Continuous Dynamic Time Warping for translation-invariant curve alignment with applications to signature verification

Mario E. Munich
California Institute of Technology 136-93
mariomu@vision.caltech.edu
Pasadena, CA 91125

Pietro Perona
California Institute of Technology 136-93
perona@vision.caltech.edu
Pasadena, CA 91125

Abstract

The problem of establishing correspondence and measuring the similarity of a pair of planar curves arises in many applications in computer vision and pattern recognition. This paper presents a new method for comparing planar curves and for performing matching at sub-sampling resolution. The analysis of the algorithm as well as its structural properties are described. The performance of the new technique applied to the problem of signature verification is shown and compared with the performance of the well-known Dynamic Time Warping algorithm.

1. Introduction and Motivation

One of the research areas that is receiving a lot of attention nowadays is the area of biometric techniques for personal identification. Signature verification belongs to this set of biometric techniques. In most systems, signature verification requires the use of electronic tablets or digitizers for on-line capturing and optical scanners for off-line conversion [14]. These interfaces have the drawback that are bulky (they need to have at least the minimum area required to sign) and require the presence of dedicated hardware. Cameras, on the other hand, are much smaller and are becoming ubiquitous in the current computer environment. We have demonstrated [5] the feasibility of using a visual interface that can be built with video technology and computer vision techniques in order to capture signatures to be used for personal identification. This visual interface allows the user to write on a normal piece of paper with a normal pen, providing him/her with a more natural and comfortable environment to interact with the computer. We have presented the performance [5, 6, 7] of our visual-based signature verification system using the well-know Dynamical Time Warping technique in order to measure the similarity of the signatures.

Motivated by the application of interest, i.e. signature verification, in this paper we study the problem of performing comparison of parameterized curves. There is a number of different methods proposed in the pattern recognition literature that could be applied to this problem. Some of the methods require the extraction of a prototype that summarizes the "mean" behavior of the examples and a measure of the deviation from this prototype that exists in the training set. Some other methods compute the similarity of the curves based on a time-distortion function that best aligns the curves. A measure of this similarity is later on used for classification.

Dynamic Time Warping (DTW) is a technique that is well-suited for the latter type of mentioned methods. DTW finds for each sample in one of the curves, the correspondent sample in the other curve that is closest to the original sample using some predefined metric. Given this correspondence, it is possible to calculate a "distance" between the curves under comparison. DTW has been successfully used for signature verification [3, 4, 6, 7, 8, 9, 11, 15] in the past. However, this technique has some disadvantages. In parts of the curves where the sampling is sparse, there is a lack of resolution in the matching process due to the fact that the algorithm matches only discrete samples rather than continuous curves. One possible solution for this problem is to oversample the curves using a spline interpolation before matching them. This oversampling provides the desired resolution; however, it increases the computational cost of the matching proportionally to the square of the oversampling factor. Furthermore, it is not clear how to choose this oversampling factor in a principled way and, depending on the local constraints imposed on the algorithm, the resultant distortion function could be non-invertible. In the case of aligning several examples with a reference one, the calculation of the correspondence of all the examples with the reference would allow to obtain a prototype of the training set. However, due to the non-invertibility of the distortion function, it is difficult to obtain such prototype.

This paper describes an algorithm based on the general method of Dynamic Programming [2] that overcomes the mentioned disadvantages of DTW by using a continuous formulation. The algorithm would be allowed to find correspondence not only from sample points in one curve to sample points in the other curve but also from sample points in one curve to inter-sampling points in the other and vice-

versa (figure 3).

To our knowledge, the only existing previous work is the one of Serra and Berthod [12, 13]. They worked on matching curves or contours extracted from sequences of images or from a stereo image pairs. They also proposed a continuous dynamic programming technique in order to obtain sub-pixel matching of the contours, and, therefore, better estimation of the three dimensional structure of the scene. Our algorithm is related to the one described in [12] but it is quite different from the one presented in [13]. Both algorithms rely on the use of suitable heuristic approximations in order to keep the complexity of the algorithm under control, while, in our case, we are able to derive several properties that exploit the structure of the problem and provide the tools to decrease the spatial complexity of the algorithm.

Section 2 describes the algorithm for matching planar curves. Section 3 presents the experimental setup and the results of experiments. The final section summarizes the results and discusses future work.

2. Curve Matching using Continuous Dynamic Time Warping

2.1. A translation-invariant measure of curve similarity

Given two 2-dimensional curves $C_1 = \{P_1(t), t = 1, \cdots, T_1\}$ and $C_2 = \{P_2(t), t = 1, \cdots, T_2\}$ as in figure 1(a), and assuming that we have a warping or correspondence map $\phi = [\phi_1(t), \phi_2(t)]^T$ between C_1 and C_2, such that a point $P_1(\phi_1(t)) \in C_1$ corresponds to a point $P_2(\phi_2(t)) \in C_2$, for $t \in \{1, \cdots, T\}$. We define the following similarity measure:

$$\mathcal{D}(C_1, C_2) =$$
$$\sum_{t=2}^{T} d((P_{1(\phi_1(t-1))}, P_{2(\phi_2(t-1))}), (P_{1(\phi_1(t))}, P_{2(\phi_2(t))}))$$
$$= \sum_{t=2}^{T} \|\overrightarrow{P_{1(\phi_1(t))}P_{2(\phi_2(t))}} - \overrightarrow{P_{1(\phi_1(t-1))}P_{2(\phi_2(t-1))}}\|^2$$

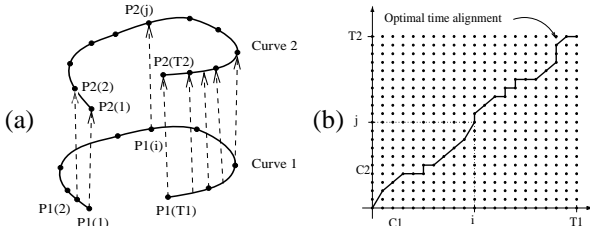where $\|.\|^2$ is the Euclidean norm.

Figure 1. *(a) Two curves C_1 and C_2 with the correspondence between points indicated. (b) Warping plane and warping path.*

It easy to see that the above distance will be invariant with respect to any translation of C_1 and C_2.

Given that we have defined the distance between curves, the actual problem to solve is to find the warping or distortion function $\phi = [\phi_1(t), \phi_2(t)]^T$ that minimizes this distance. So, the problem to solve will be the following:

$$\phi = \begin{bmatrix} \phi_1 \\ \phi_2 \end{bmatrix} = \underset{\phi}{argmin} \mathcal{D}(C_1, C_2)$$

This high dimensional minimization has in general combinatorial complexity in the number of samples. Dynamic Programming allows to perform the full minimization as a sequence of 1-dimensional minimizations in the so-called a *multi-stage decision process* [2]. Let's call $\mathcal{D}(t-1) = \mathcal{D}(\phi(t-1))$ the cumulated distance up to the $t-1$ decision stage, i.e., up top the $t-1$ matching, and $d((P_{1(\phi_1(t-1))}, P_{2(\phi_2(t-1))}), (P_{1(\phi_1(t))}, P_{2(\phi_2(t))}))$ the elementary distance added by making $P_1(\phi_1(t))$ corresponds to $P_2(\phi_2(t))$ given that $P_1(\phi_1(t-1))$ corresponds to $P_2(\phi_2(t-1))$. The solution of the above problem will be based in the following recursion:

$$\mathcal{D}(t) = \underset{\phi(t-1)}{min} \{\mathcal{D}(t-1) +$$
$$d((P_{1(\phi_1(t-1))}, P_{2(\phi_2(t-1))}), (P_{1(\phi_1(t))}, P_{2(\phi_2(t))}))\} \quad (1)$$

The matching process could be visualized on the "warping plane" of figure 1 where $\phi_1(t)$ will be represented on the x-axis and $\phi_2(t)$ will be represented on the y-axis. The correspondence function ϕ defines a path on this plane, parameterized by t, that encodes in this way the causality of the matching process. The set of sample points on C_1 and C_2 defines a grid on the warping plane. If the warping path crosses one vertex (i, j) of the grid, it means that point $P_1(i) \in C_1$ corresponds to $P_2(j) \in C_2$.

2.2. Dynamic Time Warping (DTW)

This algorithm provides the discrete solution to the above problem. The individual warping functions ϕ_1 and ϕ_2 are allowed to take values only on $\{1, \cdots, T_1\}$ and $\{1, \cdots, T_2\}$ respectively. In other words, the warping path is only allowed to go through vertices of the grid as shown in figure 2 (a). The recursion equation 1 has the following form:

$$\mathcal{D}(i, j) = \underset{(i', j')}{min} \{\mathcal{D}(i', j') + d((i', j'), (i, j))\} \quad (2)$$

where we have simplified the notation by identifying $P_1(\phi_1(t) = i)$ with the index i and $P_2(\phi_2(t) = j)$ with the index j.

From the recursion equation 2, the obtainment of the optimal path is straight forward. For each node on the discrete plane (i, j), the minimum cumulated cost is computed sequentially column-wise or row-wise. The previous node that provides the minimum cost is stored in memory. Finally, the last column or row are searched for the node with minimum cost and then the optimum warping path is found by backtracking the stored nodes.
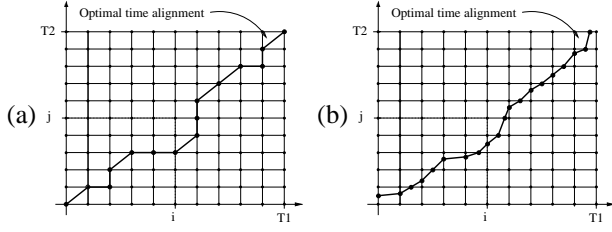
Figure 2. *(a) Warping path corresponding to a Dynamic Time Warping solution of equation 1. (b) Warping path corresponding to a Continuous Dynamic Time Warping solution of equation 1.*

For the time alignment process to be meaningful in terms of time normalization for different realizations of a signatures, some constraints on the warping functions are necessary. Unconstrained minimization in equation 2 may conceivably result in a near-perfect match between two different signatures, thus making the comparison meaningless for recognition purposes. Typical time warping constraints that are considered reasonable for time alignment include end-point constraints, monotonicity conditions, local continuity constraints, global path constraints and slope weighting (see [10] for a more extensive treatment of the subject).

In the results described in references [5, 6, 7], we enforced only a few simple constraints. First, we only allow monotonic paths to be explored, so if point i is matched with point j, then point $i+1$ can only be matched with a point after point j. Second, we only allow point (i, j) to be reached from points $(i-1, j)$, $(i-1, j-1)$ and $(i, j-1)$. Third, we require that the warping path starts at point $(0, 0)$ and ends at point (T_1, T_2). Finally, we constrain the number of points j that can be explored for each point i in minimizing equation 2. Similar constraints are used for the continuous algorithm.

2.3. Continuous Dynamic Time Warping(CDTW)

This algorithm is the continuous counter-part of DTW. We are still trying to solve the same minimization problem that gives rise to the recursion equation 1 and we are still trying to find matches for the sample points of C_1 and C_2. The only difference is that a sample point in one of the curves is allowed to match a point in-between two samples in the other curve as shown in figure 3(b). In other words, the warping path is allowed to go through points between the vertices of the grid as shown in figure 2 (b). The recursion equation will be the same as 1, with the condition that if $\phi_1(t)$ takes values on $\{1, \cdots, T_1\}$, then $\phi_2(t)$ is allowed to take non-integer values, and vice-versa.
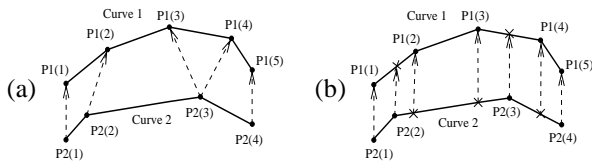

Figure 3. *(a) Matching of the two curves using DTW. (b) Matching using CDTW, where the crosses show matching points that are not samples.*

The generation of these intermediate matching points assumes a particular interpolation model for the curves. We assume a linear interpolation model between sample points since it allows us to derive the equations for the recursion in closed and simple form.

Figure 4 shows the parameterization of the curves and the notation that is used in the continuous algorithm. Arc-length parameterization is the most convenient way to describe the curves using the linear interpolation model. The equations for the coordinates of points belonging to the piece-wise linear segments of C_1 and C_2 will be the following:

$$C_1 : \begin{cases} x_1 = x_1(i-1) + r_1 \frac{\Delta_{x_1}}{\Delta_1} \\ y_1 = y_1(i-1) + r_1 \frac{\Delta_{y_1}}{\Delta_1} \end{cases} \quad C_2 : \begin{cases} x_2 = x_2(j-1) + r_2 \frac{\Delta_{x_2}}{\Delta_2} \\ y_2 = y_2(j-1) + r_2 \frac{\Delta_{y_2}}{\Delta_2} \end{cases}$$

$$\begin{aligned} \Delta_{x_1} &= x_1(i) - x_1(i-1) & \Delta_{x_2} &= x_2(j) - x_2(j-1) \\ \Delta_{y_1} &= y_1(i) - y_1(i-1) & \Delta_{y_2} &= y_2(j) - y_2(j-1) \\ \Delta_1 &= \sqrt{\Delta_{x_1}^2 + \Delta_{y_1}^2} & \Delta_2 &= \sqrt{\Delta_{x_2}^2 + \Delta_{y_2}^2} \end{aligned}$$
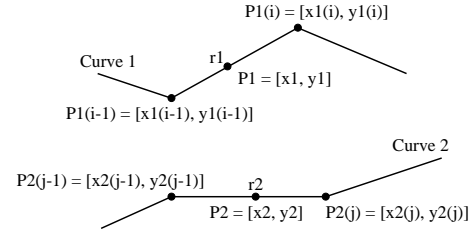

Figure 4. *Curve parameterization used in CDTW.*

2.3.1 Analysis of a single step of the algorithm

Due to the recursive nature of the dynamic programming method, a single step of the algorithm is the basic building block to be studied. This single step corresponds to a segment joining two sides of one of the square boxes of the grid imposed onto the warping plane by the samples of the curves, as shown in fig. 2(b). Figure 5 shows the four possible matching cases and the correspondent segments in the warping plane.

Using the cosine law, the elementary distance $d((P_1(\phi_1(t-1)), P_2(\phi_2(t-1))), (P_1(\phi_1(t)), P_2(\phi_2(t))))$ is calculated as follows:

c1: $\quad d(r_1, r_2) = r_2^2 + (\Delta_1 - r_1)^2 - 2r_2(\Delta_1 - r_1)\cos\theta$
c2: $\quad d(r_1, r_2) = r_1^2 + (\Delta_2 - r_2)^2 - 2r_1(\Delta_2 - r_2)\cos\theta$
c3: $\quad d(r_1', r_1) = \Delta_2^2 + (r_1 - r_1')^2 - 2\Delta_2(r_1 - r_1')\cos\theta$
c4: $\quad d(r_2', r_2) = \Delta_1^2 + (r_2 - r_2')^2 - 2\Delta_1(r_2 - r_2')\cos\theta$

$$(3)$$

where r_1' and r_2' are the corresponding values of r_1 and r_2 obtained in the previous iteration and $\cos\theta$ is the cosine of the angle defined by the vectors $\overrightarrow{P_1(i-1)P_1(i)}$ and $\overrightarrow{P_2(j-1)P_2(j)}$. We see that case 2 is the dual of case 1, and case 3 is the dual of case 4. The use of a linear interpolation in the curves gives rise to elementary distances that are quadratic in the variables of interest r_1 or r_2. Assuming that
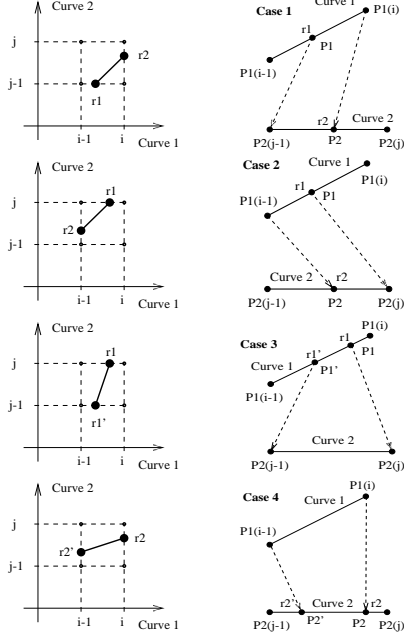
Figure 5. *Different matching cases that are possible at each step of the algorithm.*

the value of the one of this variables is given, the minimization of this elementary distance is quite simple and results in a quadratic function of this fixed variable. The recursion equation 1 takes the following form for case 1:

$$\mathcal{D}(i, r_2) = \min_{r_1} \ \{\mathcal{D}(r_1, j-1) + d(r_1, r_2)\}$$

where the $\mathcal{D}(i, r_2)$ is now a function of a continuous variable r_2 and the minimization is performed w.r.t. a continuous variable r_1. We have simplified the notation as in the case of DTW, by identifying $P_1(\phi_1(t) = i)$ with the index i, $P_1(\phi_1(t-1))$ with r_1, $P_2(\phi_2(t) = j-1)$ with the index $j-1$, and $P_2(\phi_2(t))$ with r_2. We see that, at each step in the algorithm, we have a function of a continuous variable instead of a single number, as we had in DTW. This continuous function will propagate from one step to the other via the recursion equation. Given the simple expression for the elementary distances shown in equation 3, it is easy to show using mathematical induction that the cumulated distance function $\mathcal{D}(.,.)$ is a quadratic function of the continuous variable of interest. In fact, the cumulated distance is a quadratic function for the first step of the induction as shown in equation 3. Assuming that we are working in case 1 (the study of the other ones is very similar) and that the cumulated distance at step k is a quadratic function of r_1, then the cumulated distance at step $k + 1$ is computed as follows:

$$\mathcal{D}(i, r_2) = \min_{r_1} \ \{(A_1 r_1^2 + 2B_1 r_1 + C_1) + r_2^2 +$$
$$(\Delta_1 - r_1)^2 - 2r_2(\Delta_1 - r_1)\cos\theta\} = A_2 r_2^2 + 2B_2 r_2 + C_2$$

$$A_2 = \frac{A_1 + 1 - \cos^2\theta}{A_1 + 1} \qquad B_2 = -\frac{\cos\theta(A_1\Delta_1 + B_1)}{A_1 + 1}$$
$$C_2 = C_1 + \Delta_1^2 - \frac{(\Delta_1 - B_1)^2}{A_1 + 1} \quad r_2 = \frac{-r_1\cos\theta + (\Delta_1 - B_1)}{A_1 + 1}$$

(4)

The above equations show how the cumulated distance function propagates through the warping plane. At each step, there is a linear relationship between the corresponding curvilinear coordinates r_1 and r_2 that makes the back-propagation of the optimal path through the warping plane very simple. It can be shown that the value of coefficient A has the property that $0 \leq A \leq 1$.

Consider the square box corresponding to point (i, j) in the grid. Assume the sides of the box corresponding to (i, r_2) or (r_1, j) to be the "output" sides and the remaining two sides to be the "input" sides of the box (this assumption makes sense since we explore the warping from left to right and from bottom to top). We observe that for each square box in the grid, there are 4 different cumulated distance functions, each of them corresponding to each of the 4 possible cases. Then, for each output side we have two cumulated distance functions corresponding to each of the input sides. The result is that each step of the algorithm **doubles** the number of distances functions associated to each side of the square box.

2.3.2 Complexity of the algorithm

In the previous section, we observe that there is a combinatorial explosion in the number of cumulated distance functions to be stored at each step of the algorithm and, therefore, the spatial complexity of the algorithm grows combinatorially. Serra and Berthod [12, 13] proposed the use of heuristic constraints in order to keep the complexity under control. They also divided the range of excursion of $r_{1,2}$ into a set of intervals, each of them corresponding to a particular distance function that is the minimum of all distance functions for this interval. This division of the range of $r_{1,2}$ in intervals is questionable because it is based on the assumption that the quadratic functions will keep their intersection points in correspondence as the algorithm proceeds. In fig. 6 we show a counter-example to this assumption. What goes wrong is that each distance function has different propagation equations as shown in equation 4 and, therefore, the intersection point of the two parabolas at one iteration does not correspond to the intersection point of the two parabolas after propagation.
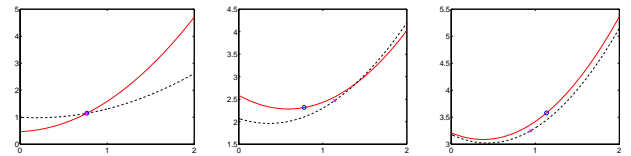

Figure 6. *Propagation of two quadratic distance functions through 2 iterations of the algorithm. We observe that the position of the intersection of the parabolas at the initial condition corresponds to two different points after propagating the parabolas.*

From figure 6, we observe that the relative position and shapes of the parabolas change from one iteration to the next. Therefore, the properties that allow us to discard a particular parabola in comparison with another parabola, are

the ones that state a relationship between the parabolas that is preserved through the iterations.

Given two parabolas, we study their relationship by looking at their difference function and the propagation of this function through the computation. The following properties are stated considering case 1 and similar properties could be derived for the other cases. Let $\mathcal{D}_{b_1}(r_1) = A_{b_1} r_1^2 + 2B_{b_1} r_1 + C_{b_1}$ and $\mathcal{D}_{b_2}(r_1) = A_{b_2} r_1^2 + 2B_{b_2} r_1 + C_{b_2}$ be two cumulate distance functions at a certain iteration of the algorithm. Let's call $\mathcal{D}_{a_1}(r_2) = A_{a_1} r_2^2 + 2B_{a_1} r_2 + C_{a_1}$ and $\mathcal{D}_{a_2}(r_2) = A_{a_2} r_2^2 + 2B_{a_2} r_2 + C_{a_2}$ the correspondent distance functions after propagation. The subindices b and a stands for *before* and *after* propagation of the distance functions. Let's call $f_b(r_1) = \mathcal{D}_{b_1}(r_1) - \mathcal{D}_{b_2}(r_1) = \Delta A_b r_1^2 + 2\Delta B_b r_1 + \Delta C_b$ and $f_a(r_2) = \mathcal{D}_{a_1}(r_2) - \mathcal{D}_{a_2}(r_2) = \Delta A_a r_2^2 + 2\Delta B_a r_2 + \Delta C_a$ the corresponding difference functions, then the following properties hold:

Property:

$$
\begin{aligned}
\Delta A_a &= \frac{\Delta A_b \cos^2 \theta}{(1+A_{b_1})(1+A_{b_2})} \\
(\Delta B_a{}^2 - \Delta A_a \Delta C_a) &= \frac{(\Delta B_b{}^2 - \Delta A_b \Delta C_b)\cos^2 \theta}{(1+A_{b_1})(1+A_{b_2})}
\end{aligned}
$$

The proof is conceptually simple and is omitted for lack of space.

Corollaries:

1. If two parabolas intersect at a certain iteration in the algorithm, then they will intersect at all further iterations.

2. If two parabolas do not intersect at a certain iteration in the algorithm, then they will not intersect at any further iterations.

3. If $\mathcal{D}_{b_1}(r_1) > \mathcal{D}_{b_2}(r_1)$ for all r_1, then $\mathcal{D}_{a_1}(r_2) > \mathcal{D}_{a_2}(r_2)$ for all r_2.

The proof of corollaries 1 and 2 follows directly from the second part of the above property and from the observation that $\cos^2 \theta \leq 1$ and $0 \leq A \leq 1$ for all cases. The proof of corollary 3 follows by the fact that ΔA keeps its sign after propagation as seen from first part of the above property, so the parabola $f_b(r_1)$ will have its opening pointing upwards and will not intersect the horizontal axis after the propagation.

The above properties allow us to compare all the quadratic distance functions pairwise and discard the ones that are "minimized" by another parabola for all the real line. This properties provides the tool to break down the combinatorial explosion of the spatial complexity of the algorithm; however, the computational cost is increased because we need to perform $o(N * (N-1)/2)$ comparisons at each step in the algorithm, where N is the number of parabolic functions that need to be compared with each other. Whether there exists a theoretical bound on the average of the maximum number of parabolic functions that need to be stored for a given set of curves is still an open research area. However, we found experimentally that this bound exists and it is a function of the number of samples

in each of the curves and the constraints imposed on the warping plane.

3. Experiments

We evaluate the performance of our continuous dynamic time warping algorithm (CDTW) in comparison with DTW with and without oversampling. We use synthetic data as well as real signatures in order to perform the comparison. The system used to acquire the signatures have been described in references [5, 6, 7]. Figure 7 shows a block diagram of the system and the experimental setup.
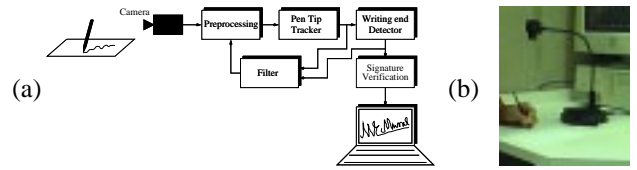

Figure 7. *(a) Block Diagram of the signature acquisition system. The camera feeds a sequence of images to the preprocessing stage. This block initializes the algorithm and selects the template to perform the tracking of the pen tip. The tip tracker obtains the position of the pen tip in each image of the sequence. The filter predicts the position of the pen tip in the next image. Finally, the last block of our system performs signature verification. (b) Experimental setup. The camera is looking at a the pen tip while the user is signing on a piece of paper.*

3.1. Experiment 1: Comparison of CDTW with DTW with and without oversampling for the case of synthetic data

Figure 8 shows the matching results for the three algorithms under comparison. The first plot corresponds to DTW without oversampling, the second plot corresponds to DTW with oversampling of the more coarsely sampled curve with a factor such that both curves have similar number of samples, the third plot corresponds to CDTW, and the fourth plot shows the warping functions corresponding to each of the algorithms. We observe that the the warping path corresponding to the continuous case is smoother than the warping path obtained with DTW. The case of DTW with oversampling provides a reasonable result in terms of matching, even though the correspondence map is still not invertible. This problem appears because the re-sampling of the curve is uniform and independent of the position of the samples of the other curve, therefore, many new samples may be allocated in a region in which the other curve has few samples. The continuous algorithm instead, by its very nature, adapts to the number of samples in each of the curves, providing in this way a better matching between the two curves.
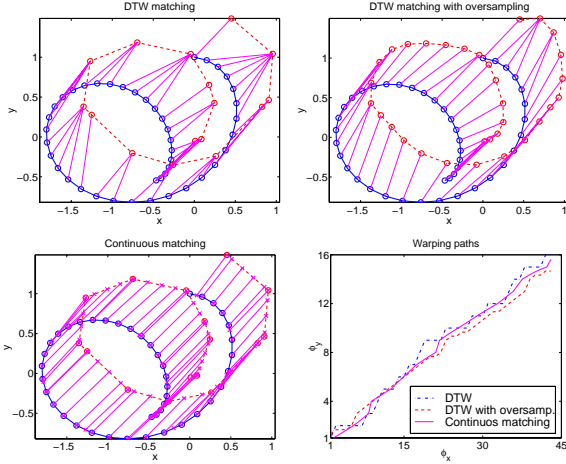
Figure 8. *The first 3 plots shows the results of DTW, DTW with re-sampling and CDTW applied to a synthetic pair of curves. The last plot display the warping plane and the corresponding warping paths for each of the algorithms.*

3.2. Experiment 2: Comparison of the algorithm with DTW for the case of signatures

Figure 9 shows the matching results using DTW and CDTW for two signatures in the data set. Both algorithms seem to perform in a similar way, although they have a big difference in the warping path. The path corresponding to DTW saturates at some point due to the discrete nature of the matching while the continuous algorithm manages to give a reasonably smoother result. We also plot the corresponding path obtained for the case in which the perform DTW after having re-sampled the two signatures by a factor of 5, this last warping path is much more similar to the linear warping than the other two as expected when increasing the number of samples.
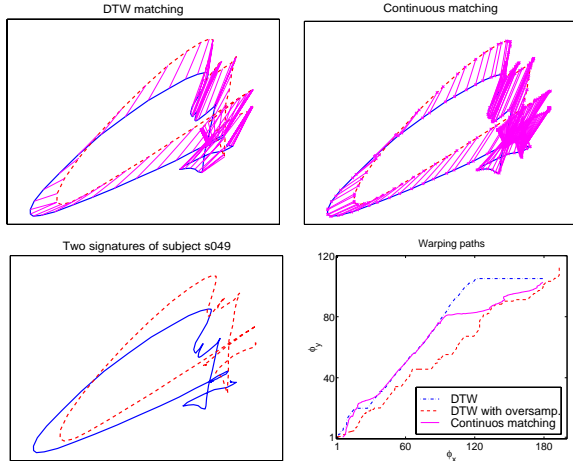

Figure 9. *Signature matching using DTW and CDTW. The first two plots show the correspondence provided by the matching. The third plot shows the signatures and the fourth plot shows the corresponding warping paths and the warping path for DTW applied on a re-sampled version of the signatures.*

3.3. Experiment 3: Experimental evaluation of the computational cost

In this experiment we evaluated the computational performance of CDTW in comparison with DTW with and without oversampling. We took a couple of examples from each of the subjects in the database and we proceed to align them with each of the three methods, for four different values of the maximum deviation from linear warping in the warping plane. For the oversampling case, we re-sampled the signatures by a factor of five. We measure the time required to for each algorithm to compute as well as the maximum storage required for CDTW. Figure 10 shows the result of this experiment. The first column corresponds to computational time measurements of the three algorithms and the second column corresponds to maximum storage of CDTW.
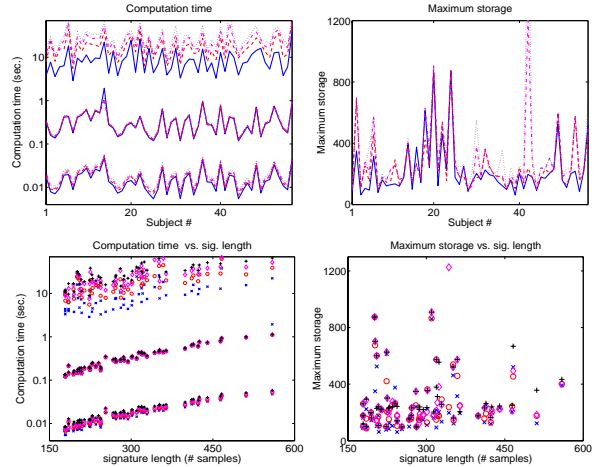

Figure 10. *The first column displays the results of the time required to perform the computation for each of the three algorithms and for each of the four cases of the warping plane constraint. The first plot shows the the value of the computation time for each of the subjects. The lower curves in the plot corresponds to DTW without oversampling, the middle curves corresponds to DTW with oversampling and the upper curves corresponds to CDTW. The different curves within each group represent the different warping plane constraints. The second plot in the first column shows the same time computation as a function of the length of each of the signatures used in the alignment. We observe that the CDTW is three orders of magnitude slower than DTW, but, if oversampling of the signatures is used, the computational time becomes similar. We observe that the time required by the three algorithms has a roughly linear relationship with the length of the sequences, in semilogarithmic space, all of them with similar slopes. However, for CDTW, this linear relationship depends on the warping constraint used, i.e., it depends on the number of nodes needed to be visited in the warping plane. The second column shows the maximum storage required by CDTW for each of the subjects and as a function the signature length. We observe that the average maximum storage required by the algorithm is roughly 200 distance functions. We also see that the maximum storage does not diverges as a function of neither the signature length nor the warping constraints, showing in this way that the property that we described previously is powerful enough to keep the spatial complexity of the algorithm bounded.*

3.4. Experiment 4: Application to signature verification

We used for this experiment the same data set that we used in reference [7]. It consists of signatures from 56 subjects, 18 of them were women and 4 were left handed. Each of them was asked to provide 25 signatures, 10 of them to be used as the training set and the other 15 to be used as the test set. The data was collected in three sessions that took place in different days in order to get a sample of the variability of the subject's signatures while avoiding the distortion produced by the boredom of the repetitive task of signing. We also asked a few of the signers to provide forgeries for each of the subjects in the database, as the ones shown in figure 11.

There are two different errors that characterize the performance of the algorithm. The Type I error (or False Rejection Rate (FRR)), measures the number of true signatures classified as forgeries as a function of the classification threshold. The Type II error (or False Acceptance Rate (FAR)), evaluates the number of false signatures classified as real ones as a function of the classification threshold. The test set allows us to compute the FRR. We computed the FAR in two different ways. First, we used all the signatures from the other subjects as *random forgeries*, and second, we used the acquired forgeries.

During training the system must *learn* a representation of the training set that will yield minimum generalization error. The algorithm provides the optimal alignment of two signatures, so we perform pairwise alignment between all elements in the training set. The signature that yields minimum alignment cost with all the remaining ones is chosen to perform the final matching. All signatures are placed in correspondence with this particular one. The prototype that represents the training set is computed as the mean of the aligned signatures. The individual costs of aligning each of the signatures in the training set with this reference signature are collected in order to estimate the statistics of the alignment process. This statistics is subsequently used for classification. In figure 11 we show several examples of signatures collected for our database, their corresponding training reference and one of the forgeries provided by other subject. We observe that the prototype obtained with DTW is much noisier than the one obtained with CDTW, due to the fact that DTW computes the prototype only with the given discrete samples while CDTW calculates the reference signature with inter-sample points.

As we stated before, we used a test set of 15 signatures for computing the FRR and all the other signatures from other subjects or the forgeries, for computing the FAR, both of them as a function of the classification threshold. Clearly, we can trade off one type of error for the other type of error. As an extreme example, if we accept every signature, we will have a 0% of FRR and a 100% of FAR, and if we reject every signature, we will have a 100% of FRR and a 0% of FAR. The curve of FAR as a function of FRR, using the classification threshold as a parameter, is called the
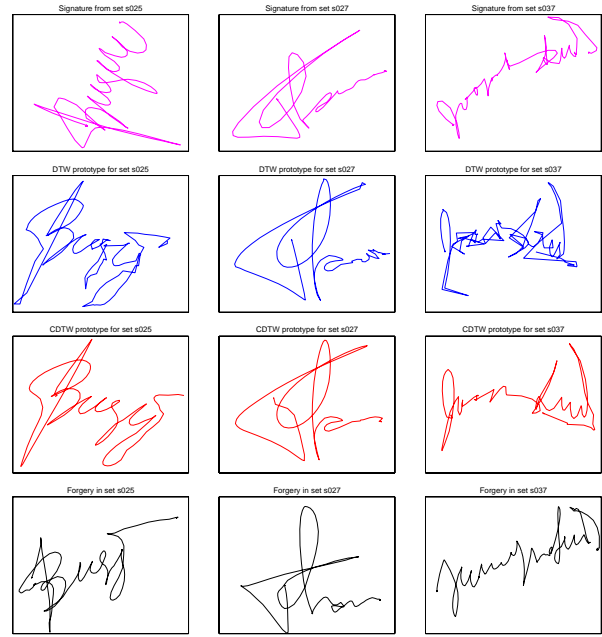

Figure 11. *Several examples of signatures in our database. On the first row we display signatures captured with the visual tracker, on the second row and third row, we show the corresponding reference signatures of the training set obtained with DTW and CDTW, and on the fourth row we display the forgeries provided by the subjects.*

error trade-off curve. It provides the behavior of the algorithm for any operation regime and it is the best descriptor of the performance of the algorithm. In practice, this curve is often characterized by the *equal error rate*, i.e., the error rate at which the percentage of false accepts equal the percentage of false rejects. This equal error rate provides an estimate of the statistical performance of the algorithm, i.e., it provides an estimate of its generalization error. We calculate the value of the equal error rate by intersecting the FAR and FRR curves that we computed, considering them to be piecewise linear.

One common problem of many on-line system for signature verification is the lack of examples needed to build a reliable model for a signature and to asses the performance of the algorithm. This problem is inherent to the application since it is not feasible to ask a subject for all the examples of his/her signature required to perform these two tasks reliably. Thus, we have to build a model of the signature that will perform well in practice and we have to infer the generalization error of the algorithm, all with very few examples. We could increase the number of examples in both the training and test set by using *Duplicate Examples* as described by Y. Abu-Mostafa [1] if we know that the model that we are building should be invariant with respect to some transformation of the examples. In our particular case, one possible example of this transformation is time origin translation since our system should be insensitive to the particular instant of time in which we started acquiring the signature. Another possible transformation is given by affine deformation of the signatures, provided that the acceptable range of the parameters of this affine deformation could be estimated

from the examples. We used both transformations in order to produce duplicate examples for this experiment.

Figure 12 shows the error trade-off curves for CDTW and DTW, calculated using our database of signatures. We only plot a section of the curve that is most informative. We observe that the performance of CDTW is about 0.1% better than the performance of DTW for the case of random forgeries and about 0.3% worse than the performance of DTW for the case of intentional forgeries. We see that the performance of both methods is very similar, with CDTW not showing a definite improvement over DTW. Nevertheless, this results is an initial ground for the development of better similarity measures in order to decrease the classification error.
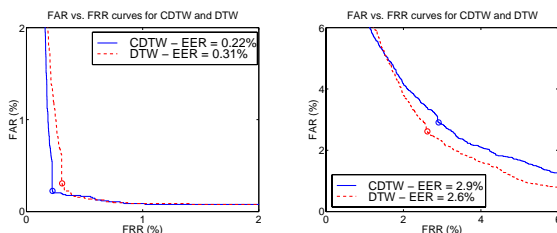

Figure 12. *Error trade-off curves for CDTW and DTW working on our database of signatures. The first plot was obtained using random forgeries and the second plot was obtained using intentional forgeries.*

4. Conclusions and Further Work

We presented a novel algorithm for establishing the correspondence and measuring the similarity of pairs of planar curves. The algorithm belongs to the general class of dynamic programming algorithms. We derived structural properties that allows to keep the spatial complexity of the algorithm bounded, as shown in the experimental results, without the need for heuristic approximations as the ones described in previously proposed algorithms for continuous matching.

We have compared the performance of CDTW with the performance of DTW on both synthetic and real data. DTW was used with and without oversampling of the curves in order to provide better matching spatial resolution. In terms of computational time required to perform the matching, CDTW is three orders of magnitude slower than plain DTW, being equivalent for DTW with an oversampling factor of 25 (approximately).

We have compared the performance of CDTW with the performance of DTW applied to signature verification, showing that both algorithms achieve similar results. We have shown that CDTW provides a less noiser way of computing the reference from the training set, so, in principle, the performance of CDTW could be improved by using a better set of parameters for classification, that take advantage of the matching at subsample resolution provided by the algorithm. One unsolved problem is how to make use of this alignment in order to extract a better prototype from the training set as well as a better characterization of the variability within the training set. Another problem for further work is the development of a better similarity measure between signatures given the correspondence obtained with CDTW.

References

[1] Y. S. Abu-Mostafa. Hints. *Neural Computation*, 7:639–671, 1995.

[2] R. Bellman. *Dynamic Programming*. Princeton Univ. Press, 1957.

[3] K. Huang and H. Yan. On-line signature verification based on dynamic segmentation and global and local matching. *Optical Engineering*, 34(12):3480–3487, 1995.

[4] R. Martens and L. Claesen. On-line signature verification by dynamic time-warping. In *Proc. 13th Int. Conf. Pattern Recognition*, pages 38–42, 1996.

[5] M. Munich and P. Perona. Camera-based id verification by signature tracking. In *Proc. 5th Europ. Conf. Comput. Vision, H. Burkhardt and B. Neumann (Ed.), LNCS-Series Vol. 1407-1408, Springer-Verlag*, pages 782–796, 1998.

[6] M. Munich and P. Perona. Visual-based id verification by signature tracking. In *Proc. 2nd Int. Conf. Audio- and Video-Based Person Authentication*, 1999.

[7] M. Munich and P. Perona. Visual signature verification using affine arc-length. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, 1999.

[8] V. S. Nalwa. Automatic on-line signature verification. *Proceedings of the IEEE*, 85(2):215–239, 1997.

[9] M. Parizeau and R. Plamondon. A comparative analysis of regional correlation, dynamical time warping and skeletal tree matching for signature verification. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 12(7):710–717, 1990.

[10] L. Rabiner and B. Juang. *Fundamentals of Speech Recognition*. Prentice Hall, Inc., 1993.

[11] Y. Sato and K. Kogure. On-line signature verification based on shape, motion and writng pressure. *Proc. 6th Int. Conf. on Patt. Recognition*, pages 823–826, 1982.

[12] B. Serra and M. Berthod. Subpixel contour matching using continuous dynamic programming. *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 202–207, 1994.

[13] B. Serra and M. Berthod. Optimal subpixel matching of contours chains and segments. *Proc. 5th Int. Conf. Computer Vision*, pages 402–407, 1995.

[14] C. Tappert, C. Suen, and T. Wakahara. The state of the art in on-line handwriting recognition. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 12:787–808, 1990.

[15] B. Wirtz. Stroke-based time warping for signature verification. In *Proc. Int. Conf. on Document Analysis and Recognition*, pages 179–182, 1995.