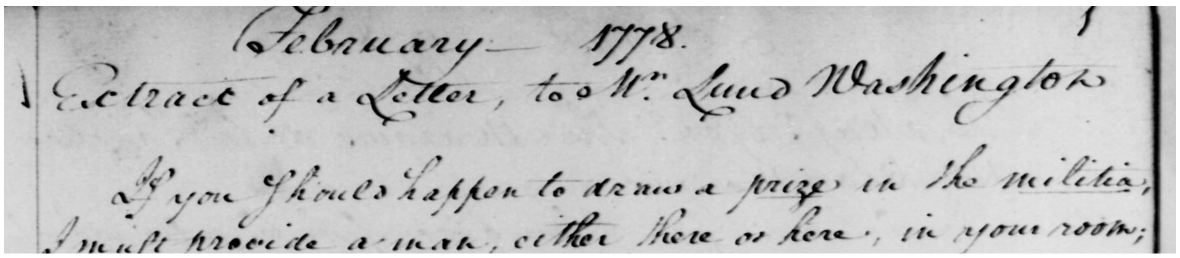
1. XOR[3]: aligns images and counts pixel-by-pixel differences.
2. SSD[3]: images are shifted relative to one another in order to find the minimum cost which is calculated using the SSD error measure.

[2]These images result from segmentation errors.
[3]Image pairs were discarded if they were not similar according to simple heuristics, such as image length.


306. Letters Orders and Instructions. Decem.r 1755.

has been at 2/. per bushel. You may procure
six or eight hundred bushels at that rate,
and see that they a forwarded to Winches-

(a) Good quality image.


February 1778. 1
Extract of a Letter, to Mr. Lund Washington

If you should happen to draw a prize in the militia,
I must provide a man, either there or here, in your room;

(b) Bad quality image.

Figure 4. Example images from the two testsets used in the evaluation.

3. SLH[3]: uses the Scott and Longuet-Higgins algorithm [10] to recover an affine warping transform between sample points taken from the outlines of words. The residual between template and warped points is the distance measure.

4. SC [1]: shape context matching.

5. EDM[3]: like XOR, but weighs clusters of pixel differences more heavily.

6. DTW[7]: Matching using dynamic time warping on column feature time series.

7. CORR: Approach proposed in this work.

In Kane et al.'s experiments [3], each query image is also part of the candidate image set. This pushes the scores up, since the query image is always retrieved at rank 1 for most matching methods. In order to provide a more realistic view of our algorithm's performance, we have calculated average precision scores without considering queries as candidates: the four rightmost columns in table 1 contain the resulting scores for test runs that were available to us in raw (i.e. ranked list) format. The remaining entries in the table were obtained with Kane et al.'s evaluation method.

CORR outperformed all methods on the set of 15 good quality images (set A) and performed slightly worse than DTW on the corresponding full set B. CORR also lead when tested on the smaller set of 32 poor quality images (set C), but fell behind DTW when tested on the corresponding full set D. Overall, our approach performed better than all methods except DTW, for which it performed almost as well. In addition to this, CORR is twice as fast as DTW and much faster than the other approaches evaluated in this work (see table 2). The success of the technique results from picking good points of interest. Our intuition for the good performance is that the Harris corner detector picks up points that provide a good description of the overall image structure.

We found that our algorithm would fail to assign a correct score to two corresponding images when one image was shifted significantly (for example, due to a segmentation error). We believe that this failure is a result of choosing the upper left-hand corner as the origin when computing the Euclidean distance. When this upper-left hand corner is chosen for two images, one of which is shifted x pixels, x will be added to the error of each corresponding point of interest. We attempted to correct this by implementing a common origin detector using several different methods. However, this detector did not perform well and the results did not improve. We also found that the precision scores from

Test set/Algor.	XOR	SSD	SLH	SC	EDM	DTW	CORR	SC	EDM	DTW	CORR
A	54.14%	52.66%	42.43%	48.67%	72.61%	73.71%	73.95%	40.58%	67.67%	67.92%	69.69%
B	n/a	n/a	n/a	n/a	n/a	65.34%	62.57%	n/a	n/a	40.98%	36.23%
C	n/a	n/a	n/a	48.11%	49.56%	58.81 %	59.96%	9.46%	n/a	13.04%	14.84%
D	n/a	n/a	n/a	n/a	n/a	51.81%	51.08%	n/a	n/a	16.50%	15.49%

Table 1. Average precision scores on all data sets(results for test set A and B have been corrected, XOR: matching using difference images, SSD: sum of squared differences technique, SLH: technique by Scott & Longuet-Higgins [10], SC: shape context matching [1], EDM: euclidean distance mapping, DTW: dynamic time warping matching, CORR:recovered correspondences). Scores to the right of the double line were calculated using a different evaluation method (see text).

Algorithm:	XOR	SSD	SLH	SC	EDM	DTW	CORR
Running time [s]:	13	72	121	∼50	14	∼2	∼1

Table 2. Run times for the compared algorithms in *Matlab* **on a 400MHz machine. Given values include time necessary for normalization (e.g. relative scaling), feature extraction, and similar processing steps.**

smaller words which occur frequently (for example, stop words such as “the”, “and”, “of”) were much lower than the scores from larger words. We speculate two reasons for this are that smaller words have less points of interest and that the fixed SSD window was trained on larger images. A dynamic window size may improve performance.

4 Summary and Conclusions

We presented a simple algorithm that matches word-images by recovering correspondences between points of interest. Specifically, we were able to match good quality images with around 64% accuracy with each image pair taking only 1 second to be compared. Our further development of this algorithm will include adding adaptive window sizes, and implementing an indexing scheme. We believe that this will greatly improve both the running time and accuracy.

Acknowledgments

We thank R. Manmatha for suggesting the idea and the Library of Congress for providing the scanned images of George Washington’s handwritten manuscripts. We used Peter Kovesi’s (University of Western Australia) Harris Detector code and thank him for making it available on line.

References

[1] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24:509–522, 2002.

[2] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147–151, 1988.

[3] S. Kane, A. Lehman, and E. Partridge. Indexing George Washington’s handwritten manuscripts. Technical report, Center for Intelligent Information Retrieval, University of Massachusetts Amherst, 2001.

[4] A. Kołcz, J. Alspector, M. Augusteijn, R. Carlson, and G. V. Popescu. A line-oriented approach to word spotting in handwritten documents. *Pattern Analysis & Applications*, pages 153–168, 2000.

[5] R. Manmatha, C. Han, E. M. Riseman, and W. B. Croft. Indexing handwriting using word matching. In *Digital Libraries ’96: 1st ACM Int’l Conf. on Digital Libraries*, pages 151–159, 1996.

[6] R. Manmatha and N. Srimal. Scale space technique for word segmentation in handwritten manuscripts. In *Proc. 2nd Int’l Conf. on Scale-Space Theories in Computer Vision*, pages 22–33, 1999.

[7] T. M. Rath and R. Manmatha. Word image matching using dynamic time warping. Technical report, Center for Intelligent Information Retrieval, University of Massachusetts Amherst, 2002. accepted for CVPR’03.

[8] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 19(5):530–535, 1997.

[9] C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of interest point detectors. *International Journal of Computer Vision*, 37(2):151–172, 2000.

[10] G. L. Scott and H. C. Longuet-Higgins. An algorithm for associating the features of two patterns. *Proc. of the Royal Society of London*, B224:21–26, 1991.

[11] C. I. Tomai, B. Zhang, and V. Govindaraju. Transcript mapping for historic handwritten document images. In *Proc. 8th Int’l Workshop on Frontiers in Handwriting Recognition 2002*, pages 413–418, 2002.