Automatic Trajectory Extraction And Validation Of Scanned Handwritten Characters

Ralph Niels and Louis Vuurpijl
Nijmegen Institute for Cognition and Information
Radboud University Nijmegen
Nijmegen, The Netherlands
{r.niels;vuurpijl}@nici.ru.nl

Abstract

A well-established task in forensic writer identification is the comparison of prototypical character shapes (allographs) present in the handwriting. Using elastic matching techniques like Dynamic Time Warping (DTW), comparison results can be made that are plausible and understandable to the human expert. Since these techniques require the dynamics of the handwritten trace, the "online" dynamic allograph trajectories need to be extracted from the "offline" scanned documents. We have implemented an algorithm that can automatically extract this information from scanned images. The algorithm makes a list of all possible trajectories. Using a number of traditional techniques and DTW for evaluation, the best trajectory is selected. To be able to make a quantitative assessment of our techniques, rather than a qualitative discussion of a small number of examples, we tested the performance on two large datasets, one containing online and the other containing offline data. Two different methods (one for the online, and one for the offline dataset) are used to validate the generated trajectories. The results of the experiments show that DTW can significantly improve the performance of trajectory extraction when compared to traditional techniques.

Keywords: Trajectory extraction; Forensic writer identification; Allograph matching; Dynamic Time Warping.

1. Introduction

Forensic writer identification has been enjoying new interest due to an increased need and effort to deal with problems ranging from white-collar crime to terrorist threats. Finding the identity of the writer of a document is traditionally done by forensic handwriting experts, using methodologies like the ones described by Huber and Headrick [5] and Morris [11]. Comparing a so-called "questioned document" to a large database of handwritten documents is manual labor, which typically focuses on the presence of particular allographs. If a certain set of allographs occurs in both the questioned document and in one of the documents in the database, they may have been produced by the same writer.

The elastic matching technique Dynamic Time Warping [9, 12, 20] is very suitable for automating this application, because it is able to match two allographs in a human-congruous way [14] (i.e., the comparison yields results that are visually convincing to the human observer). The main practical objection to use this technique is the fact that it uses online data to compare two allographs, when, most of the time, only scanned documents (i.e., offline data), are available. One way of solving this is by manually copy-drawing scanned images. Because this can be very time consuming, the challenge is to perform the extraction of dynamic information automatically.

Our current research, which is executed in the Trigraph project [16], which aims at developing techniques to improve the quality of writer identification systems available today. The work presented here focuses on a method to automatically extract the production order from scanned images. Our technique creates a number of possible theories about the production order, and automatically evaluates them, to yield the correct one. We implemented an algorithm for automatic trajectory extraction and a number of traditional methods for evaluating those extracted trajectories. We also explored the possibilities of using trajectory matching techniques like DTW for evaluation of the extracted trajectories. Starting point of this discussion is that for each trajectory that is extracted from a handwritten character image, there must exist a prototypical allograph that matches this trajectory. Given a proper set of prototypes, it must thus be possible to make a correct evaluation of the extracted trajectory. This approach is particularly suitable for forensic document examination, which relies heavily on the detection of particular allographs in document databases for writer identification purposes.

To test whether DTW can indeed correctly evaluate automatically extracted trajectories given a certain prototype database, and to find out whether this approach outperforms traditional methods, we conducted an experiment that is described in this paper. As is shown, the results indicate that trajectory matching can significantly improve the quality of the selection of correctly extracted trajectories. Given our findings that DTW is a technique that yields results plausible to humans [14], this method promises to be useful for forensic document examination.

In the next section, the trajectory extraction method and verification methods are described. Subsequently, the experiment we conducted to test the methods and its results are presented. This paper concludes with a discussion and future research issues.

2. Trajectory extraction

Our trajectory extraction algorithm is based on the algorithms described by Jäger [6] and Kato and Yasuhara [7]. The technique is restricted to single stroke allographs (i.e., those characters for which the pen is not lifted from the paper during writing) with identifiable begin and end points (i.e., where the begin and end points do not coincide with the stroke). Note that this differs from [7], which is also restricted to characters with junctions where no more than two lines intersect. Given a pre-segmented handwritten character image, a number of theories on the possible writing order are automatically derived as described below.

2.1. Graph representation

The algorithm uses a graph representation of the original image, generated through thinning and edge following. This is a standard approach from which our implementation differs only in details. Three steps are performed in the process:

(i) The scanned image is binarized and thinned, resulting in a skeleton image. For skeletonization of the binarized image, we employed the technique described by Huang, Wan and Liu [4].

(ii) Clusters of pixels are detected at the start or end of a stroke or at points where two or more lines intersect. A cluster is defined as a set of 8-neighboring pixels that each have either only one 8-neighbor or that have more than two 8-neighbors. Two cluster types are distinguished: (I) boundary clusters, i.e., clusters that have one connected line (these are candidates for the start and end point of the trajectory) and (II) junction clusters, i.e., clusters that have more than two connecting lines (these are clusters where two or more lines intersect). Clusters that have two connecting lines are deleted, since such cases are most probably caused by ink blobs within strokes, and do not represent positions where the writing direction was changed. Figure 1 depicts an example image, and the detected clusters.

(iii) A graph is constructed based on the detected clusters and their connecting lines. Clusters are represented by nodes, and lines by edges. See Figure 1 for an example.

2.2. Theory creation

Based on the graph representation created in the previous step, a graph traversal algorithm generates a list of theories containing possible trajectories. Theories are represented by an ordered list of edge numbers, such as the one illustrated in Figure 1. Each theory corresponds to a trajectory that is determined by following the coordinates of the pixels in the skeleton image, in the order that is represented by the theory. For a theory to be valid, the following conditions need to be satisfied:

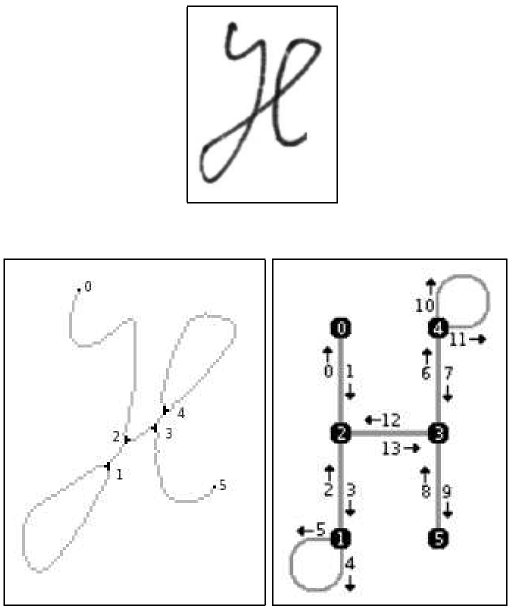

Figure 1. Graph representation: The top figure depicts the original image. The bottom left figure shows the clusters that were found in that image. The bottom right figure shows the graph representation of the image. Each node and each directional edge have unique identification numbers. Nodes 0 and 5 are boundary nodes, and all other nodes are junction nodes. The correct theory here is 1, 3, 4, 2, 13, 6, 11, 7, 9.

(i) The starting point of the first edge and the ending point of the last edge should be boundary clusters (we suppose that the starting point and ending point of the trajectory are at boundary clusters).

(ii) The theory should at least contain one of the two direction edges of each edge, to make sure that all the line segments in the image are part of the theory.

(iii) Each direction edge can only occur in a theory once, i.e., we suppose that every edge is traced no more than two times (one time in both directions).

(iv) Each edge representing a loop (i.e., connecting a node to itself) can be traced only once (combined with the second condition, this means that either one of the two directions is traced, and the other is not).

We use two different approaches to create theories. In the first one, which we call the *brute force*-approach, an exhaustive search is executed to generate every possible theory. These theories are then evaluated using four distinct evaluation methods. Using these methods, the most suitable theory and corresponding trajectory can be found. However, as argued in [7], for more complex characters and words, this exhaustive approach becomes computationally less attractive. Therefore, in the second approach, which we call the *ideal path*-approach, a preselection of theories is made during the search. Starting at a boundary cluster, at each junction, those paths that differ too much from the straightest path are not followed. This results in a decrease of the number of theories, and therefore a decrease in computational complexity. Especially when extracting dynamic information from complex characters, other alphabets, like the Tamil alphabet [13], bigrams, trigrams, or even complete words, this can reduce the amount of required computing power dramatically. The two approaches that we use are described in more detail below.

2.2.1. Brute force-approach

This method examines every possible theory. For each theory, the corresponding trajectory was evaluated by four different evaluation methods:

(i) Trajectory length: Sum of the Euclidian distances between each pair of succeeding pixels.

(ii) Average curvature: Average angle between each triplet of succeeding pixels.

(iii) Local curvature: Average curvature in the traversed junction clusters. This is calculated by concatenating the trajectory segments corresponding with the ingoing and outgoing edges at each junction cluster, limiting the result by finding the minimum and maximum y-coordinate (i.e., creating one stroke), spatially resampling the stroke to 30 points [18] to avoid the effects of curvature quantization [17], and computing the average curvature in the resampled stroke (using the method described above). The local average curvatures at the junction clusters are then averaged by dividing them by the total number of junction clusters traversed in the theory (see Figure 2).

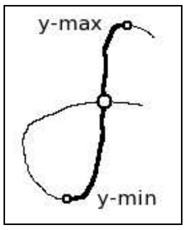

Figure 2. Example of local curvature. The local curvature of the dark trajectory segments is based on the stroke that is created by concatenating the segments, limiting the result by finding the minimum and maximum y-coordinate and spatially resampling the stroke to 30 points.

(iv) Smallest DTW distance: A trajectory is matched to the prototypes from the *Mergesamples* prototype set, which was created by semi-automatic clustering of a large database [12, 14]. The smallest distance, indicating how similar the trajectory is to the most similar prototype, is used. Note that the trajectory is only matched to those prototypes that represent the character that is also represented by the trajectory (e.g., if the sample represents an *a*, only the *a*-prototypes are considered). This restriction can be justified if one considers that in our application, forensic experts search for particular characters like "give me this particular <a>".

2.2.2. Ideal path-approach

To limit the number of possible theories, a number of suggestions are made in the literature to exploit local information. In general, these try to minimize directional changes or employ local curvature [17]. In [7], graph traversal is ruled by an algorithm that opts for the middle edge at branches, but which is therefore restricted to junctions with no more than two crossing lines. In our approach, local curvature information is employed to construct a theory. At each junction that is traversed, it is decided which edge is the best to continue with. To decide this, the local curvature between the incoming edge and each of the outgoing edges is calculated using the *local curvature*-method described in Section 2.2.1. The outgoing edge that yields the lowest local curvature is chosen. This is repeated until a valid theory is created.

3. Trajectory validation

To compare the quality of the four evaluation methods described in Section 2.2.1, we conducted an experiment in which images were offered to the algorithm. The images we used were taken from two different databases: one containing online and the other containing offline images. The *brute force*-approach was used to generate all possible trajectories. Each of those trajectories was evaluated by each evaluation method. Using validation techniques described later in this section, we decided which of the extracted trajectories was the correct one. Given this information, we were able to assess how well each evaluation method was able to select the correct trajectory.

3.1. Validation approaches

The challenge was to find a way to validate the created trajectories, i.e., to decide which of the trajectories was the correct one. A number of methods to validate extracted trajectories are mentioned in the literature. An indirect method is proposed by Lallican et al. [10]. They validated the results of their trajectory extraction algorithm by comparing the word recognition performance for the extracted trajectories and for the original online data that was available as well.

A direct validation can be performed manually, by comparing a resulting trajectory to its corresponding "ground truth". For example, Kato and Yasuhara [7] verified their results by displaying an animated pencil that follows the trajectory that has been produced by their algorithm and by using different colors to distinguish between single-traced and double-traced strokes. Boccignone et al. [1] also verified their results manually.

However, with relatively large amounts of data, visual inspection becomes a practical problem. If, on the other hand, the ground truth of each sample is available, automatic validation becomes possible. For example, if the offline and online signals were recorded simultaneously during data acquisition, both a scanned image and the trajectory actually produced are available to the system. Unfortunately it is not possible to make a direct one-to-one match between the two signals, as is described by Franke [2]: When superimposing online pen trajectories and offline ink traces, variations in pen-tilt and pen-azimuth, which occur in human handwriting, cause different displacements in the captured online signal. This problem can be solved, however, using an elastic matching technique like DTW to create a match between the two signals, as is described below.

3.2. Method

3.2.1. Online

Two different data sets were used in the experiment [15]. The first set contained online data. We randomly selected 3370 samples that satisfied the conditions described in Section 2 from the characters in the UNIPEN v07_r01-trainset [3]. 16.5% were digits, 46.8% were lowercase and 36.6% were uppercase letters. A test data set of this size allows for a quantitative assessment of our techniques, an approach that is still fairly unknown.

Using the Bresenham line generator, we created images with a pixel width of 1 that were offered to the trajectory extraction algorithm. The binarization and thinning steps were skipped. With both the image and the ground truth available, we were able to use a method based on the one described by Jäger [6]. The trajectories extracted from the images were mapped onto the ground truth by checking whether each pixel in the extracted trajectory was also present in the ground truth and whether the order in which they occurred was the same. The trajectory in which both conditions were satisfied was marked as the correct one.

3.2.2. Offline

The second set contained offline data. We manually selected 1231 samples from the Firemaker data set [19], using a tool that allowed a handwriting expert to mark characters that satisfied the conditions described in Section 2. 6.2% of the selected samples were digits, 84.1% were lowercase and 9.7% were uppercase letters.

It is well known that binarization and thinning algorithms can introduce artefacts like the ones depicted in Figure 3. Small loops and hooks at positions where two lines cross or where ink blobs are present, can cause edges to appear in the graph that do not exist in the original sample. Because our algorithm requires that all edges are visited in each theory (see Section 2.2), samples containing such spurious edges (which the ground truth does not contain), will cause our algorithm to fail in extracting the correct trajectory. Since the focus of this paper is on the validation of trajectories, given that the correct trajectory is extracted, we decided to remove those cases in which the introduced artefacts would prohibit the algorithm to find the correct trajectory. This was done by visual inspection.

To be able to validate the extracted trajectories, we let human handwriting experts manually copy-draw the data,

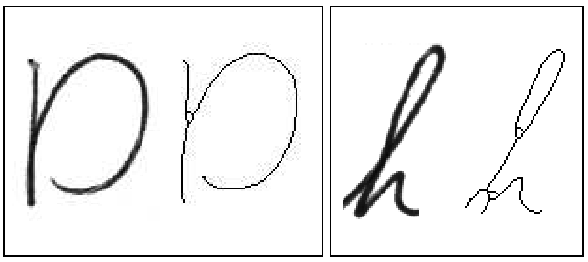
Figure 3. Examples of original images (left) and artefacts created by binarization and thinning (right). The small loops that were introduced cause the algorithm to fail in finding the correct trajectory. Samples like these were removed from the data set.

using a Wacom Cintiq 15X writing tablet (a tablet that allows the user to directly interact with a pen on the screen, and therefore create very accurate dynamic copies of the images presented). Finally, the extracted trajectories were matched to the corresponding copy-drawn trajectory using DTW. The trajectory with the lowest DTW-distance to the copy-drawn ground truth was selected as being the correct one.

The assumption underlying this method is that for each sample, the correct theory is indeed generated by the algorithm. If it is not, DTW will mark the most similar, but possibly incorrect, trajectory as being correct. To ensure that these cases would not occur, we manually inspected the data in which the DTW-distance between the ground truth and the most similar trajectory was higher than a certain threshold (i.e., if the theory marked as correct differed too much from the ground truth), or if the difference between the DTW-distances of different theories was below another threshold (i.e., if the DTW-measure was not sufficiently able to distinguish between different theories, for example in cases where small details differed). Since no problem cases were found, we conclude that the assumption is valid.

3.3. Results

From the trajectories extracted from each sample, four ranked lists, sorted on the different evaluation measures were created. Table 3.3 depicts, for each method, the relative number of samples for which the correct trajectory was found within the top-n positions in the ranking lists.

From these results, it can be concluded that DTW outperforms the other methods. The differences between the traditional methods and DTW are strongly significant for the top-1, top-2, and top-3 rankings. The relatively low top-1 performance of the other methods can be explained by the fact that when using them, the system is not able to distinguish between the same trajectory followed in two different directions, since their values are the same.

A closer examination of the cases in which DTW fails (see Figure 4) reveals that most errors are caused by mis-

Table 1. Top-n performance (the fraction of cases where the correct result is among the n best theories) of the different evaluation methods for the online ("on") and offline ("off") data set. The performance is presented for length, average curvature, local curvature, and Dynamic Time Warping.

top n	**Length**		**Average curvature**		**Local curvature**		**DTW**	
	on	off	on	off	on	off	on	off
1	72	72	73	61	75	61	86	93
2	82	74	85	64	84	76	96	98
3	92	92	90	78	89	84	96	99
4	93	92	94	79	92	90	97	99
5	96	98	95	92	94	93	98	100

sing details in the best matching prototypes, in particular the occurrence of small loops. These cases form the majority of errors and are caused by the fact that each prototype is the result of an "averaging" procedure [12]. A few errors are attributed to the occurrence of hooks at the start or beginning of a character and to samples in which the writer produced a character shape in a direction that was not covered by the prototype database. If the prototype that is most similar to a specific trajectory lacks a certain detail that is present in the trajectory, the DTW-technique may not be able to make a correct judgment about that detail. In the case of the "h" and the "n" in Figure 4, the most similar prototype does not contain loops, and DTW, therefore, cannot detect the right direction of the loops in the extracted trajectories. In the case of the "d", the allograph was started in the middle and finished at the top. However, the prototype in the database that was most similar to the image, was traced the other way. DTW was therefore not able to detect the right direction. In the case of the "l", the most similar prototype in the database was a straight line from top to bottom. The best way to match this prototype to the sample, was by starting at the top, double tracing the small "hook" on the right, and continuing to the bottom, while the allograph was actually started at the hook, after which the top piece was double traced, and the trace was continued to the bottom.

As described in Section 3.2.2, cases that caused thinning and binarization problems were removed from the offline data set. In some of removed cases, small loops, similar to those depicted in Figure 4, caused the problems. If they had been thinned correctly, some cases would probably have caused the DTW-method to fail in selecting the correct trajectory. The DTW-results mentioned should therefore be interpreted as an upper boundary. This holds particularly for the results for the offline data.

4. Conclusions

In this paper, we have argued that the use of large data sets allows for a quantitative assessment of trajectory extraction technologies, an approach that is still fairly unknown. Our results show that, when compared to traditional techniques, DTW can significantly improve the quality of trajectory extraction. However, it is well known that DTW is computationally expensive, and can only be applied to fully extracted trajectories (i.e., it cannot be used for local decisions during the trajectory extraction phase). Also, in cases where samples are not covered by the prototype database, it may select the wrong trajectory as the best one. Preliminary tests show that a combination of the *ideal path*-approach (i.e., using local curvature information to generate only a limited number of trajectories) and the DTW measure could solve both problems. Experiments to test the validity of these re-

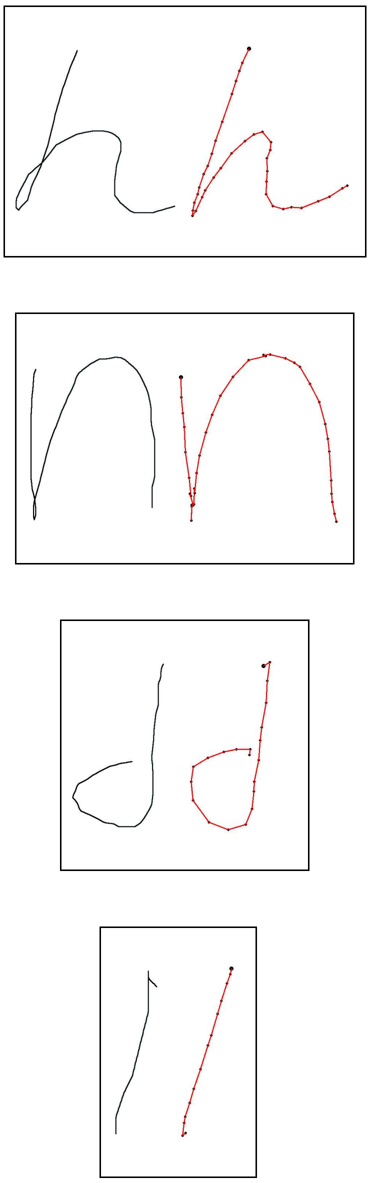

Figure 4. Examples of cases where DTW does not select the correct trajectory. The images on the left are the samples, the images on the right are the nearest prototypes according to DTW. The black dot indicates the starting point of the prototype.

sults are currently being conducted. With a lower demand for computing power, the automatic extraction of trajectories from multi stroke characters, complex characters, or even words also will become a practical possibility. Exploration of these possibilities is also part of our current research.

In the Trigraph project [16], we are currently pursuing the development of a taxonomy of the allograph shapes that are present in different handwriting databases. Knowledge of forensic handwriting experts is used to create a list of the most prominent shape categories that occur. Furthermore, automated techniques, like hierarchical clustering [21], are used to reveal statistical information about the frequency of occurrence of these shapes. This combination of top down expert knowledge and bottom up pattern recognition is expected to improve the prototype sets employed and thus improve the results of DTW as an evaluation method for extracted trajectories.

To extend the practical possibilities of our algorithm, we need to improve the quality of the binarization and thinning steps, so that artefacts causing the algorithm to fail may be eliminated. Using more advanced techniques like, for example, the piecewise linear skeletonization algorithm described by Kégl and Krzyżak [8], we hope to increase the quality of the preprocessing steps to such an extent that the system can be used in practice.

One of the advantages of using DTW is that, if a certain prototype is in the database, DTW provides an excellent basis for retrieving particular allographs that correspond to that prototype. Given the fact that DTW is able to create a human-congruous or "intuitive" match between handwritten characters [14], we can conclude that using DTW is a promising way to achieve the goal of our study: To develop techniques through which forensic experts can search for the occurrence of characters with a particular shape in a database of documents. To decrease the amount of computational power needed for DTW, we intend to employ our techniques for the batch-wise indexing of the databases that are used by DTW. Querying for the occurrence of particular allographs will then boil down to the comparison of the query characters to the set of prototypes and using the labels of the best-matching prototypes to search in the pre-indexed databases.

References

[1] G. Boccignone, A. Chianese, L. Cordella, and A. Marcelli. Recovering dynamic information from static handwriting. *Pattern Recognition*, 26(3):409–418, 2003.

[2] K. Franke. *The influence of Physical and Biomechanical Processes on the Ink Trace*. PhD thesis, University of Groningen, 2005.

[3] I. Guyon, L. Schomaker, R. Plamondon, M. Liberman, and S. Janet. UNIPEN project of on-line data exchange and recognizer benchmarks. In *Proc. ICPR'94*, pages 29–33, Oct. 1994.

[4] L. Huang, G. Wan, and C. Liu. An improved parallel thinning algorithm. In *Proc. 7th Int. Conf. Document Analysis and Recogn.*, pages 780–783, Edinburgh, 2003. IEEE Computer Society.

[5] R. Huber and A. Headrick. *Handwriting identification: facts and fundamentals*. CRC Press, Boca Raton, Florida, 1999.

[6] S. Jäger. *Recovering dynamic information from static, handwritten word images*. PhD thesis, Daimler-Benz AG Research and Technology, 1998.

[7] Y. Kato and M. Yasuhara. Recovery of drawing order from single-stroke handwriting images. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22(9):938–949, 2000.

[8] B. Kégl and A. Krzyżak. Piecewise linear skeletonization using principal curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(1):59–74, 2002.

[9] J. Kruskal and M. Liberman. The symmetric time-warping problem: from continuous to discrete. In D. Sankoff and J. Kruskal, editors, *Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparisons*. Addison-Wesley, Reading, Massachusetts, 1983.

[10] P. Lallican, C. Viard-Gaudin, and S. Knerr. From off-line to on-line handwriting recognition. In L. Schomaker and L. Vuurpijl, editors, *Proc. 7th Int. Workshop on Frontiers in Handwr. Recogn.*, pages 303–312, Amsterdam, 2000.

[11] R. Morris. *Forensic handwriting identification : fundamentals, concepts and principals*. Academic Press, San Diego, California, 2000.

[12] R. Niels. Dynamic Time Warping: An intuitive way of handwriting recognition? Master's thesis, Radboud University Nijmegen, November-December 2004. `http://dtw.noviomagum.com`.

[13] R. Niels and L. Vuurpijl. Dynamic Time Warping applied to Tamil character recognition. In *Proceedings of the 8th International Conference on Document Analysis and Recognition (ICDAR2005)*, pages 730–734, Seoul, Korea, August-September 2005.

[14] R. Niels and L. Vuurpijl. Using Dynamic Time Warping for intuitive handwriting recognition. In A. Marcellli and C. DeStefano, editors, *Advances in Graphonomics, Proceedings of the 12th Conference of the International Graphonomics Society (IGS2005)*, pages 217–221, Salerno, Italy, June 2005.

[15] R. Niels, L. Vuurpijl, and L. Schomaker. Automatic allograph matching in forensic writer identification. *International Journal of Pattern Recognition and Artificial Intelligence*. In press.

[16] R. Niels, L. Vuurpijl, and L. Schomaker. Introducing Trigraph - trimodal writer identification. In *Proc. European Network of Forensic Handwr. Experts*, Budapest, Hungary, 2005.

[17] R. Plamondon and C. Privitera. The segmentation of cursive handwriting: an approach based on off-line recovery of the motor-temporal information. *IEEE Trans. on Image Processing*, 8(1):90–91, 1999.

[18] L. Schomaker. Using stroke- or character-based self-organizing maps in the recognition of on-line, connected cursive script. *Pattern Recognition*, 26(3):443–450, 1993.

[19] L. Schomaker and L. Vuurpijl. Forensic writer identification: A benchmark data set and a comparison of two systems. Technical report, Nijmegen Institute for Cognition and Information (NICI), 2000.

[20] V. Vuori. *Adaptive Methods for On-Line Recognition of Isolated Handwritten Characters*. PhD thesis, Finnish Academies of Technology, 2002.

[21] L. Vuurpijl and L. Schomaker. Finding structure in diversity: A hierarchical clustering method for the categorization of allographs in handwriting. In *Proc. ICDAR4*, pages 387–393. IEEE Computer Society, Aug. 1997.