A Curve Segmentation Algorithm That Automates Deformable-Model-Based Target Tracking

Ioannis Pavlidis

Honeywell Technology Center
Honeywell Inc.
Minneapolis, MN 55418
pavlidis@htc.honeywell.com

Nikolaos P. Papanikolopoulos

Dept. of Computer Science
University of Minnesota
Minneapolis, MN 55455
npapas@cs.umn.edu

ABSTRACT

A novel curve segmentation algorithm for determining control points for deformable-model-based target tracking is proposed. The algorithm is parameterless enabling a full-fledged automated tracking regardless of the shape of the object being tracked. Compared with other curve segmentation algorithms, it selects a minimal number of control points that yet deliver a superior shape description. The algorithm is comparatively tested with other curve segmentation algorithms in a variety of characteristic target outlines.

1 INTRODUCTION

The need for target tracking arises in a number of different applications in robotics research. Characteristic examples include vision-based control of grasping and manipulation tasks [7] and visual tracking of moving objects [2]. Target tracking is also important in a number of other applications, like the case of pedestrian tracking in an Intelligent Transportation System (ITS) [6]. A well-established and popular technique for target tracking involves the use of deformable models [2, 6, 7]. Deformable models originate from the "snake," a model for representing image contours first developed by Kass *et al.* [3].

A necessary first step in the computation of a deformable model is to determine a set of control points to approximate the tracked object's contour. To date, this is done by hand through a user-interface. However, the possibility of using a curve segmentation algorithm is often indicated. Picking control points manually, renders difficult the automation of the entire tracking task. In addition, since the user is picking the points randomly or at best using some heuristic developed through his/her own experience, he/she tends to pick either too many or too few control points. On the other hand, using some classical curve segmentation algorithm [1, 4, 5] only half-automates the task since the performance of these algorithms depends upon the fine tuning of a number of parameters. Different object shapes may require different parameter settings or otherwise the segmentation algorithm will perform at times either excessive segmentation or sparse segmentation.

In this paper we propose a segmentation algorithm (from now on, we will denote it as P & P) that fills out the existing gap in all the respects. It fully automates the selection of the control points since it does not depend on any parameters and works equally well for most kinds of shapes. Comparatively to other curve segmentation algorithms, it manages to select the minimal number of points that yet deliver a superior description of the original shape.

The organization of the paper is as follows: Section presents some previous curve segmentation algorithms and discusses their shortcomings for the task at hand. Section describes the algorithm we propose. In Section 3.2, the results from experimental tests are presented. Finally, in Section 3.2, the paper is summarized and conclusions are drawn.

2 PREVIOUS WORK

Several interesting techniques that segment continuous lines in various ways have already been proposed in the literature in fields other than target tracking. The criteria against which the various curve segmentation algorithms should be judged for deformable-model-based tracking purposes are the following:

- First, the number of segmentation (control) points should be kept to the minimum, since the speed of deformable-model-based tracking is usually linear in the number of control points.

- Second, the control points chosen should deliver an accurate description of the tracked object's contour. This is important because it allows the

deformable model to follow small deformations.

The above two criteria appear at first contradictory and researchers resorted to either choosing too many control points, thus compromising the tracking speed, or too few control points, thus compromising the quality of tracking. Interestingly, these two criteria are not quite met without careful or ad hoc parameter tuning by many prominent curve segmentation algorithms. Two typical curve segmentation approaches are briefly presented in the following paragraphs for illustrative and comparative purposes.

A series of curve segmentation algorithms of the "split and merge" kind have been proposed by T. Pavlidis *et al.* [4, 5]. These algorithms iteratively construct a polygonal approximation of the curve. The apexes of the polygonal approximation are the resultant segmentation points. A "split and merge" algorithm (from now on we may denote it as P algorithm) uses as a parameter an error factor. The number of iterations and the approximating power of a P algorithm depend upon the fine tuning of its parameter. These algorithms perform well when they are to reconstruct polygonally simple shapes. When, however, they are to reconstruct shapes by spline interpolation with any reasonable accuracy, they result into too many segmentation points.

Brault and Plamondon [1] presented another segmentation algorithm (from now on, we will denote it as B & P) that was meant to segment complex signature curves. The main idea of their approach is that each point i of the curve is considered as a potential segmentation point (vertex). The neighboring points of point i contribute to its vertex candidacy provided they meet certain geometric conditions. The strongest candidates become the resultant segmentation points. Comparatively to a "split and merge" algorithm, the B & P algorithm performs substantially better in reconstructing shapes - particularly complex shapes - by spline interpolation. This algorithm, however, depends on two parameters that need to be fine tuned. In addition, while the algorithm employs a very powerful technique for detecting corners, it does not have an equally powerful way of detecting key points in round or flat curves.

3 THE ALGORITHM

The algorithm we propose (P & P) takes a step further than the B & P algorithm. It does not depend on any parameters and thus offers a potential for true automation. It features a coherent mechanism for detecting not only corners but also some key points between corners. In this way, it improves its spline approximating power at the minimum cost. And because of its ability to locate some key points with relatively flat or round surroundings, it also performs satisfactorily in the case of rounded and flat objects.

3.1 *Corner Determination*

The determination of corners is done in a way very similar to the method followed in the B & P algorithm [1]. The notable difference is that there is no need for parameter tuning. The basic mechanism is the same with that of the B & P algorithm. Each point c of the curve is seen as a potential corner. The neighboring points from either side of point c contribute to the cornerness of c in a degree determined by certain conditions.

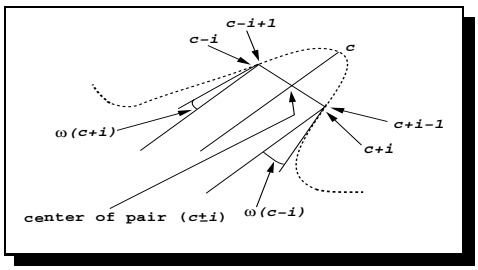

Figure 1: Geometric model for corner determination.

In more detail, the angles $\omega(c+i)$ and $\omega(c-i)$ (see Fig. 1) are computed for each pair of neighbors $c \pm i$ $(i = 1, 2...)$. In order for a pair $c \pm i$ to belong to the corner domain of point c the following inequalities must be satisfied:

$$\omega(c+i) < \frac{\pi}{2} \quad and \quad \omega(c-i) < \frac{\pi}{2}. \qquad (1)$$

The contribution CF (Cornerness Factor) of each pair $c \pm i$ to the making of the candidate corner c is computed by the formula

$$CF(c,i) = \cos(\omega(c+i)) * \cos(\omega(c-i)). \qquad (2)$$

Using $\frac{\pi}{2}$ as a fixed upper limit in the inequalities (1) is a major departure from the method followed in [1] and is what renders the corner determination parameterless. The above choice as well as formula (2) can be easily explained if we consider the geometrical model of a corner (see Fig. 2). In the domain of an ideal corner (see Fig. 2(a)) the formula (2) operates at the left extremum of the range $[0, \frac{\pi}{2})$, and henceforth gives its maximum response. The formula shuts off the domain of the ideal corner at its base where the angles $\omega(c+i)$ and $\omega(c-i)$ become $\frac{\pi}{2}$. In the neighborhood of less acute corners the formula operates at intermediate values of the range $[0, \frac{\pi}{2})$. In

these cases, the shut off of the corner's domain takes place once $\omega(c+i)$ or $\omega(c-i)$ become greater or equal than $\frac{\pi}{2}$ (see Fig. 2(b)). The formula gives its weakest response in the case of an almost straight line where it leaves the candidate corner (really, a non-corner) without domain at all (see Fig. 2(c)).

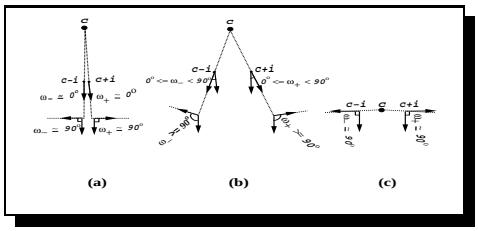

Figure 2: Range of geometric models for corners handled effectively by Formula (2). (a) Ideal corner (almost a spike). (b) Typical corner. (c) Non-corner (almost a straight line).

The first $M(c)$ points that satisfy the inequalities (1) constitute the corner domain of point c and their total contribution to the cornerness of point c is computed by

$$TCF(c) = \sum_{i=1}^{M(c)} CF(c,i). \qquad (3)$$

The corner segmentation points are identified by searching the values of the function $TCF(c)$. The TCF values of the curve points present a very consistent pattern: strings of nonzero values spaced by strings of zero values. Each of the nonzero strings corresponds to a high curvature segment, and the maximum value contained in each such string corresponds to a corner segmentation point.

3.2 *Key Flat Point Determination*

While corners are the perceptually most important parts in a curve, corners alone provide insufficient data for an accurate reconstruction of the curve by spline interpolation. If we are to fully automate the tracking task, we should keep the tension parameter of the spline fixed at a moderate value. This means, that spline-interpolating between corner points only, may yield substantial approximation errors to the original curve. The magnitude of the errors depends on the distance between the corners and the curvature of the original between-the-corners curve segments. The situation improves dramatically if we provide some key points with rather flat surroundings, that lie between corners, as extra segmentation points. The way we find these flat points is conjugate to the way we find the corner points.

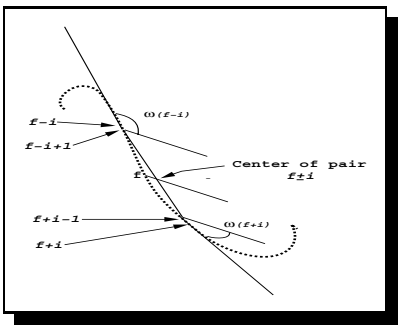

Figure 3: Geometric model for key flat point determination.

More precisely, a separate processing step is taking place for the location of the *key flat points*. The geometric parameters shown in Fig. 3 are the same with these in Fig. 1 and are computed for each pair of neighbors $f \pm i$ $(i = 1, 2...)$ of every point f of the curve. This time, however, the larger the angles $\omega(f+i)$, $\omega(f-i)$ are than $\frac{\pi}{2}$, the more the corresponding pair of neighboring points contributes to the *flatness* of point f. As a result, by a suitable analysis of the angles $\omega(f+i)$ and $\omega(f-i)$, one can determine whether or not the pair of points $f \pm i$ is part of the flat domain of f and, in addition, can estimate the importance of the contribution of these points to the flatness of point f.

The angles $\omega(f+i)$ and $\omega(f-i)$ must satisfy the following inequalities:

$$\omega(f+i) > \frac{\pi}{2} \quad or \quad \omega(f-i) > \frac{\pi}{2}. \qquad (4)$$

The contribution FF (Flatness Factor) of each pair $f \pm i$ to the making of the candidate key flat point i is computed by the formula

$$FF(f,i) = \mid \cos(\omega(f,i)) \mid * \mid \cos(\omega(f,i)) \mid . \qquad (5)$$

In contrast to equation (2), equation (5) uses the absolute value of the trigonometric function *cos* since the range of the angles $\omega(f+i)$ and/or $\omega(f-i)$ features now $\frac{\pi}{2}$ as a lower and not as an upper limit. The total contribution of the first $M(f)$ points belonging to the flat domain of f (the ones that satisfy the inequalities (4)) is computed by

$$TFF(f) = \sum_{i=1}^{M(f)} FF(f,i). \qquad (6)$$

The identification of the key flat segmentation points from the function $TFF(f)$ is done in a way analogous to the determination of the corner points from the function $TCF(c)$.

4 EXPERIMENTAL RESULTS

The algorithm was tested with the outline curve of a characteristic object (see Figs. 4) we use for target tracking in our robotic experimental setup. It was also tested with the outline figure of a pedestrian image (see Fig. 8) taken from our Intelligent Transportation System experimental setup. The outline curve of the object is produced by applying an edge following algorithm on the difference image produced from the original dynamic scene. The same curves were also subjected to segmentation by the B & P and P algorithms for comparison purposes.

The comparative experiments between the three segmentation algorithms were designed in the following way. As it was explained in the previous section, no parameters needed to be fixed in our P & P algorithm. The algorithm produced a set of control points for each curve that accurately described the original figure. The accuracy of description has been tested by using the control points as interpolating points for cardinal splines. The resulting spline curves almost completely coincided with the corresponding original curves. A careful parameter tuning for the other two segmentation algorithms was performed for each curve. This happened in order for the algorithms to produce sets of control points that deliver similarly accurate descriptions of the original figures. Apart from the fact that parameter tuning was needed for the B & P and P algorithms, our algorithm produced substantially fewer control points for the same fine descriptive results.

In the case of the rectangular target (Figs. 4 through 7), it is apparent the way our P & P algorithm works. The corners have been detected by the corner detection part of the algorithm, while the key flat points fall somewhere between the corners. The working logic is more obscure in the case of the B & P algorithm and even more so in the case of P algorithm, that seems to work simply by brute force.

In the case of pedestrian tracking (Fig. 8), the outline figure is a rather complex true real world figure. Our algorithm still outperforms the other two. It achieves this, by distributing control points more densely in parts of rapid curvature change and more sparsely in parts of slow curvature change. By following such a consistent strategy, the algorithm while economizes in control points it still delivers superior shape description.

In Fig. 12, bar charts that show the performance of the three segmentation algorithms for the various target types are presented. The results correspond to control point sets with highly accurate descriptive power for all three algorithms. It is apparent, that our

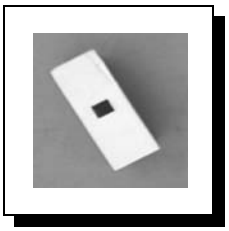

Figure 4: A rectangular target (black blob).

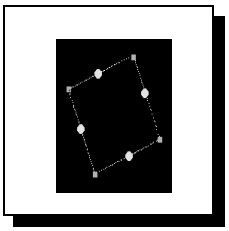

Figure 5: Rectangular outline segmented by P & P.

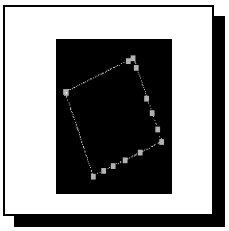

Figure 6: Rectangular outline segmented by B & P.

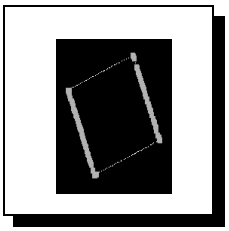

Figure 7: Rectangular outline segmented by P.

P & P algorithm delivers the same high descriptive accuracy with the other two algorithms by utilizing a smaller number of control points. Interestingly, our algorithm performs comparatively better as the shape of the targets becomes increasingly complex.

While in general, the proposed algorithm performed very well in a variety of shapes, there has been at least one instance where the algorithm exhibited a degrading performance. In Fig. 13 a wide-angle triangular shape is shown for which the P & P algorithm failed to locate one corner and one key flat point. Because of the nature of the shape, there is a large continous run of points with non-zero TCF values. That run is visualized as the grey part of the curve in Fig. 13. The run gives the maximum TCF value at point 1, where the corner segmentation point is placed. Unfortunately, the apex of the wide angle, which is perfectly legitimate to be a corner point, has a lower TCF value than point 1. Since both point 1 and the apex of the wide angle are on the same run, point 1 is chosen as the sole corner representative of the run. A similar explanation accounts for the loss of one key flat point.

5 SUMMARY

A new curve segmentation algorithm for locating control points for deformable-model-based tracking has been described. Comparatively to other curve segmentation algorithms, the proposed algorithm delivers the minimal amount of control points for the

Figure 8: A pedestrian target.

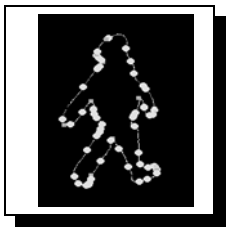

Figure 9: Pedestrian outline segmented by our algorithm.

Figure 10: Pedestrian outline segmented by B & P.

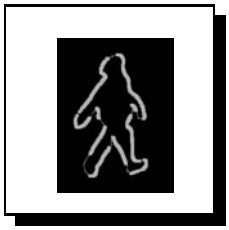

Figure 11: Pedestrian outline segmented by P & H.

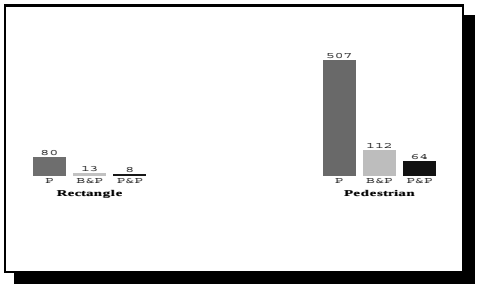

Figure 12: Performance in terms of number of control points for the three segmentation algorithms. The number of control points yielded in each case is shown at the top of the corresponding bar.

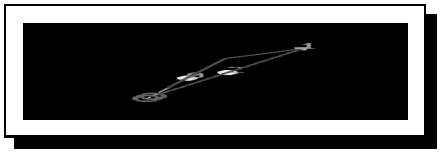

Figure 13: Failure case for the P & P algorithm. Point 1 is a corner point. Points 0 and 2 are key flat points. The little circles at the lower tip of the triangle mark the beginning and the end of the algorithmic trace.

same fine descriptive detail. Moreover, no parameters need to be fixed and the algorithm performs equally well for any type of curve. The algorithm achieves such results by segmenting the curve at its corner and some key flat points only. Its net effect is to distribute control points more densely at parts with high curvature change rate and more sparsely at parts with slow curvature change rate.

On one hand, the proposed algorithm completely automates the deformable-model-based robotic tracking and on the other hand optimally solves the contradictory requirements of using as few control points as possible for the deformable model, that still describe accurately the original figure.

References

[1] J. J. Brault and R. Plamondon. "Segmenting Handwritten Signatures at Their Perceptually Important Points." *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 15, pp. 953-957, 1993.

[2] P. A. Couvignou, N. P. Papanikolopoulos, and P. K. Khosla. "Hand-Eye Robotic Visual Servoing Around Moving Objects Using Active Deformable Models." In *Proceedings of the 1992 IEEE International Conference on Intelligent Robots and Systems*, pp. 1855-1862, 1992.

[3] M. Kass, A. Witkin, and D. Terzopoulos. "Snakes: Active Contour Models." *International Journal of Computer Vision*, Vol. 1, pp. 321-331, 1987.

[4] T. Pavlidis. *Structural Pattern Recognition*, pp. 168-184. Springer-Verlag, 1977.

[5] T. Pavlidis and S. T. Horowitz. "Segmentation of Plane Curves." *IEEE Transactions on Computers*, Vol. 23, pp. 860-870, 1974.

[6] M. J. Sullivan, C. A. Richards, C. E. Smith, O. Masoud, and N. P. Papanikolopoulos. "Pedestrian Tracking from a Stationary Camera Using Active Deformable Models." In *Proceedings of the Intelligent Vehicles '95 Symposium*, pp. 90-95, 1995.

[7] B. H. Yoshimi and P. K. Allen. "Visual Control of Grasping and Manipulation Tasks." In *Proceedings of the 1994 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pp. 575-582, 1994.