

Gesture Recognition Based on Elastic Deformation Energies

Radu-Daniel Vatavu^{1,2}, Laurent Grisoni¹, and Stefan-Gheorghe Pentiu²

¹ Laboratoire d'Informatique Fondamentale de Lille, Villeneuve d'Ascq 59650, France

² University Stefan cel Mare, Suceava 720229, Romania

vatavu@eed.usv.ro, laurent.grisoni@lifl.fr, pentiu@eed.usv.ro

Abstract. We present a method for recognizing gesture motions based on elastic deformable shapes and curvature templates. Gestures are modeled using a spline curve representation that is enhanced with elastic properties: the entire spline or any of its parts may stretch or bend. The energy required to transform a gesture into a given template gives an estimation of the similarity between the two. We demonstrate the results of our gesture classifier with a video-based acquisition approach.

Keywords: gesture recognition, elastic matching, deformation energies, video, gesture acquisition.

1 Introduction

Gestures have been given considerable attention lately as an effective mean for human computer interaction. The motivation lies in naturalness and efficiency: people use gestures in the real-world in order to interact with real objects and to convey information. A gesture-based interface would thus provide the net advantage of familiarity and intuitiveness with respect to other input devices. That is of course if one considers the ideal gesture framework that does not importunate, distract or add cognitive load [1,2].

A successful implementation of a gesture-based interface requires the selection of an appropriate technology for the acquisition process, a gesture representation technique that will suit the implementation of a robust classifier and finally, providing appropriate and efficient feedback to the user. Many of the above issues have been widely discussed in the literature. Good overviews on the state-of-the-art in gesture-based interaction including gesture taxonomies for HCI, existing technologies, recognition and interpretation techniques are given in [3,4]. An important problem that may be particularly identified relates to gesture motion trajectories. Trajectory recognition is difficult due to the variability that comes with gesture execution: different users will input different patterns for the same gesture type and even more, the same user will perform the same gesture differently at different moments in time by unwillingly including a certain degree of variability. Robust approaches are hence needed in order to support variations which translate into local deformations of the trajectory parts such as stretching or extra bending, articulations or any other small differences.

We propose in this paper a motion trajectory recognition method based on previous results on deformable shapes [5,6,7,8]. Robust energy measures for local deformations are discussed together with a procedure that allows automatic computation of templates from a given set of training samples. In order to demonstrate our gesture classifier we equally built a video-based acquisition system for detecting hands and retrieving their motion trajectories in a tabletop-like scenario. The paper is organized as follows: section 2 gives an overview of related work on gesture-based interfaces with a focus on video acquisition; section 3 provides details on the acquisition of hands and motion trajectories as well as on the gesture representation method using spline curves; the elasticity concepts of the deformation approach are presented in section 4 together with the classification method and recognition results; we believe that the elastic view on gestures may lead to interesting new future work as stated under the conclusions section.

2 Related Research

Gesture recognition research has been extensively conducted lately and the idea of interacting by gestures received support among public as it was induced by various media. A great variety of devices for acquiring gestures have been developed such as trackers, pointing and whole hand or body movements sensing and acquisition apparatus [9]. Among all, video-based acquisition presents the main advantage of not being intrusive and not requiring users to wear additional equipments or devices. The final feeling is thus of comfortability and naturalness. Video-based acquisition comes however with several drawbacks such as: high processing power required by the real-time interaction demand especially when more video cameras are involved; dependency on the working scenario and environment conditions such as lighting, user skin color, changing background; hands occlusion. Surveys on visual gesture recognition may be found in [9,10].

When it comes to detecting and recognizing gestures performed by hand, common approaches are to follow colored gloves [11], detect skin color [12,13] and track local features [14,15] and active shape models [16]. KLT features, named after Kanade, Lucas, Tomasi are *good features* to track [14]; sets or *flocks* of KLT features have been used by Kolsch and Turk [15] for achieving robust hand tracking and their technique is available in the HandVu system. Recognition was performed using Markov models [17], finite state machines [18], temporal motion templates [19], geometric features [20], probability signatures [21] and various shape similarity measures [22].

In what concerns the recognition of motion trajectories, many researchers have considered shape analysis approaches that make use of local parameters such as the curvature: CSS (Curvature Scale Space) representations [22], detection of high curvature points [23], similarity measures based on curvature differences [24] or various curvature-based representations [25]. Trajectory recognition remains a difficult problem as already mentioned in the introduction due to the variability that comes with each gesture execution.

3 Gesture Acquisition and Representation

We consider a gesture as a point moving in time $gesture(t) : [t_0, t_1] \rightarrow \mathbb{R}^2$ in the continuous domain as well as a series of points sampled at a given resolution r in the discrete case: $gesture = \{p_i/p_i \in \mathbb{R}^2, i = \overline{1, r}\}$. We are only interested in the motion trajectory that is associated with gestures performed by hand and discard all information related to postures.

3.1 Video Acquisition of Hand Gestures

Hand gestures are acquired using a top-view mounted video camera that monitors the surface of a table as illustrated in Figure 1. Hands segmentation is achieved using a simple low-cost skin filtering in the HSV color space on the Hue and Saturation components:

$$pixel\ p\ is\ skin \Leftrightarrow Hue(p) \in [H_{low}, H_{high}] \wedge Saturation(p) \in [S_{low}, S_{high}] \quad (1)$$

where p is the current pixel submitted to classification and $H_{low}, H_{high}, S_{low}, S_{high}$ are the low and high thresholds for the Hue and Saturation components. The homogeneous blue colored background of the table was chosen as it provides contrast with the users' skin color which allows for a robust hand and forefinger detection in real-time at 25 fps. The values for the Hue and Saturation thresholds were chosen experimentally as $H_{low} = 180, H_{high} = 240, S_{low} = 20, S_{high} = 150$ where Hue varies from 0 to 359 and Saturation from 0 to 255. We preferred to use this simple low-cost solution for detecting hands in a controlled environment for real-time processing purposes although there are other skin modeling approaches that deal very well with various scenario conditions [12,13] but which go beyond the scope of this paper. We are only interested at this stage in retrieving an accurate motion trajectory of gestures performed by hand.

The gesture trajectory corresponds to the forefinger of the user's hand while it is pointed. Our one-video camera acquisition approach will generate 2D curves yet all the following discussion is also relevant and may be equally extended to 3D curves as stated under the conclusions section.

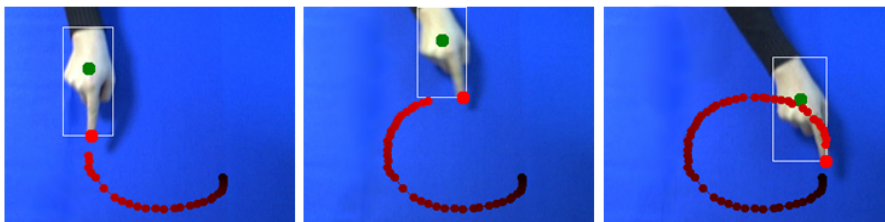


Fig. 1. Gesture acquisition above the table surface using a top-view video camera

3.2 Spline-Based Motion Representation

The acquired raw trajectory of the hand motion is simplified with a fast version of the Douglas-Peucker polyline reduction algorithm [26] into a series of 2D points $\{p_i, i = \overline{1, n}\}$ and modeled using a Catmull-Rom spline representation [27]. We perform data simplification followed by spline modeling in order to reduce and attenuate small variations that may occur in gesture execution and that may affect the performance of classifiers at later stages.

The Catmull-Rom splines are a family of cubic interpolating splines defined such that the tangent at each control point p_i is calculated using previous and next neighboring points:

$$t_i = r_i \cdot [p_{i-1}p_i] + (1 - r_i) \cdot [p_i p_{i+1}] \quad (2)$$

where $[p_{i-1}p_i]$ denotes the vector from point p_{i-1} to p_i . The spline is completely defined by the control points p_i and their associated tangents t_i , $i = \overline{1, n}$. The i th segment of the spline is defined between the control points p_i and p_{i+1} as $p(u) = \sum_{k=0}^3 c_k \cdot u^k$ where u is a local parameter that varies in the $[0, 1]$ interval. The coefficients c_k , $k = \overline{0, 3}$ are computed for each segment using end-continuity conditions:

$$p(0) = c_0, p(1) = c_0 + c_1 + c_2 + c_3, p'(0) = c_1, p'(1) = c_1 + 2c_2 + 3c_3 \quad (3)$$

Catmull-Rom splines have C^1 continuity and local control [27]. The parameters r_i are known as tensions and affect how sharply the curve bends at the interpolated control point p_i (common value is 0.5). We have chosen r_i to be $r_i = |p_i p_{i+1}| / (|p_{i-1} p_i| + |p_i p_{i+1}|)$ where $|\cdot|$ denotes the Euclidean distance, which allows for each segment $[p_i p_{i+1}]$ to be weighted inversely proportional to its length and thus giving much smoother curve shapes. Examples of a few gesture trajectories and their corresponding spline representations are given in Figure 2.

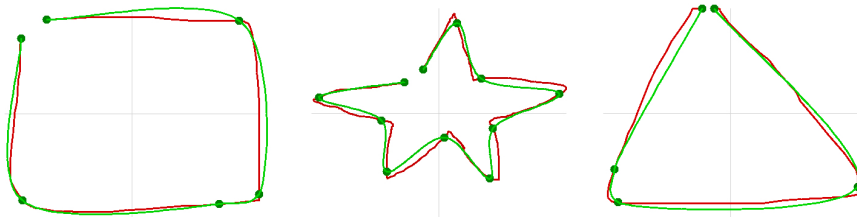


Fig. 2. Raw gesture trajectory (in red) and spline representation (in green, control points displayed) for three gesture types: rectangle, star and triangle

3.3 Curvature Functions

The fundamental theorem of differential geometry of curves [28] states that the curvature signature function $\kappa(s)$ of a planar curve $C(s)$ parameterized by arc-length s fully prescribes it up to a rigid motion transformation. The curvature

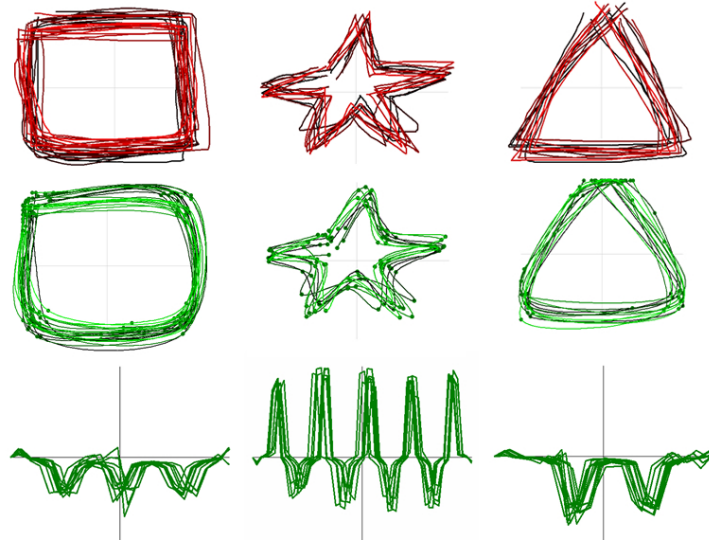


Fig. 3. Different gesture representations: raw acquired trajectories (top), spline representations (middle) and curvature functions from spline representations (bottom). Multiple executions from the same user are displayed superimposed.

functions describe the shape of gestures completely and may be used for shape matching. Moreover, plotting the curvature functions for multiple instances of the same gesture visually confirms high inter-class and small intra-class variances. Figure 3 illustrates plotting the curvature functions of several gesture trajectories. The figure also gives an idea of the variability that exists within gestures while they are executed: all gestures were performed by the same user at different moments in time. The intra-class variance is given by variations in length and bending while executing gestures.

4 An Elastic Deformation Approach for Gesture Recognition

4.1 Deformation Energies

Deformation-based approaches for the purpose of curve matching consider the transformation of one curve into another by minimizing a performance functional of energies in accordance with elastic theory [5,6,7,8]. Ideal alignments between curves will allow for similar parts to be compared together, leaving out errors that may be caused by articulations, deformation of parts or other variations in the curves' shapes.

A gesture curve is being viewed as a chain of connected elastic springs with infinitesimal lengths. Each spring may be subjected to stretching and bending. The amount of stretching is measured by the difference in length while bending

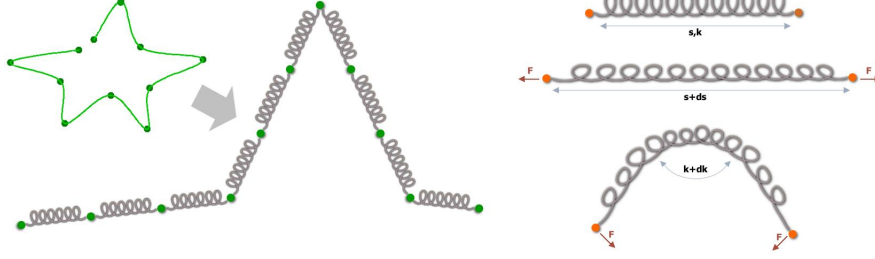


Fig. 4. Elastic gesture representation: the gesture trajectory is a chain of connected elastic springs (left); springs may be subjected to deformations such as stretching and bending (right)

accounts for the difference in curvature. The larger these differences are the larger the associated energy measure is and consequently, the cost of deforming one spring into another becomes bigger. Figure 4 illustrates this idea for a star shaped-like gesture. The principle of the approach is that similar curves would need small energies for transforming one into other while different curves would require bigger deformation costs.

The stretching energy required to deform one spring of length ds into one of length \overline{ds} may be given in analogy with Hookes' law from the elastic theory [29]:

$$E_{stretching} = \frac{1}{2}\alpha (ds - \overline{ds})^2 \quad (4)$$

where α is the stiffness coefficient of the springs. Similarly, instead of considering springs of length we consider springs of angle between tangents at the spring extremities. The energy needed to bend a spring of angle $d\phi$ into another of angle $\overline{d\phi}$ may be expressed in terms of curvatures as:

$$E_{bending} = \frac{1}{2}\alpha (d\phi - \overline{d\phi})^2 = \frac{1}{2}\alpha (dk \cdot ds - \overline{dk} \cdot \overline{ds})^2 \quad (5)$$

where curvature is defined as $dk = \frac{d\phi}{ds}$. We may further express the total energy cost needed to transform one spring into another as:

$$E_{deformation} = E_{stretching} + R \cdot E_{bending} \quad (6)$$

where R is a coefficient that controls the distribution of the two energy terms.

4.2 Aligning Curves

Let $C(s)$ and $\overline{C}(\overline{s})$ be two curves indexed by arc-lengths s and \overline{s} where $s \in [0, L]$, $\overline{s} \in [0, \overline{L}]$ and L, \overline{L} are the lengths of the curves. Let a mapping $g : [0, L] \rightarrow [0, \overline{L}]$, $g(s) = \overline{s}$ represent an alignment of the two curves. The principle of elastic

alignment may be described by finding the best alignment g that minimizes the total stretching and bending energy [6]:

$$\mu[g] = \int_C (C(s) - \overline{C}(\overline{s}))^2 + R \cdot (k(s) - \overline{k}(\overline{s}))^2 ds \quad (7)$$

where $k(s), \overline{k}(\overline{s})$ are the curvatures along the C, \overline{C} curves and R is the parameter that controls the contribution of each energy term. The optimal match is given as $\mu[C, \overline{C}] = \mu^* = \min_g \{\mu[g]\}$. A premise of the approach is that the goodness of the optimal match is the sum of the goodness of infinitesimal matches. The above functional is asymmetrical but it may be transformed into a symmetrical one by expressing both s and \overline{s} as functions of a new parameter as in [5]. Furthermore, [5] shows that the optimal alignment of the two curves respects the properties of a metric function.

In the discrete case, the two curves C, \overline{C} are sampled into n and m points respectively, $C = \{s_i/i = \overline{1}, n\}$, $\overline{C} = \{\overline{s}_j/j = \overline{1}, m\}$ and the alignment becomes a sequence of ordered pairs $\alpha_k = (s_{i_k}, \overline{s}_{j_k})$ with (s_1, \overline{s}_1) and (s_n, \overline{s}_m) being the first and last pairs of the sequence, $i_k \in \{1, n\}$ and $j_k \in \{1, m\}$. The optimal alignment is found via dynamic programming by considering an energy cost propagation scheme. Let $cost_{i,j}$ be the total cost of transforming the first i springs of curve C into the first j springs of curve \overline{C} . Also let $e_{i \rightarrow j}$ be the energy term required to transform the i th spring of curve C into the j th spring of curve \overline{C} :

$$e_{i \rightarrow j} = e_{stretching, i \rightarrow j} + R \cdot e_{bending, i \rightarrow j} = (ds_i - \overline{ds}_j)^2 + R \cdot (dk_i - \overline{dk}_j)^2 \quad (8)$$

We also define the energy of removing the i th spring by transforming it into a void/nil spring of 0 length and 0 curvature: $e_{i \rightarrow nil} = ds_i^2 + R \cdot dk_i^2$. The final energy cost propagation scheme is given by:

$$\begin{cases} cost_{1,1} = e_{1 \rightarrow 1} \\ cost_{1,j} = cost_{1,j-1} + e_{j \rightarrow nil} \\ cost_{i,1} = cost_{i-1,1} + e_{i \rightarrow nil} \\ cost_{i,j} = \min \begin{cases} cost_{i-1,j-1} + e_{i \rightarrow j} \\ cost_{i-1,j} + e_{i \rightarrow nil} \\ cost_{i,j-1} + e_{j \rightarrow nil} \end{cases} \end{cases} \quad (9)$$

We use the elastic deformation approach based on the notion of the alignment curve in the discrete case and apply it directly to the curvature functions. An illustration of the alignment result between the curvature functions of two star gestures is given in Figure 5.

4.3 Gesture Matching

Gesture recognition is performed by matching new gestures against pre-defined templates in the form of average curvature functions. Let $C_i(s_i), i = \overline{1, N}$ be N curves representing multiple executions for the same gesture type. The curves

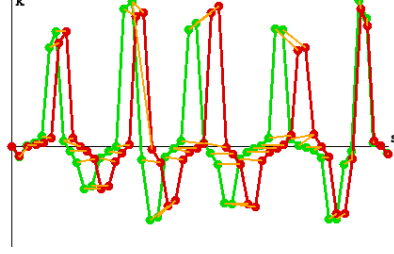


Fig. 5. Elastic alignment of the curvature functions for two star gestures

are parameterized by arc-lengths $s_i \in [0, L_i]$ with L_i being the length of the i th curve. The average curve \tilde{C} is computed by averaging the lengths and curvatures of the corresponding subsegments (springs) as they are paired during the alignment procedure of each curve to a reference. We are selecting as reference the curve C_{i^*} that produces the minimum total alignment cost:

$$\min_{i^*=1, N} \left\{ \sum_{i=1, i \neq i^*}^N \mu[C_i, C_{i^*}] \right\} \quad (10)$$

The average curve \tilde{C} will have the same sampling resolution as the reference curve $\tilde{n} = n_{i^*}$ while the arc-length and curvature values are averaged during the alignment process:

$$\tilde{s}_j = \frac{\sum_{t=1, t \neq i^*}^N \sum_{\alpha_r^{i^*} = (s_j^{i^*}, s_{j_r}^t)} s_{j_r}^t}{\sum_{t=1, t \neq i^*}^N 1}, \quad \tilde{k}_j = \frac{\sum_{t=1, t \neq i^*}^N \sum_{\alpha_r^{i^*} = (s_j^{i^*}, s_{j_r}^t)} k_{j_r}^t}{\sum_{t=1, t \neq i^*}^N 1} \quad (11)$$

where $j = 1, \tilde{n}$ and $\alpha_r^{i^*} = (s_j^{i^*}, s_{j_r}^t)$ denotes the part of the alignment between C_{i^*} as reference curve and C_t that considers only those points $s_{j_r}^t$ from C_t that are aligned to the point $s_j^{i^*}$ from C_{i^*} . Figure 6 shows the average curvature function as computed from multiple examples of the rectangle gesture.

The averaging process outputs the curve \tilde{C} sampled in $\{\tilde{s}_j, j = \overline{1, \tilde{n}}\}$ with curvatures $\{\tilde{k}_j, j = \overline{1, \tilde{n}}\}$, $\tilde{s}_1 = 0$ and $\tilde{s}_{\tilde{n}} = 1$. Classification of a new gesture C



Fig. 6. Average curvature function computed from multiple rectangle gestures

sampled in $\{s_i, i = \overline{1, n}\}$ with curvature values at sampled points $\{k_i, i = \overline{1, n}\}$ into a particular class of already pre-defined types is done by computing an alignment and a cost of match with respect to the average curvature function of each class in a nearest-neighbor approach. Let $\tilde{C}_1, \tilde{C}_2, \dots, \tilde{C}_k$ be the set of gesture templates (or curvature functions averages) representing k gesture types. The new gesture C is classified as being of type $j \in \{1, k\}$ where j is the index of the class that minimizes the alignment cost:

$$j = \min_{i=1, k} \left\{ \mu \left[C, \tilde{C}_i \right] \right\} \tag{12}$$

4.4 Performance Results

Classifier performance. We acquired a data set of 2,000 gesture samples from 10 subjects. Each subject performed each of the 10 gesture types from Figure 7 several times. Figure 8 lists several matching results given twice: as numerical values and using visual cues. The gray-levels range from white to black where black stands for maximum difference while white shows a perfect match.

We had a total of 200 samples for each gesture type. Out of these samples, we chose T that made up the training set. We performed the training stage 100 times by randomly choosing the T samples. The rest of $200 - T$ samples were added to the testing set. From each testing set and for each of the 10 gesture types, one sample was randomly selected and classified against the trained classifiers. We performed these steps 100 times and updated each time the classification error rate ($10 \times 100 = 1,000$ tests were computed for a specified T value). We varied T from 2 to 10 samples which led to an error rate of 6% (or 94% accuracy) when using only 2 training samples and 2.5% (97.5% accuracy) for 10 training samples (or 1 sample from each user).

Discussion. The main cause for classification errors was related to wrongly classifying down-left gestures as right-arrows, down-right gestures as left-arrows and vice versa. This was due to the fact that our classifiers are rotation invariant hence the only difference between these gesture types stands in their turn angle (which is 90 degrees for down-left and down-right gestures and smaller for right- and left- arrows). By selecting only 2 stroke samples from the entire dataset acquired from 10 users (or 0.2 samples per user), the achieved accuracy performance was of 94%. The accuracy percent went up to 97.5% with 10

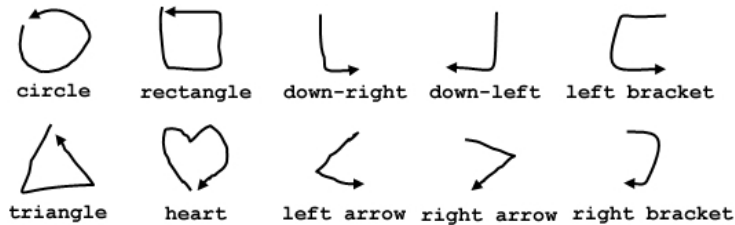


Fig. 7. Set of 10 gesture types

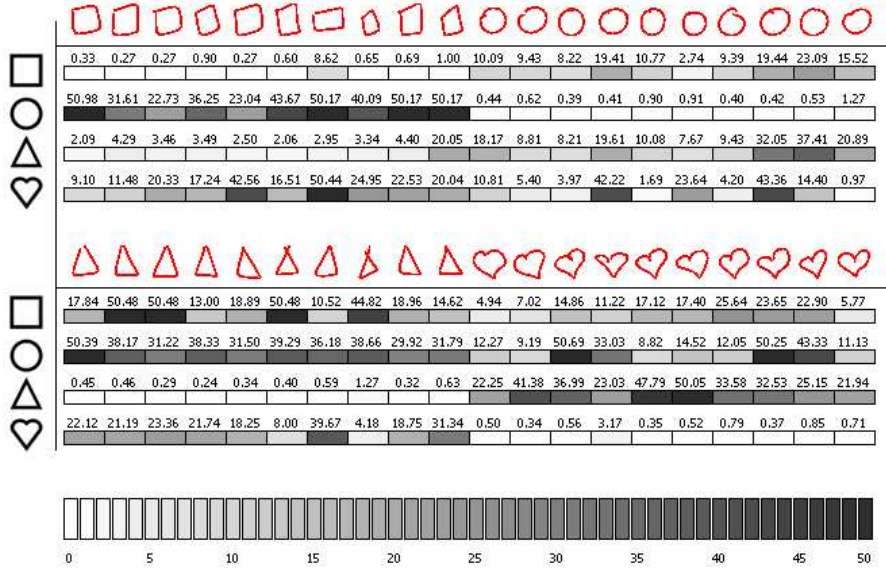


Fig. 8. Matching results for several gesture types

stroke samples (or 1 sample from each user) which makes our method suitable for multi-user gesture recognition.

System performance. Video-based acquisition was performed on a dual core P4 2.66GHz machine at a video rate of 25fps with a 320x240 image resolution. The CPU load varied around 25-30% mainly due to video processing (15-20%). The maximum time interval allowed for executing a gesture was of 5 seconds which limited the spline representation to maximum 125 control points at 25fps. The resolution step chosen for the spline segments (i.e. the number of elastic springs per segment) was 5 giving in the end a maximum dimensionality for the algorithm data of order $n \approx 600$ with an average of $n \approx 250$. The complexity of our classification algorithm for a gesture set composed of $|G|$ templates is $O(|G| \times n \times m)$ where n and m represent the resolutions of the two sampled gestures. Although the complexity is quadratic, the domain range is limited, n and m being of order 250 in the average case.

Video. A demonstrative video of the system running including acquisition and classification for the purpose of creating virtual objects may be downloaded from <http://www.eed.usv.ro/~vatavu>.

5 Conclusions

We presented in this paper a method for recognizing gestures based on a spline representation for motion trajectories. The classification algorithm is invariant

to translation, scale and rotation due to the use of curvature functional representation however it depends on the starting point and direction of execution (due to the fact that the curvature function changes sign when direction changes). Also, we were interested in our approach solely on the recognition of gesture shapes and discard other execution parameters such as speed. However, it may prove interesting to investigate how execution speed may help classification.

Although we only demonstrate 2D gesture recognition, we believe that the method can be extended to the 3D case by taking into account another parameter, i.e. the torsion. Similar to the fact that a planar curve is completely described (up to translation) by its curvature function, a space curve is equally defined by the pair curvature and torsion. We believe that extending our method to the 3D case is interesting as future work while the analysis of the relationship between the two components curvature and torsion may reveal useful facts with regards to gesture execution.

Acknowledgments. The work presented in this paper was partially supported by IRCICA funding, the AUF Bourse de Formation a la Recherche Ref. No. 1021FR58ML/2006 and the Research of Excellence funding grant Ref. No. CEEEX 131/2006.

References

1. Cerney, M.M., Vance, J.M.: Gesture recognition in virtual environments: A review and framework for future development. Iowa State, TR ISUHCI-2005-01 (2005)
2. Nielsen, M., Moeslund, T., Storrang, M., Granum, E.: A procedure for developing intuitive and ergonomic gesture interfaces for HCI. In: Proc. 5th Int. Workshop on Gesture and Sign Language based HCI, Genova, Italy (2003)
3. Jaimes, A., Sebe, N.: Multimodal human computer interaction: a survey. In: Proc. of IEEE Int. Workshop on HCI, Beijing, China, pp. 1–15 (2005)
4. Watson, R.: A survey of gesture recognition techniques. TR TCD-CS-93-11, Trinity College Dublin (1993)
5. Sebastian, T.B., Klein, P.N., Kimia, B.B.: On Aligning Curves. IEEE Trans. Pattern Anal. Mach. Intell. 25(1), 116–125 (2003)
6. Cohen, I., Ayache, N., Sulger, P.: Tracking Points on Deformable Objects Using Curvature Information. In: Sandini, G. (ed.) ECCV 1992. LNCS, vol. 588, pp. 458–466. Springer, Heidelberg (1992)
7. Basri, R., Costa, L., Geiger, D., Jacobs, D.: Determining the similarity of deformable shapes. Vision Research 38, 2365–2385 (1998)
8. Azencott, R., Coldefy, F., Younes, L.: A distance for elastic matching in object recognition. In: Proc. 13th Int. Conf. on Pattern Recognition, pp. 687–691 (1996)
9. LaViola, J.: A survey of hand posture and gesture recognition techniques and technology. TR CS-99-11, Brown University (1999)
10. Pavlovic, V., Sharma, R., Huang, T.: Visual interpretation of hand gestures for human-computer interaction A review. IEEE Trans. on PAMI 19(7) (1997)
11. Dorner, B: Chasing the colour glove: Visual hand tracking. Master's thesis, Simon Fraser University (1994)
12. Jones, M.J., Rehg, J.M.: Statistical color models with application to skin detection. Cambridge Research Laboratory, TR 98/11 (1998)

13. Lee, J.Y., Yoo, S.I.: An elliptical boundary model for skin color detection. In: Proc. Int. Conf. on Imaging Science, Systems and Technology, Las Vegas, USA (2002)
14. Shi, J., Tomasi, C.: Good features to track. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition, Seattle (1994)
15. Kolsch, M., Turk, M.: Fast 2d hand tracking with flocks of features and multi-cue integration. In: Proc. IEEE Workshop on Real-Time Vision for HCI (2004)
16. Chang, J.S., Kim, E.Y., Jung, K., Kim, H.J.: Real time hand tracking based on active contour model. In: Gervasi, O., Gavrilova, M.L., Kumar, V., Laganá, A., Lee, H.P., Mun, Y., Taniar, D., Tan, C.J.K. (eds.) ICCSA 2005. LNCS, vol. 3483, pp. 999–1006. Springer, Heidelberg (2005)
17. Wilson, A., Bobick, A.: Realtime online adaptive gesture recognition. In: Proc. ICPR 2000 (2000)
18. Hong, P., Turk, M., Huang, T.S.: Constructing finite state machines for fast gesture recognition. In: Proc. 15th ICPR Barcelona, Spain, vol. 3, pp. 691–694 (2000)
19. Bobick, A.F., Davis, J.W.: The recognition of human movement using temporal templates. *IEEE Trans. Pattern Anal. Mach. Intell.* 23(3), 257–267 (2001)
20. Fonseca, M.J., Pimentel, C., Jorge, J.A.: CALI: An Online Scribble Recognizer for Calligraphic Interfaces. *AAAI Sketch Understanding*, 51–58 (2002)
21. Martin, J., Hall, D., Crowley, J.L.: Statistical gesture recognition through modelling of parameter trajectories. In: 3rd Gesture Workshop, France (1999)
22. Mokhtarian, F., Abbasi, S.: Shape Similarity Retrieval under Affine Transforms. *Pattern Recognition* 35(1), 31–41 (2002)
23. Deriche, R., Faugeras, O.: 2-D curve matching using high curvature points: application to stereo vision. In: Proc. 10th Int. Conf. on Pattern Recognition (1990)
24. Femiani, J.C., Razdan, A., Farin, G.: Curve Shapes: Comparison and Alignment. *TPAMI* (November 2004)
25. Gatzke, T., Grimm, C., Garland, M., Zelinka, S.: Curvature Maps For Local Shape Comparison. In: Proc. Int. Conf. on Shape Modeling and Applications (SMI) (2005)
26. Hershberger, J., Snoeyink, J.: Speeding Up the Douglas-Peucker Line-Simplification Algorithm. In: Proc. of 5th Symposium on Data Handling, pp. 134–143 (1992)
27. Catmull, E., Rom, R.: A class of local interpolating splines. In: Barnhill, R.E., Reisenfeld, R.F. (eds.) *Computer Aided Geometric Design*. Academic Press, London (1974)
28. Do Carmo, M.: *Differential Geometry of Curves and Surfaces*. Prentice-Hall, Englewood Cliffs (1976)
29. Kosevich, A.M., Lifshitz, E.M., Landau, L.D., Pitaevskii, L.P.: *Theory of Elasticity*, 3rd edn., Butterworth-Heinemann (1986)