Character Recognition under Severe Perspective Distortion

Peng Zhou, Linlin Li, Chew Lim Tan
School of Computing, National University of Singapore
zhoupen1,lilinlin,tancl@comp.nus.edu.sg

Abstract

Perspective deformation is one of the main issues needed to be addressed in real-scene character recognition. An effective recognition approach, which is able to handle severe perspective deformation, is to employ Cross Ratio Spectrum and Dynamic Time Warping techniques. However, this solution suffers from a time complexity of $O(n^4)$. In this paper, a clustering based indexing method is proposed to index cross ratio spectra and thus expedite the recognition. Cross ratio spectra of all templates are clustered. A query is compared with the centroid of each cluster instead of spectra of all templates. Our method is 40 times faster than the previous method, and has archived about 15-time speed up while preserving almost the same recognition accuracy in the real scene character recognition experiment.

1 Introduction

A common problem encountered in real-scene character recognition is the perspective distortion of characters. Because it is impossible to make the text plane (such as signboards), on which the text appears, always parallel to the image plane. A popular approach[2] [8] [12][10] to address the issue is to rectify the distorted image into a frontal-parallel view first, and then to apply OCR program on the rectified images. However, these works put constraints either on the perspective angle (such as weak perspective) or the contextual information(such as there should be at lest a text line) , and thus they only work when those particular assumptions hold. There were also a few works[9] [7] to recognize individual characters under perspective distortion directly using perspective invariants, without any rectification. In Lu's method [9], it is still necessary to estimate the direction of the text line in order to estimate ascender/descender structures, and thus this method suffers from the same problem as those methods above. Our earlier method[7], makes no assumption on either perspective angle or context information. However, the algorithm has a time complexity of $O(n^4)$, which hampers the feasibility to integrate the method into a real recognition system. In particular, a shape description called Cross Ratio Spectrum, which is similar to a time-series, is proposed in [7]. In a spectrum, the perspective effect can be modeled as an one-dimensional uneven stretching, and thus Dynamic Time Warping(DTW) algorithm is employed to compare spectra. Because the recognition process consists of many DTW comparisons,which has a quadratic time complexity, this leads to a very slow execution speed.

Therefore, a clustering-based indexing method is proposed in this paper in order to expediting the real-scene character recognition process making use of cross ratio spectrum. Also, a prototype of real-scene character recognition system developed by us will be introduced, which is more than 40 times faster than the method presented in [7].

2 Related Works

Expediting time series comparison has been extensively studied, and many speed optimization techniques have been proposed, falling into three categories:

Constraints: This limits the number of cells that evaluated in the cost matrix [4][13]. Global constraints will slightly speed up the DTW comparison, and more importantly, it will prevent over-fitting, where a small section of one spectrum maps onto a large section of another. This has already been employed in [7].

Data abstraction: This performs DTW on a reduced representation of time series [6]. In particular, a very important algorithm towards faster DTW algorithm based on iterative data abstraction, named FastDTW, was proposed in [14]. It is an accurate approximation of DTW, which has a linear time and space complexity. However, when the length of the time series is less than 1000, the running time is almost the same as a normal DTW algorithm. Unfortunately, a cross ratio spectrum normally has a length less than 100 points.

Indexing: This reduces the number of candidate templates. Indexing time-series aims to reduce the number of times to conduct DTW. An survey about data indexing and retrieval in time series databases is presented in [5]. The

978-0-7695-3725-2/09 $25.00 © 2009 IEEE
DOI 10.1109/ICDAR.2009.86

IEEE computer society

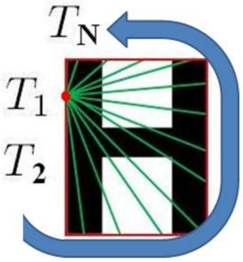


Figure 1. Cross Ratio Spectrum.

	Q_1	...	Q_m	Q_1	...	Q_M
T_N						
...						
T_1						

Table 1. DTW-distance-table.

dominant approach of indexing time series is based on a lower-bound technique. Lower-bound function provides an estimated minimum distance between two time series, to efficiently prune undesirable time series. Two important lower-bound functions are proposed in [3] and [15]. However, both of them give a very loose bound for cross ratio spectra, and make no optimization of the speed when applied.

3 Method

3.1 Cross Ratio Spectrum

In this subsection, the character representation and comparison presented in [7] will be introduced briefly. Suppose M points $\{Q_1, ..., Q_M\}$ are sampled in a equal-distance manner along the convex hull of the query character Q, and N points $\{T_1, ..., T_N\}$ are sampled along the convex hull of a template character T. The cross ratio spectrum of a point, denoted by $CRS(.)$, is a sequence of cross ratio values. An example is shown in figure 6, the cross ratio spectrum of T_1 is the sequence of cross ratios, which calculates from T_1 and $T_j, j = 2 : N$. Cross ratio spectrum is the similar to time series. The distance between Q and T is estimated as follows. First, the DTW distance between each pair of $CRS(Q_i)$ and $CRS(T_j)$ is calculated and stored in a DTW-distance-table as shown in table 1. Then, each time, a DTW is applied to a sub-table comprising of column$\{\hbar, \hbar+1, ..., \hbar+M-1\}$ of the global table, to align T_1 with $Q_\hbar$ and T_n with $Q_{\hbar+M-1}$ as the initial condition. M DTW comparisons are conducted, and thus M distances are gotten. Among M DTW distances, the smallest one gives the desirable global distance between Q and T.

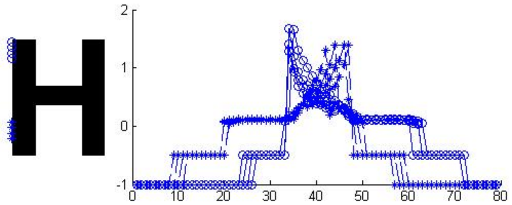


Figure 2. Neighbouring points have similar spectra.

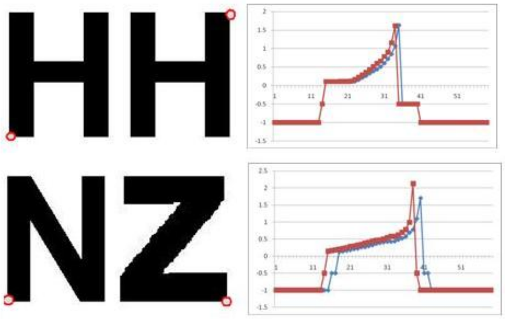


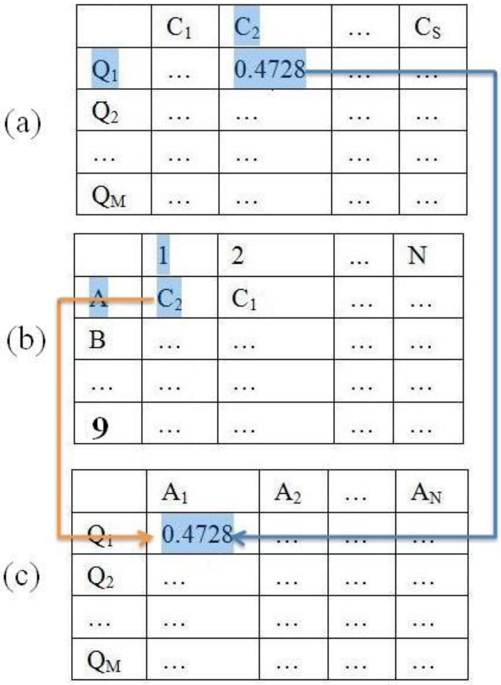
Figure 3. Points have similar spectra.

3.2 Character Recognition

The main drawback of the proposed method is the speed. DTW itself is an expensive operation due to its nature. Constructing the DTW-distance-table takes $M \times N$ DTW comparisons and each DTW comparison has a time complexity of $O(M \times N)$. The time complexity of comparing one query against one template is then $O(M^2 \times N^2)$. Hence, we aim to reduce the number of DTW comparisons so as to decrease the total recognition time. In the rest of this section, a clustering based optimization is explained in detail.

The intuition is based on the observation that there might be many sample points with a similar spectrum. For example, in figure 2, two groups of neighbouring points labeled $\star$ and $\circ$ are selected, and their spectra are drawn. We could observe that neighbouring points do have a similar spectrum pattern. By shifting them along x-axis, some of the spectra can almost match the others. Not only the neighbouring points possess this property, we also noticed that some of the cross-ratio spectra are similar to a large extent even if they are not neighbours or not from the same image. For example in figure 3, for character image "H", the top-right corner's cross ratio spectrum is almost the same as the bottom left corner's; the bottom left corner of image "N" has a similar spectrum with the bottom right corner of image "Z", etc. Hence, we come with the idea of clustering all the similar template sample points.

K-means clustering algorithm was selected because it can specify the number of clusters we would like to have.



(a)

	C₁	C₂	…	Cs
Q₁	…	0.4728	…	…
Q₂	…	…	…	…
…	…	…	…	…
QM	…	…	…	…

(b)

	1	2	…	N
A	C₂	C₁	…	…
B	…	…	…	…
…	…	…	…	…
9	…	…	…	…

(c)

	A₁	A₂	…	AN
Q₁	0.4728	…	…	…
Q₂	…	…	…	…
…	…	…	…	…
QM	…	…	…	…

Figure 4. (a)Temporary table (b)Cluster-Index-Table (c)DTW-distance-table.

After importing all the CRS information of template images, DTW is performed between each pair of CRS to determine their mutual distance. This is an expensive operation too, but since it is a one-time operation and can be treated as the training process, its complexity does not affect our actual recognition speed. The centroids of each cluster were chosen to be the one with minimum DTW distance to the rest of spectra in the cluster.

Suppose the number of clusters is S, and we denote the centroid of clusters as $\{C_s, s = 1 : S\}$; the number of templates is R, and we denote the templates as $\{T_r, r = 1 : R\}$; remember that the number of sample points for a template image is N and for a query image is M. During training, a Cluster-Index-Table is built with dimension of $(N \times R)$ as shown in figure 4 (b). For example, for template “A”, its 1st sample point falls into cluster C_2, 2nd sample point falls into cluster C_1, etc. When a query comes in, the first step is to calculate the DTW distances between points of the query $\{Q_i, i = 1 : M\}$ and clusters $\{C_s, s = 1 : S\}$. The results are stored in a temporary table as shown in figure 4(a). For instance, the DTW distance between Q_1 and C_2 is 0.4728. When a query is evaluated, a DTW-distance-table must be built, such as figure 4(c). This time, the table can be filled directly by reading results from the Cluster-Index-Table and the temporary table. For instance, if we wish to get the DTW between Q_1 and A_1, the Cluster-Index-Table will tell us that A_1 belongs to cluster C_2, and then by using DTW-result-table, the DTW comparison result of Q_1 and C_2 can be retrieved directly and filled into the corresponding cell of DTW-distance-table. Originally, if each query is compared with R templates, each time a DTW-distance-table is formed by performing $(N \times M)$ DTW comparisons, followed by DTW comparisons for M sub-tables. After clustering, $(M \times S)$ DTW comparisons are needed to construct the DTW-result-table, and for each template, M comparisons are to be made without any other overheads. Thus, the number of DTW operations needed is reduced drastically from $R \times M \times (N + 1)$ to $(S + R) \times M$.

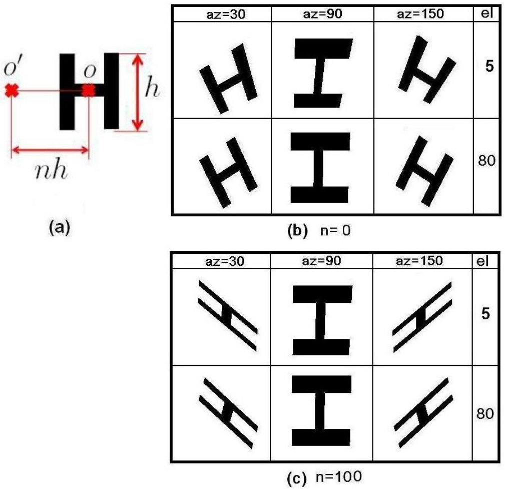


Figure 5. Samples of synthetic character image.

4 Experiment

Two experiments were designed and conducted in order to examine the proposed method. The first experiment aims to show how the number of clusters will affect the speed and accuracy. The second experiment is to show the speed improvement by clustering.

A template set Γ was trained on synthetic fronto-parallel images of 62 characters, namely 26 English characters in both upper and lower case, as well as 10 Arabic digits. 12 datasets were generated using matlab with various perspective parameters. Both the templates and the datasets are of Arial font and bold style. Since ‘I’, and ’l’ have exactly the same cross ratio sequence, they were considered the same in our experiment. The perspective images were generated by setting the target point at a specific point o', and setting the perspective viewing angle as 25° (to model a general camera lens), while changing the azimuth (az) and elevation (el) angles gradually. o' is at the same horizontal line

Table 2. Average recognition speed and accuracy per query.

Clusters	140	120	100	80	60
Time(s)	2.88	2.23	1.58	1.13	0.72
Accuracy(%)	93.81	92.20	91.67	76.61	55.10

as the center of a character, denoted by o, with a distance of $n \times h$, where n is a positive integer and h is the height of the character. An illustration is shown in figure 5(a). Generally, the larger n is, the greater the deformation is. For each testing set, n and el are predefined, and az is set as $\{30°, 90°, 150°, 210°, 270°, 330°\}$ respectively. Therefore, each testing set comprises of $6 \times 62 = 372$ characters. Examples of the character 'H' under different perspective parameters are shown in figure $5(b)$ and (c).

First, we would like to examine how the selection of number of clusters will affect the recognition speed and accuracy. In this experiment, the dataset with parameters$\{n = 100, el = 10\}$ was used, which has the most severe perspective distortion. Various numbers of clusters from 60 to 140 were selected as shown in table 2. Each time, k-means clustering was performed with a specific number of clusters on the CRS of all template sample points. The centroids were saved and stored into a table so that during the actual recognition process, the system could directly read in the table and process it accordingly. Both methods were implemented in Java, and run on a PC configured with Pentium 4 CPU 3GHz, 0.99GB of RAM. 60 points were sampled on each character.

The original method took 70.43 seconds to process a query (compare against 62 templates). Table 2 shows the effect of number of clusters on both the recognition speed and accuracy. We could observe that by decreasing the number of clusters, the recognition time could be further reduced. At the same time, the accuracy has been tradeoff to a certain extent as well. We also found that speed decrease almost linearly when the number of clusters decreases. However, the accuracy drop suddenly from 100 clusters to 80 clusters, and thus in the following experiment, the number of clusters was set to be 100, which is able to expedite the recognition process up to 40 times. With 100 clusters, the total number of DTW comparisons needed is reduced to 4.2% of the original algorithm. Also, the index will largely reduce computational overhead of the original algorithm when implemented.

Table 3. Average recognition accuracy per query for the original method and our clustering based method.

(a) original method

el=	10°	30°	50°	70°
$n = 0$	97.31	97.04	100	100
$n = 50$	97.04	97.04	100	100
$n = 100$	97.04	97.04	97.84	97.84

(b) our method

el=	10°	30°	50°	70°
$n = 0$	93.27	93.81	94.35	100
$n = 50$	91.93	93.27	94.08	100
$n = 100$	91.39	93.27	91.93	93.81

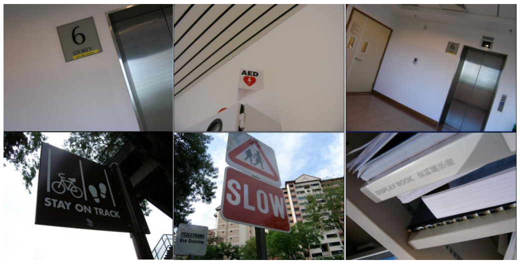


Figure 6. Examples of sign boards in real scene.

In the second experiment, all 12 testing datasets were used for both the original method and our clustering method. For clustering optimization, all the 62 templates are preprocessed by performing a k-means clustering algorithm on all the CRS of template sample points. Table 3(a) and 3(b) show the time and accuracy comparison of the two methods. We can observe that with 100 clusters to represent all the CRS patterns, the accuracy are dropped with a reasonable limit of 6 percent but the recognition speed has been improved by 40 times. Errors often occurred within characters with similar shape, like 'b' and 'q'. Another important observation is that, even though the recognition result might not be correct, it is most likely that the top 3 templates contain the answer.

5 Real-Scene Character Recognition

From the above two experiments, we have shown that by clustering, the recognition speed could be improved to a large extent, but the accuracy was compromised due to the approximation. We observed that even though the smallest DTW comparison result might not be correct, it is most likely that the top 3 templates contain the answer. Hence, we could employ a two level coarse-to-fine matching scheme in order to reduce the error. When a query Q comes in, 3 nearest templates of Q are identified by our clustering based method, and then these 3 templates are reranked by the original comparison.

Table 4. Average recognition speed and accuracy per query for real-scene character recognition.

		Time(s)	Accuracy(%)
Set I	original method	72.16	91.57
Set I	our method	4.63	90.92
Set II	original method	71.29	94.74
Set II	our method	4.64	94.74

5.1 Experiment Results

In this experiment, the template set Γ and 100 clusters built from it were used again. The testing dataset comprised of 20 sign boards which has fonts similar to Arial were used. For each signboard, 3 photos of each were taken from different angle and distance, leading to 60 photos in total. Examples of these photos are shown in figure 6. Words were extracted by the method proposed by Chen [1], and word images were binarized by an adaptive thresholding method proposed by Otsu [11]. Components smaller than 60×60 pixels were thrown away, because the binarization results of such components were often not recognisable to human. In order to guarantee that there is no error introduced by the extraction algorithm, non-character elements (here a character means either an English character or a digit) were manually eliminated in both training and testing datasets. The testing set was further divided into two sub-sets. Set I has 463 characters smaller than 100 pixels; set II has the remaining 419 characters. The recognition accuracy and speed for the original method and our method is shown in table 4. Only 3 characters were mis-recognized in set I. These 3 characters are very small, and thus the binirization result is not good.

6 Conclusion and future work

This paper proposed an clustering based cross ratio spectrum indexing method for recognition of characters under perspective distortion. This method increased the recognition speed of the original method, and hence made it more viable for real-life applications. By clustering, we achieved about 15 times speed up while preserving almost the same accuracy. This speed could be improved by code optimization. More importantly, one obstacle makes real-scene recognition for the original method difficult is that: it emphasizes an exact match between template and query, a comprehensive template set with different fonts is necessary in order to deal with the widely variance of fonts used in real scene. Our method will make this possible. Our future work will include evaluating how the indexing method will reduce a template set with more fonts.

Acknowledgment: This research is supported in part by IDM R&D grant R252-000-325-279.

References

[1] X. Chen and A. L. Yuille. Detecting and reading text in natural scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

[2] P. Clark and M. Mirmehdi. Recognizing text in real scenes. *International Journal Document Analysis and Recognition*, 4(4):243–257, 2004.

[3] K. Gollmer and C. Posten. Detection of distorted pattern using dynamic time warping algorithm and application for supervision of bioprocesses. In *Preprints of the IFAC Workshop on On-line Fault Detection and Supervision in the Chemical Process Industries*, 1995.

[4] F. Itakura. Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 23:52–72, 1975.

[5] E. Keogh and S. Kasetty. On the need for time series data mining benchmarks: a survey and empirical demonstration. In *Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 102 – 111, 2004.

[6] E. Keogh and M. Pazzani. Scaling up dynamic time warping for datamining applications. In *Proceedings of the 6th ACM International Conference on Knowledge Discovery and Data Mining*, pages 285–289, 2000.

[7] L. Li and C. Tan. Character recognition under severe perspective distortion. In *Proceedings of the 19th International Conference on Pattern Recognition*, 2008.

[8] S. Lu, B. M. Chen, and C. C. Ko. Perspective rectification of document images using fuzzy set and morphological operations. *Image and Vision Computing*, 23(5):541–553, 2005.

[9] S. Lu and C. L. Tan. Camera text recognition based on perspective invariants. In *Proceedings of the 18th International Conference on Pattern Recognition*, volume 2, pages 1042–1045, 2006.

[10] G. Myers, R. Bolles, Q. Luong, and J. Herson. Rectification and recognition of text in 3-D scenes. *International Journal Document Analysis and Recognition*, 7(2-3):147–158, 2005.

[11] N. Otsu. A threshold selection method from gray level histograms. *IEEE Transactions on Systems, Man and Cybernetics*, 9(1):62–66, 1979.

[12] M. Pilu. Extraction of illusory linear clues in perspectively skewed documents. In *Proceedings of IEEE on Computer Vision and Pattern Recognition*, volume 1, pages 363–368, 2001.

[13] H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 26:59 – 165, 1978.

[14] S. Salvador and P. Chan. Toward accurate dynamic time wrapping in linear time and space. *Intelligent Data Analysis*, 11:561–580, 2007.

[15] B. Yi, H. Jagadishand, and C. Faloutsos. Efficient retrieval of similar time sequences under time warping. In *Proceedings of 14th International Conference on Data Engineering*, pages 201–208, 1998.