International Journal on Document Analysis and Recognition manuscript No.
(will be inserted by the editor)

Word matching using single closed contours for indexing handwritten historical documents

Tomasz Adamek, Noel E. O'Connor, Noel Murphy, Alan F. Smeaton

Centre for Digital Video Processing, Dublin City University, Dublin 9, IRELAND
e-mail: {adamekt,oconnorn,murphyn}@eeng.dcu.ie, alan.smeaton@computing.dcu.ie

The date of receipt and acceptance will be inserted by the editor

Abstract. Effective indexing is crucial for providing convenient access to scanned versions of large collections of handwritten historical manuscripts. Since traditional handwriting recognizers based on Optical Character Recognition (OCR) do not perform well on historical documents, recently a holistic word recognition approach has gained in popularity as an attractive and more straightforward solution [1]. Such techniques attempt to recognize words based on scalar and profile-based features extracted from whole word images. In this paper, we propose a new approach to holistic word recognition for historical handwritten manuscripts based on matching word contours instead of whole images or word profiles. The new method consists of robust extraction of closed word contours and the application of an elastic contour matching technique proposed originally for general shapes [2]. We demonstrate that contour-based descriptors can effectively capture intrinsic word features. Our experiments show a recognition accuracy of 83%, which considerably exceeds the performance of other systems reported in the literature.

Key words: Historical manuscripts – Holistic word recognition – Contour matching – Annotation – Indexing

1 Introduction

The approach described in this paper is motivated by the work of Lavrenko, Rath and Manmatha [3][4][1] who promote the idea of holistic word recognition for handwritten historical documents. Their main focus is on achieving reasonable recognition accuracy, which enables retrieval of handwritten pages from a user-supplied ASCII query. They argue that, although currently it is not feasible to produce near-perfect recognition results, satisfactory retrieval can still be performed using the noisy outputs of recognizers [5]. In their word spotting approach for indexing historical handwritten manuscripts, pages are segmented into words which are then matched as images and grouped into clusters which contain all instances of the same word. A partial index is constructed for the collection by tagging a number of the resulting clusters.

Since the quality of historical documents is often significantly degraded it is crucial to select the right features for word matching. The feature set proposed by Rath and Manmatha [3] include smoothed versions of the word images and various profile-based features. For example, the shape of a word is captured by upper and lower word profiles extracted by traversing the upper (lower) boundary of a word's bounding box and recording for each image column the distance to the nearest "ink" pixel in that column. In order to capture the "inner" structure of a word these features were extended by the number of background to "ink" transitions. It appears that the biggest limitation of the above set of features is their strong dependency on good normalization, mainly skew/slant angle normalization and baseline[1] detection. It was shown in [1] that the application of statistical language models can further improve word recognition accuracy as a post-processing step. A simple Hidden Markov Model with one state for each word resulted in over 10% improvement in recognition accuracy.

In this work, our primary goal is to investigate the possibility of matching words using their contours instead of whole images or word profiles. We believe that contour-based descriptors can better capture a word's important details and eliminate the need for skew and slant angle normalization. Furthermore, elastic contour matching based on a Dynamic Programming (DP) technique provides a flexible way to compensate for inter-character and intra-character spacing variations. In this initial approach, we assume that bounding boxes for each word are already known [1][6]. The inner holes in letters and dots will be ignored.

The remainder of this article is organized as follows: the next section describes extraction of a single closed contour for each word. Then, the contour matching algorithm proposed originally for general shapes [2] is briefly described in section 3. Next, experimental results are presented in section 4. Our outlook on future research is discussed in section 5 and conclusions are formulated in section 6.

[1] Baseline - imaginary line upon which a line of text rests.

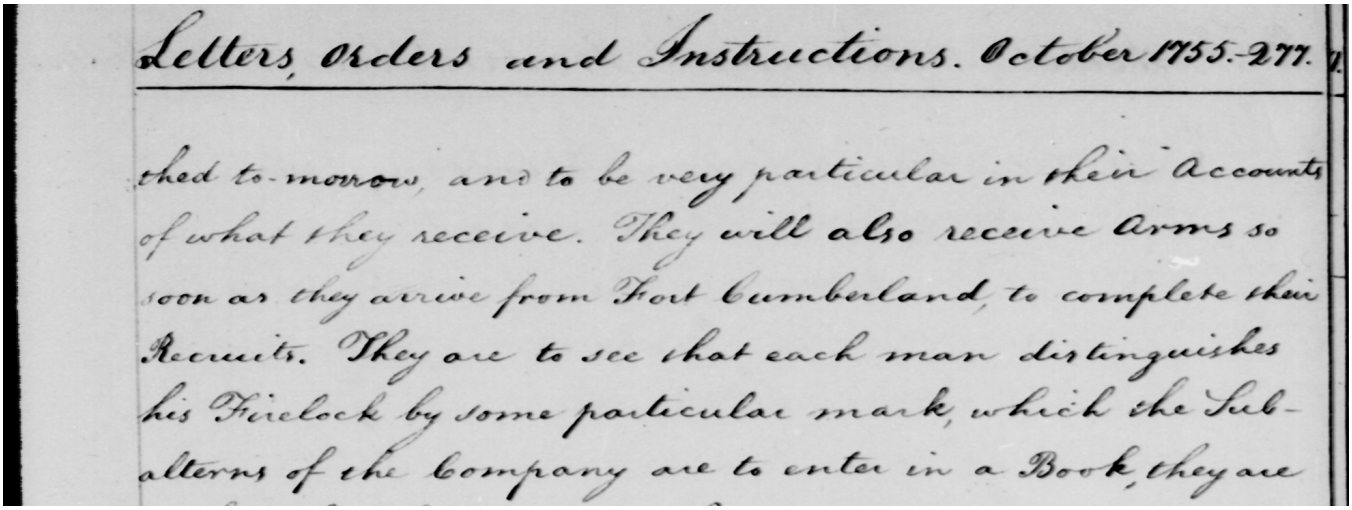


Fig. 1. Manuscript sample (2770277).

2 Contour extraction

Extraction of a single closed contour for each word is a key component of the proposed approach. The extraction procedure has to deal with the poor quality typical of manuscripts (noise, stained paper, etc), contrast variations (faded ink, differences in pressure on the writing instrument) and the most challenging problem – disconnected letters. Figure 1 shows a sample of a typical manuscript under consideration. Contour extraction is performed in five steps: binarization, localization of the main body of lower case letters, connected components labelling, connecting disconnected letters and contour tracing.

2.1 Binarization

In the first step, the pixels from the input grey level image are classified as either "ink" or "paper" by comparing their values with a threshold. Since the contrast of the input word image can vary considerably, mainly due to differences in pressure on the writing instrument, there is no single optimal threshold that provides acceptable binarization for different parts of the word. Therefore, the optimal threshold has to be estimated individually for each pixel based on its local neighborhood. We chose Niblack's [7] algorithm where the threshold T for a given pixel is computed using the mean m and the standard deviation σ of the gray values in a small window centered at the pixel. The threshold is calculated according to formula proposed by Sauvola et al. [8]:

$$T = m \cdot (1 - k \cdot (1 - \frac{\sigma}{R})) \quad (1)$$

where k is a constant set to 0.02 and R denotes the dynamics of the standard deviation and is set to 128.

However, the above approach does not produce smooth word outlines and as such was modified. Prior to binarization, morphological filtering [9] is used to create eroded and opened versions of the input image - the structuring element employed is shown in Figure 2. The dynamic threshold T is calculated based on the eroded image. The binary classification of a single pixel as "ink" or "paper" is made by comparing its value from the opened image with the dynamic threshold calculated for this pixel based on the eroded image. An example of binarization is shown in Figure 3. Note that weak strokes (top of double 't' and connections between 'r' and 's') were detected and even dilated without widening the strong strokes.

0	1	0
1	1	1
0	1	0

Fig. 2. Kernel used for morphological filtering.

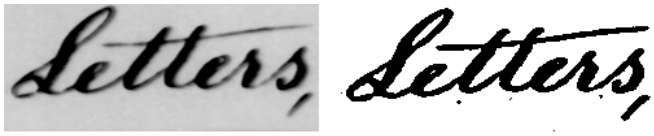


Fig. 3. Binarization using dynamic threshold.

2.2 Position estimation

Often binarization alone is not sufficient to obtain a single connected "ink" region for each word. In such cases, separated parts of the word have to be connected. The proposed connecting procedure utilizes the position of the word baseline and the x-height[2]. Detection is performed by analysis of the number of "ink" pixels in each line of the word binary image – see Figure 9. Similarly, horizontal boundaries of the text can be detected by analyzing the number of "ink" pixels in each column of the binary word image. Additional rules can be applied to eliminate the residuals of the vertical margin line as illustrated in the example shown in Figure 9(c).

[2] x-height - distance between the baseline and tops of the main body of lower case letters.

It should be noted that the position of the baseline and the length of x-height are not directly used by the matching algorithm, but serve only as guidelines for the robust extraction of word contours. Moreover, if all "ink" pixels are already connected in the binarization step, the proper contour can be extracted even if the estimated baseline and x-height are inaccurate.

2.3 Connected component labelling

The connected component labelling algorithm scans the word's binary image and groups "ink" pixels into components based on pixel 8-neighborhood connectivity. Once all groups are determined, each pixel is labelled according to the component it was assigned to. Only components containing a sufficient number of pixels within the area of the main body of lower case letters are retained for further processing. As a result, small "ink" regions and letter residuals from the line above or below the current word are eliminated – examples are shown in Figure 9(4th column).

2.4 Connecting disconnected letters

Once the relevant components are identified they are sorted based on the horizontal positions of their center of gravity. Then, successive components are connected by adding the best connecting link ("ink line") into the binary image. The best candidate for the link between two disconnected components is chosen in the following procedure. At first, candidate links are created by pairing contour points from both connected components. This is followed by link validation. A link is considered as valid only if:

- both ends of the link are inside, or within a close distance to, the area of the main body of lower case letters – see examples (a), (b) and (e) in Figure 9;
- both ends of the link are considerably far from the main body of lower case letters – see the top of letter 'I' in example 9(h).

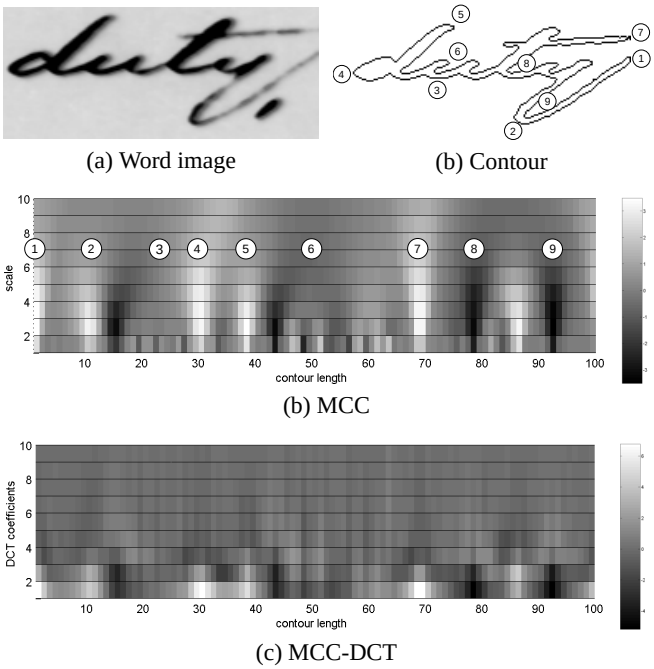
The shortest valid link is chosen as the best "ink" line connecting two components. Links connecting "ink" regions are highlighted in red in Figure 9(4th column).

2.5 Contour tracing

Once all components are connected with the links, the final binary mask for the word can be created – see examples from the 5th column in Figure 9. This is followed by a contour tracing procedure which extracts a single ordered contour for each word – see examples from the last column in Figure 9.

3 Contour matching

In our approach, similarities between word contours are measured using the contour matching technique proposed originally for comparing general shapes [2][10].



(a) Word image (b) Contour

(b) MCC

(c) MCC-DCT

Fig. 4. Extraction of MCC representation.

In [2], we introduced a rich shape description method, termed the Multi-scale Convexity Concavity (MCC) representation. In this representation, information about the amount of convexity/concavity at different scale levels is stored for each contour point (Figure 4(b)). A Dynamic Time Warping (DTW) technique [11] is used to find an optimal refined alignment along the contours upon which the dissimilarity measure is defined. The approach is robust to several transformations including translation, scaling, rotation, modest occlusion and symmetric transformation. The method performs particularly well in cases of elastic deformations and where the similarity between curves is weak.

In [10], we proposed an alternative version of the MCC representation, termed MCC-DCT, where a one dimensional Discrete Cosine Transform (DCT) is applied to each multi-scale contour point feature vector de-correlating information from different scale-levels and placing most of the energy in low frequency coefficients. In addition, we proposed an iterative optimization framework for determining the relative proportions in which information from different DCT coefficients should be combined in the final similarity measure. We showed that such optimization can further improve matching performance for a particular application.

For the purpose of this work, we used the MCC-DCT version of the contour representation. The matching procedure was optimized using the shape collection from the MPEG-7 Core Experiment "CE-Shape-1" (part B) [12]. It should be noted that this dataset contains general shapes and does not contain any examples of word contours. An example of matching word contours using the above method is shown in Figure 5. A more detailed description of the MCC representation and the associated matching algorithm can be found in [2].

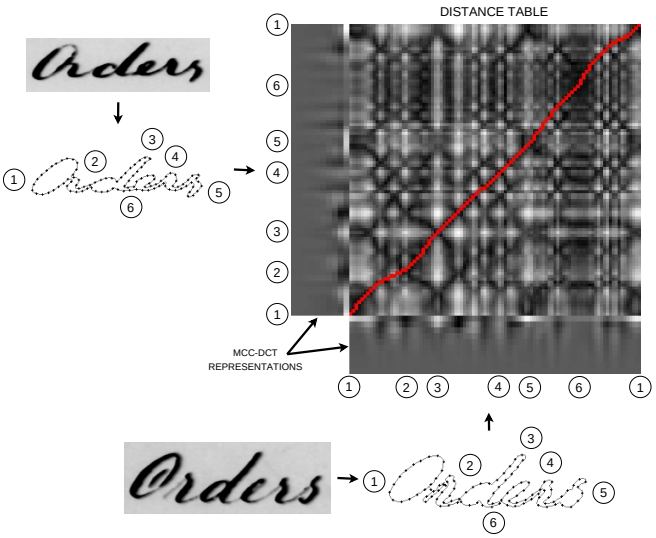


Fig. 5. Matching example. Finding the optimal match between two contour representations corresponds to finding the lowest cost diagonal path through the table containing pairwise distances between contour point features from both words. In our original approach all possible combinations of starting points are examined to ensure an optimal match. The optimal path has to begin and end at the cell corresponding to the alignment of the two starting contour points under consideration. Deviations of the path from a straight diagonal path compensates for elastic deformations between contours i.e. stretching and shrinking of parts.

4 Experiments

4.1 Overall Results

For our experiments we used a set of 20 pages from the George Washington collection at the Library of Congress. These contain 4,856 word occurrences of 1,187 unique words. The collection is accurately segmented into words, and ground-truth annotations for the word images were created manually [1].

The contours for all words were extracted according to the procedure described in section 2. Then, for each word contour the MCC-DCT descriptor with 100 equally spaced contour points, each with a feature vector consisting of 10 DCT coefficients, was extracted and stored.

Each of the 4,856 word occurrences was used as a query and its contour representation was matched against word representations from all manuscript pages except the page with the query word. A 1-nearest neighbor method was used to classify the query word and the classification result was compared against manual annotation. It should be noted that words from the same manuscript page as the query were excluded from matching for comparability reasons with the results reported in [1].

The results were measured in terms of average Word Error Rate (WER). To separate out-of-vocabulary (OOV) errors from mismatches we report two types of WER, one that includes OOV words and one that omits them from the evaluation. The lowest WER reported until now in the literature for the George Washington collection was 0.449 when OOV words were included and 0.349 when OOV words were excluded [1]. It should be noted that this result was obtained after utilizing additional resources to train a statistical language model. Using a 27-dimensional feature vector representation of each word, without any language model post-processing yielded a WER of 0.603 when OOV words are included and 0.531 when OOV words are excluded. In comparison our method showed an average WER of 0.306 with OOV words and 0.174 without OOV words. This represents approximately 50% reduction of the non-OOV error rate obtained by our system despite the fact that it is still at an early stage of development and despite the fact that it is based on contour mapping only.

4.2 Comparison with CSS

The contour extraction procedure presented in section 2 opened a new possibility of utilizing matching techniques requiring ordering of the contour points for the purpose of word matching. In this section, we evaluate the performance of the *Curvature Scale Space* (CSS) approach [13], which was adopted as contour-based descriptor by the ISO/IEC MPEG7 standard, in the context of word recognition. The CSS descriptor is very compact, allows fast matching, and extensive tests using general shape collections [14] revealed that the method is quite robust with respect to noise, scale and orientation changes of objects. In the experiment, the CSS extraction and matching was performed using the MPEG7 eXperimentation Model (XM software v5.6) [15].

Adopting the CSS technique for recognition of words from the George Washington collection resulted in relatively high average WER of 0.350 (excluding OOV words). Such poor performance can be explained by two major drawbacks of the CSS technique: the occurrence of ambiguity with regard to concave segments and inability to represent convex segments. This result confirms the need for utilization of a rich shape descriptor, as for example the one presented in section 3.

4.3 Towards fast recognition

The contour matching technique described in section 3 has been proposed originally for comparing general shapes [2][10]. Although the method can be used without any changes, in this section we will demonstrate possibilities for further adaptation to the specific task of word matching in order to improve performance and reduce computational load. In addition, a pruning technique for discarding unlikely matches will be proposed. All experiments presented in this section were performed using a standard PC with 1.6GHz Pentium 4 processor. For clarity, only WER without OOV words will be considered. Note that the goal of the experiments is to demonstrate the influence of various parameter settings on speed and recognition performance and not a proposal for the final tuning of the algorithm. We are aware the latter case would require use of two collections, one for tuning and one for testing.

Modifications of the matching algorithm. When matching two shapes, their relative rotation and therefore the circular shift between their contour representations, is generally unknown.

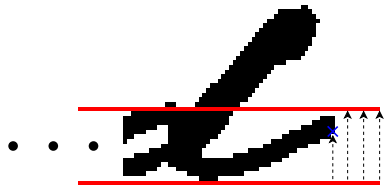


Fig. 6. Location of starting point.

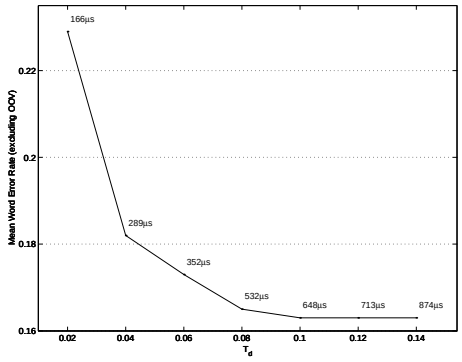


Fig. 7. Word Error Rate (excluding OOV) as a function of allowed path deviation T_d (starting points utilized). Labels at each data point denote times required for matching two word contours for a given setting of parameter T_d.

Hence all circular shifts have to be examined to ensure an optimal match, implying a total complexity of $O(N^3)$, where N is the number of contour points used. In the case of word matching, we can locate a single starting point for each contour in such a way that its position along the contour is consistent among different instances of the same word. Close to optimal matching can then be performed by examining only one circular shift corresponding to the alignment of starting points and therefore reducing the matching complexity to $O(N^2)$. We propose to locate the starting point from the part of the contour corresponding to the end of the word, rather than the beginning, due to better shape consistency at this point. The point is located by scanning the main body of lower case letters and searching for the first "ink" pixel. The scanning starts at the right-bottom and is performed from right to left and from bottom to top as shown in Figure 6. Utilizing the location of starting points, reduced the average matching time for two words from $6ms$ to less than $289\mu s$ whilst maintaining a low average WER of 0.182 (without OOV words).

In the case of the matching algorithm used, finding the optimal match corresponds to finding the lowest cost diagonal path through the table containing pairwise distances between contour points from both words – as explained in Figure 5. The path has to begin and end at the cell corresponding to the alignment of the two starting contour points. Deviations of the path from a straight diagonal path compensates for inter-character and intra-character spacing variations.

It is common practise in the case of DTW techniques [11], to restrict the path to lie in the area close to the ideal straight diagonal line in order to speed up the matching process. In our original approach, the path is allowed to deviate from the straight diagonal path by no more than a predefined deviation threshold $\pm T_d \cdot N$, where $T_d = 0.04$ (reducing the complexity to $O(2T_dN^2)$). In most cases when matching general shapes, this restriction imposed on the path's geometry had no affect on the optimal path. In order to investigate the influence of this restriction on the word recognition rate we plotted WER against different values of the parameter T_d together with corresponding times required for matching two word contours – see Figure 7. From the plot we can observe that constraining the path to the diagonal line would result in WER above 0.230 which confirms the requirement of elastic matching. The experiment also verified the linear dependency between T_d and computational load of the matching. For the remainder of this article we adopt $T_d = 0.08$ as a good trade-off between recognition efficiency and matching speed. Although, this setting increases the average matching time to $532\mu s$ whereas the average WER without OOV words is further reduced to 0.165(starting points utilized).

Table 1. Effect of pruning based on global contour complexity.

τ_x	$\frac{\sharp pruned\ pairs}{\sharp total\ pairs}$ [%]	WER (excluding OOV words)
0.1	80%	0.215
0.2	**63%**	**0.170**
0.3	48%	0.165
0.4	37%	0.165
...	...	...
∞	0%	0.165

Pruning unlikely matches. Pruning techniques can be used to quickly discard unlikely matches by requiring word images to have similar statistics [4]. In [16], pruning of word pairs was performed based on the area and aspect ratio of their bounding boxes. This idea was further extended in [17] by the additional requirement of two words to have the same number of descenders (strokes below the baseline). In our approach, the pruning statistics are extracted directly from the word contours. Pruning is performed based on contour complexity and the number of descenders and ascenders. For clarity, each pruning rule will be first discussed independently and only the best settings for each of the rules will be used in the final combined pruning criterion.

Shape complexity x_i of a word contour C_i can be defined as the ratio between its perimeter length l_i and the square root of its area a_i: $x_i = l_i/\sqrt{a_i}$. Both can be easily estimated directly from the word contour [18]. The following rule was used to discard unlikely matches [19]:

Rule 1: Two word contours C_i and C_j are similar only if $|x_i - x_j|/min\{x_i, x_j\} \leq \tau_x$.

Clearly, increasing τ_x in *Rule 1* allows for more matches to be processed by the matching algorithm (reducing recognition speed) and leading to lower values of average WER. Table 1 demonstrates empirically the effect of pruning using different values of threshold τ_x on WER and number of pruned pairs. The results show that setting $\tau_x = 0.3$ would prune almost half of the total number of word pairs, which would otherwise have to be processed by the matching algorithm, while maintaining low average WER of 0.165.

Words' descenders and ascenders were identified by traversing their contours and labelling contour points as belonging to a descender or an ascender based on the comparison be-

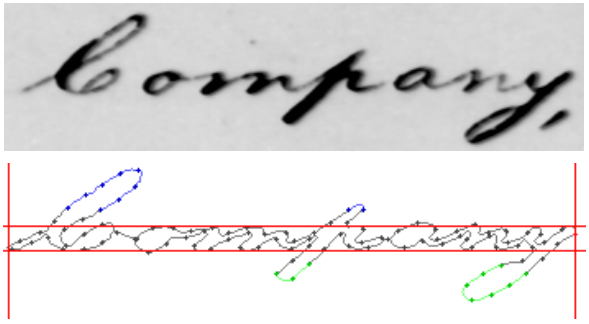


Fig. 8. Identification of descenders and ascenders.

Table 2. Effect of pruning based on number of descenders.

τ_{desc}	$\frac{\sharp pruned\ pairs}{\sharp total\ pairs}$ [%]	WER (excluding OOV words)
0	**50%**	**0.180**
1	9%	0.164
2	2%	0.165
...	...	...
∞	0%	0.165

tween their vertical positions with the vertical limits of the body of lower case letters – see Figure 8. Final numbers of ascenders and descenders were found by counting the number of continues sequences of points marked as ascenders or descenders.

Let $DESC_i$ and ASC_i be the estimated number of descenders and ascenders of word contour C_i. The following two rules were used to discard unlikely matches:

*Rule 2:*Two word contours C_i and C_j are similar only if $|DESC_i - DESC_j| \leq \tau_{desc}$.

*Rule 3:*Two word contours C_i and C_j are similar only if $|ASC_i - ASC_j| \leq \tau_{asc}$.

The thresholds τ_{desc} and τ_{asc} were introduced to control the tolerance for the absolute differences between numbers of descenders and ascenders when pruning word pairs and should have only integer values. Tables 2 and 3 demonstrate the effects of pruning on the average WER and the number of pruned pairs using rules 2 and 3 respectively. Table 2 shows that the number of descenders provides very reliable evidence for identifying unlikely matches. Specifically, $\tau_{desc} = 0$ would lead to early discarding of 50% of word pairs without noticeable difference in average WER. In contrast, pruning based on the number of ascenders (Table 3) is somehow less reliable due to generally more complex shape of the upper parts of the words. Therefore, to ensure that only small proportion of valid matches will be discarded by the *Rule 3* we have to allow certain tolerance, e.g. $\tau_{asc} = 1$ would result in WER of 0.164.

In the last experiment all three pruning rules were combined:

*Rule 4:*Two word contours C_i and C_j are similar only if Rules 1,2 and 3 are satisfied.

Incorporating the above rule and setting $\tau_x = 0.2$, $\tau_{desc} = 0$ and $\tau_{asc} = 1$ would led to discarding 85% of total matches while maintaining a low average WER of 0.183 (excluding OOV words).

Table 3. Effect of pruning based on number of ascenders.

τ_{asc}	$\frac{\sharp pruned\ pairs}{\sharp total\ pairs}$ [%]	WER (excluding OOV words)
0	73%	0.204
1	**30%**	**0.164**
2	9%	0.165
...	...	...
∞	0%	0.165

In summary, the relatively minor adaptation of the matching algorithm and the incorporation of the pruning technique could reduce the average time required to recognize a single word to $0.6s$ (for the particular size of the annotated collection) whilst maintaining a low average WER. This corresponds roughly to a 70 fold increase in speed compared with the original approach.

5 Future work

An obvious improvement to further reduce WER would be the optimization of the similarity measure using the framework proposed in [10] in conjunction with a suitable training collection of handwritten words. Furthermore, currently all contours from the database are re-scaled to a predefined size prior to the extraction of the MCC representations. In the case of word matching, the x-height should be sufficient for size alignment. Replacing the size invariance property with x-height invariance could further improve discrimination between words. Incorporation of additional features in order to capture the "inner" structure of a word and dots, e.g. size and centroid of the inner holes [20], should also be investigated. The number of contour matchings necessary to classify a single word could be further reduced by developing an appropriate indexing strategy for a training dictionary. We would like to investigate the possibility of reducing WER caused by OOV words by augmenting the training dictionary by synthesizing new word contours based on the initial training set. In theory, this could significantly reduce OOV errors. Finally, it should be stressed that word matching is just a first step in word recognition. Incorporating a statistical language model as a post-process could further improve the recognition accuracy of the system as it was shown in [1].

In the future we plan to incorporate this algorithm into a complete indexing system for large collections of handwritten historical documents, such as those used for experimentation purposes here, but also illuminated Gaelic manuscripts. Furthermore, in the latter case, a specific challenge we intend to investigate is the differentiation of multiple scribes within a single document based on characterizing the shapes of words.

6 Conclusions

In this article, a new method for word recognition for historical manuscripts has been proposed. Extensive experimentation conducted using the George Washington collection shows

that systems based on word contour matching can significantly outperform existing techniques. Specifically, it has been shown that on a set of twenty pages, the average recognition accuracy was 83%. Adaptation of the contour matching to the specific task of word recognition in order to improve performance and reduce computational load together with a simple pruning technique were also discussed. Moreover, preliminary investigation of potential simplifications allow us to speculate that additional development would further improve the performance.

Acknowledgements. We would like to thank the Center for Intelligent Information Retrieval from Univ. of Massachusetts for providing the annotated collection of George Washingtons manuscripts.

References

1. V. Lavrenko, T. Rath, and R. Manmatha, "Holistic word recognition for handwritten historical documents," in *Proceedings of Document Image Analysis for Libraries (DIAL)*, 2004, pp. 278–287.
2. Tomasz Adamek and Noel E. O'Connor, "A multi-scale representation method for non-rigid shapes with a single closed contour," *Special Issue on Audio and Video Analysis for Multimedia Interactive Services in IEEE Transactions on Circuits and Systems for Video Technology*, 5 2004.
3. T. Rath and R. Manmatha, "Features for word spotting in historical manuscripts," in *ICDAR 03 conference*, 2003, vol. 1, pp. 218–222.
4. T. Rath and R. Manmatha, "Word image matching using dynamic time warping," in *CVPR-03 conference*, 2003, vol. 2, pp. 521–527.
5. S. M. Harding, W. B. Croft, and C. Weir, "Probabilistic retrieval of ocr degraded text using n-grams," in *Proc. of the 1st European Conference on Research and Advanced Technology for Digital Libraries. Pisa, Italy*, September 1997, pp. 345–359.
6. R. Manmatha and N. Srimal, "Scale space technique for word segmentation in handwritten manuscripts," in *Proc. 2nd Intl Conf. on Scale-Space Theories in Computer Vision, Corfu, Greece*, September 1999, p. 2233.
7. W. Niblack, *An Introduction to Digital Image Processing*, Englewood Cliffs, N.J.: Prentice Hall, 1986.
8. J. Sauvola, T. Seppänen, S. Haapakoski, and M. Pietikäinen, "Adaptive document binarization," in *International Conference on Document Analysis and Recognition*, 1997, vol. 1, pp. 147–152.
9. L. Vincent, "Morphological grayscale reconstruction in image analysis: Applications and efficient algorithms," *IEEE Transactions on Image Processing*, vol. 2, no. 2, pp. 176–201, April 1993.
10. T. Adamek, N. E. O'Connor, and N. Murphy, "Multi-scale representation and optimal matching of non-rigid shapes," in *submitted to CBMI 2005*.
11. C. M. Myers, L. R. Rabiner, and A. E. Rosenberg, "Performance tradeoff in dynamic time warping algorithms for isolated word recognition," *IEEE Trans. Acoust. Speech Signal Processing*, vol. 28, no. 12, 1980.
12. S. Jeannin and M. Bober, "Description of core experiments for mpeg-7 motion/shape," MPEG-7, ISO/IEC/JTC1/SC29/WG11/MPEG99/N2690, Seoul, March 1999.
13. F. Mokhtarian and A. K. Mackworth, "A theory of multiscale, curvature-based shape representation for planar curves," *IEEE transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-14, no. 8, pp. 789–805, August 1992.
14. Longin Jan Latecki, Rolf Lakämper, and Urlich Eckhardt, "Shape descriptors for non-rigid shapes with a single closed contour," *IEEE Conf. On Computer Vision and Pattern Recognition (CVPR)*, pp. 424–429, 2000.
15. "Iso/iec mpeg7 experimentation model (xm software)," www.lis.ei.tum.de/research/bv/topics/mmdb/e_mpeg7.html.
16. R. Manmatha and W. B. Croft, "Word spotting: Indexing handwritten archives," in *Intelligent Multi-media Information Retrieval Collection, M. Maybury (ed.), AAAI/MIT Press*, 1997.
17. S. Kane, A. Lehman, and E. Partridge, "Indexing george washingtons handwritten manuscripts," Tech. Rep., Technical Report MM-34, Center for Intelligent Information Retrieval, University of Massachusetts Amherst, 2001.
18. T. Adamek and N. O'Connor, "Efficient contour-based shape representation and matching," in *MIR 2003 - 5th ACM SIGMM International Workshop on Multimedia Information Retrieval, Berkeley, CA*, November 2003.
19. L.K. Huang and M.J.J. Wang, "Efficient shape matching through model-based shape recognition," *Pattern Recognition*, vol. 29, no. 2, pp. 207–215, 1996.
20. Dahai Cheng and Hong Yan, "Recognition of handwritten digits based on contour information.," *Pattern Recognition*, vol. 31, no. 3, pp. 235–255, 1998.

Fig. 9. The intermediate results during the contour extraction process (columns in order from left to right): input gray-scale image, “ink”/“paper” identification by applying dynamic threshold, localization of the main body of lower case letters and removal of margin line residua, connected components alg.(gray levels) and best connecting links (red lines), binary mask, word contour.