Stabilisation of Active Contours using Tangential Evolution: An Application to Tracking

V. Srikrishnan, Subhasis Chaudhuri, Sumantra Dutta Roy
Vision and Image Processing Lab, EE Dept., I.I.T Bombay

***Abstract*— Active contours are very widely used in computer vision problems. Their usage has a typical problem, that of bunching together of curve points. This becomes apparent especially when we use active contours for tracking leading to instability in curve evolution. In this paper, we propose a tangential term to stabilise the evolution while at the same time ensuring that the curve shape is not changed. The proposed method is simple and the computational overhead is minimal, while the results are good.**

I. Introduction

Active contours are very widely used in computer vision tasks like tracking and segmentation. A flurry of research was sparked off by the original paper of Kass and Witkin [1] which still continues. Active contours are simply connected closed curves which move so as to minimise some energy functionals. The minimisation yields the curve evolution equations and depending on the numerical implementation, contours have been classified as parametric active contour or geometric active contour. As their name suggests, parametric active contours are implemented using parametric curves like splines [2] or finite element method [3] in a Lagrangian framework. On the other hand, geometric active contours are implemented in an Eulerian framework using the level set methods [4] [5]. An interesting paper which links these two approaches is [6].

It is out of scope of this article to review the entire active contour literature, however we mention a few of the important works. The initial energy functional defined by Kass et al. [1] was based on image gradients while Ronfard [7] extended it to region based energy functionals. Malladi [8] introduced the level set method into the computer vision community. Another landmark paper is [9] which converted the initial gradient problem into that of finding a geodesic path in a Riemannian space defined by the image. They have used the level set method for implementation. Some of the other important works are [10] [11] [12] [13] [14].

The advantages and disadvantages of both these methods are well documented [15]. Briefly, the level set method representation allows topological change but has the disadvantage of being slow; converse is the case with parametric representation. In applications like tracking, which is our primary interest, topological changes seldom occur. Therefore we concentrate on parametric active contours only in this paper. We use a spline based implementation similar to that of [2].

In the next section, we describe the problem and discuss a few solutions which have been proposed in literature. After that, we describe our solution and finally we present the results and conclusions.

II. Notation

We first describe the notation used in this paper. A curve is denoted by $C(p,t)$, where p is the curve parameter and t is the artificial time parameter. Thus t parameterises a family of curves while p parameterises a single member of this family. The initial curve is $C(p,0)$ and the family of curves is obtained by evolving $C(p,0)$ as per some curve evolution equation. The local tangent and *inward* normal are denoted by $\mathbf{t}$ and $\mathbf{n}$ respectively. The curvature is denoted by κ and the arc length parameter by s. The quantity $g = |C_p|$, is interpreted as the speed of a particle on the curve. This quantity is a measure of the parameterisation of the contour.

The force at each point on the curve can be resolved into two components: along the local tangent and normal denoted by α and β, respectively. This is written as:

$$\frac{\partial C}{\partial t} = \alpha(p,t)\mathbf{t} + \beta(p,t)\mathbf{n}. \tag{1}$$

Given this equation, g varies as follows [16] [17]:

$$\frac{\partial g}{\partial t} = -g\kappa\beta + \frac{\partial \alpha}{\partial p}, \tag{2}$$

It is seen from the above equation that the curve speed function depends on both the components. On the other hand, it has been shown by researchers [17] that only the normal component of the force β influences the shape of the curve. The tangential component α reparameterises the curve. Based on this fact, most works have concentrated on constructing energy functions and paid attention to the normal term to speed up the convergence, increase the capture range etc. No specific efforts were made to give some shape to the tangential term or at best it got constructed as a side effect. This did not pose any problems as these works used level set methods. However, there are some typical problems with the Lagrangian implementation which is discussed next. We also discuss a few common solutions as well as some works which had a different approach.

III. Problems with Curve Evolution

Difficulties with Discrete curves : A well known problem with the parametric representation of curves is that during evolution the points on the curve bunch close together at certain regions and they space out elsewhere. This increases error in numerical approximation of curves measures like

tangent and curvature. In a spline implementation, although the tangents and normals are computed analytically and non uniform spacing of points is not a problem, in regions where the points come close together the control points also bunch together. This may lead to formation of discontinuities in the curve. Subsequently the normal is ill-defined, leading to formation of small local loops. These loops blow up in size and ultimately the curve degenerates. In regions where the points space out, the segmentation will of course be much poorer. This problem which is disturbing in segmentation problems, becomes intolerable in tracking. Therefore our aim in this work is to maintain a uniform spacing of points.

As an example, we show two frames from a tracking sequence of a hand. We use the final curve of the previous frame as the initialisation for the current frame. Figure1(a) shows the curve just after initialisation. The points on the curve are nearly equidistant. We use a minor modification of the region competition model [11] for tracking (this is explained in the next section). After four frames, as marked in figure (1(b)), the points accumulated in two regions are marked by red circles. In the very next frame, in figure (1(c)), we notice that small loops have formed in these regions. These loops blow up and the curve becomes unstable. The occurence of degeneracy depends partly on the motion direction. In the example shown, as the hand moves from the right to left, the points accumulate to the right and vice versa. Of course, the exact number of frames between initialisation and loop formation depends on different image sequences.

Few Solutions : In this section, we present a few possible approaches described in the literature to tackle this problem and discuss their limitations.

1) Reinitialisation of curve can be done either after a fixed number of frames or when the distance between successive control points falls below a certain threshold. As proposed in [2], this can be done by minimising the least squared distance between the current curve and the new curve while penalising the distance between the control points. However, this is not a very good solution because the shape of the curve would change during the re-positioning of the control points. The computation is also increased in checking the distances in each frame after every iteration.
2) Another ad-hoc solution is inserting or deleting points from the curve when the distance between them exceeds or falls below a certain threshold. This again is not a very good solution; the thresholds have to be set manually and in general is a naive procedure.
3) In a spline based implementation; we could also control the curve by deleting or inserting control points. Although algorithms exist for such a procedure; this solution is not natural, is specific to splines and is computationally expensive. Also, if we were to use the control points to represent the shape space, these operations would change the dimensions of the feature space.

The above methods are rather ad-hoc in the sense that they are methods to adjust Euclidean distance between points after they space out and do not actually try to prevent this phenomenon from occuring. Some better methods to obtain a more uniform point spacing have been proposed in [15] [16] [18]. We however postpone the discussion of these methods to the next section. This would enable us to compare immediately our method with these approaches.

IV. Proposed Method

We first qualitatively describe the cause for the bunching of the points on the curve and the control points. As mentioned previously, β controls the shape while α controls the parameterisation. It might be thought that if we set the tangential component α to zero; the curve would retain its parameterisation and be well behaved. It is seen from equation (2) that g depends on both the components of the force. Therefore, while reconstructing the curve with a discrete set of points the spacing between the points varies in an unpredictable manner. This leads to uneven spacing of points at certain portions of the curve and the consequent problems described in the earlier section.

In our approach we ensure curve stability by using a very simple equation to control g. It is a well known fact that the arc length is the desired parameterisation to describe the curve. This is an intrinsic description of the curve. Also, we note that when the curve is parameterised by its arc length, the curve speed function quantity g becomes equal to 1. We make use of this simple fact to control the curve. Though arc length parameterisation is most desirable; it cannot always be achieved in practice. This is because of the representation used. For example, when we use closed periodic B Splines to represent a curve, the parameter range is N_b, the number of basis functions. Obviously, it cannot be guaranteed that the length of the curve would always be equal to this or even close to this. Therefore, the next practical compromise would be to have g to be a constant K.

It is then natural to use equation (2) to force the curve towards the parameterisation which would make $g = K$. The left hand side of this equation predicts how g changes given β and α. We know the normal component β; this is obtained from minimising the energy function defined on the curve. Equation 2 can be rewritten as:

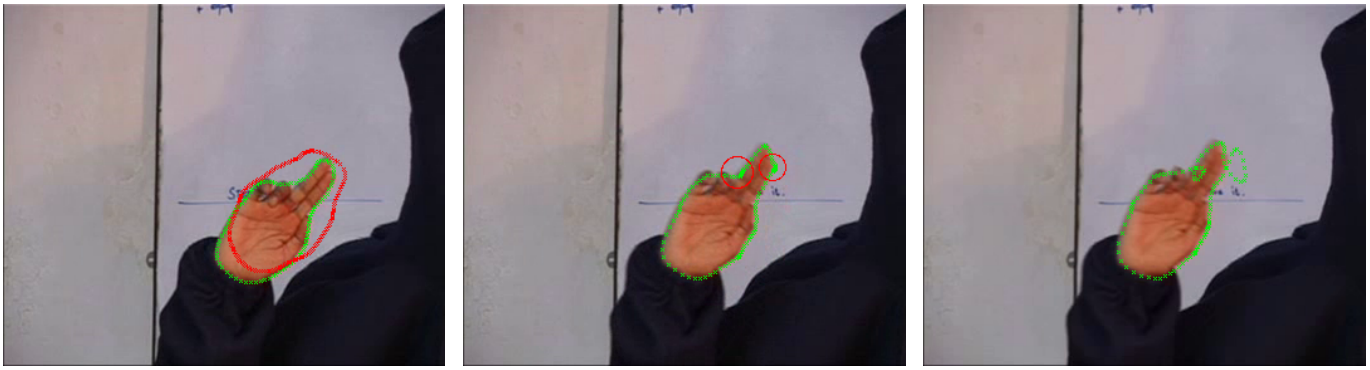
$$\frac{\partial \alpha}{\partial p} = \frac{\partial g}{\partial t} + g\kappa\beta, \tag{3}$$

Let us set:

$$\frac{\partial g}{\partial t} = K - g. \tag{4}$$

Qualitatively, at each point we try to find α by pushing g at that point to the constant K. We can set K by simply averaging it over the curve in the first frame. We obtain α by substituting equation (4) in equation (3) and then numerically solving the resulting ODE:

$$\frac{\partial \alpha}{\partial p} = \frac{\partial g}{\partial t} = K - g + g\kappa\beta \tag{5}$$



(a) Frame 26 (b) Frame 30 (c) Frame 31

Fig. 1: Curve Degeneration:Fig 1(a) Initial curve marked in red. Convergence of curve to target(in green). Fig 1(b) Illustration of loop formation (in red) due to target motion. Fig 1(c) Increasing loop size at the locations marked in red in previous frame.

After solving for $\alpha(p,t)$, we use the values in equation(1). This simple term gives very good stabilisation of the curve as we shall see in the next section on results.

It is interesting to compare the proposed energy term with that proposed in [15], [16] and [18]. In [15] the tangential energy term (for maintaining uniform parameterisation) is shown to be $\alpha = \frac{\partial g}{\partial p}$.

When the normal force component β has a smaller magnitude compared to the tangential part; the above force is equivalent to the diffusion of g along the curve as time progresses. The above assumption may not be strictly valid in regions of high curvature. The term proposed in this paper is better because it directly addresses the issue at hand. We do not make any assumptions in our work. In fact, we use β while computing α at each point.

In [16], the authors have proposed two terms for calculating α. In the first term, which is a non local term, α is obtained by solving the following ODE:

$$\frac{\partial \alpha}{\partial s} = \kappa\beta - < \kappa\beta > + (\frac{L}{g} - 1)\omega$$

where $< . >$ denotes averaging over the curve and $\omega = k_1 + k_2 < \kappa\beta >$, k_1 and k_2 are constants. The authors have shown that this term leads asymptotically to a uniform parameterisation. Note that there are two parameters to be fixed here.

In the second method, α is obtained by letting $\alpha = \partial_s \theta$, where $\theta = ln(\frac{g}{L})$. The rationale behind this term is that it is obtained as the tangential component of the solution of the intrinsic heat equation. The normal component of the solution is the mean curvature motion. However, it is well known that mean curvature motion is too slow in practice for convergence [19]. Therefore researchers speed up convergence by adding a normal term. Hence, we feel that the term might not perform so well in practice. One drawback of both these methods stable numerical implemention requires updation of g, curvature and tangent angle after α is calculated; only then are the curve points updated. In our method we can directly apply the calculated α in equation(1). Finally, in [18], the authors obtain the internal energy term by minimising the following:

$$E = \int_0^M (g^2 - c)^2 dp \tag{6}$$

where M depends on the representation used and c is proportional to the length of the curve. However, the above term will also cause a shrinkage of the curve. Therefore, although there may be a stabilisation of the curve, there is also a change in the shape of the curve because of the normal component. This is not at all a desirable side effect. In our method, there is no feedback term like this and hence we expect better results. The biggest disadvantage we feel in the non local term of [16] and [18] is that both require calculation of length of the curve or practically speaking, its numerical approximation. This can only be done by using a large number of points to approximate the curve. Therefore, the dimensions of various matrices will increase resulting in higher computation time. Our proposed method is free of such constraints.

As mentioned in the beginning of this paper, we have used the proposed term to stabilise the curve applied to tracking using the Region Competition model. We describe the tracking algorithm next.

V. Tracking Algorithm

We have used the region competition model [11] for tracking. This model was proposed for segmentation of an object in an image I using the statistical properties of the object. The idea is to move a point on the curve C in either inward or outward normal direction depending on the image properties of the point on the curve $I(C)$. We build histograms of the target and the background. These are denoted by p_T and p_B respectively. An image point lying on the curve is denoted by $I(C)$. Therefore, the probability of this pixel belonging to the target and the background is $p_T(I(C))$ and $p_B(I(C))$.

The curve evolution equation then is as follows [11]:

$$\frac{\partial C}{\partial t} = \mu\kappa\mathbf{n} + \log\left[\frac{p_B(I(C))}{p_T(I(C))}\right]\mathbf{n} \tag{7}$$

(a) Frame 31 (b) Frame 40

Fig. 2: Curve Stabilisation using proposed method. Even for rapid motions the curve remains stable.

The interpretation of the above equation is as follows; if the probability that the curve point belongs to the background is higher than the probability that it is a part of the target then the point moves in the inward direction and vice versa. This means that the initialisation is such that the curve should at least partially cover the target. This is a very common assumption in tracking. In [11] the authors have used parametric distributions to model the target and background; we however have used histograms because they are simple and fast. We ignore the bins of the histogram where the probability values are too low.

We extend the same model for tracking. We use the converged contour in the previous frame as the initialisation of the contour in the next image. We use histograms to model the target and background feature distributions in the RGB colour space. We generate the target histogram offline manually and generate the background dynamically in the following manner. The B Spline curve lies entirely within the convex hull of its control points and we assume that the target lies mostly within the region enclosed by the spline curve. however computing the convex hull is computationally expensive. We therefore find the biggest rectangular bounding box enclosing the curve and sample the image randomly outside this box. We can build a histogram of the whole image excluding the region inside the curve but in our work we found that about 3500 points from the image suffice for most purposes.

VI. Results

In this work, we have used B Splines [20] for implementation. The advantages of cubic splines and especially cubic B-Splines is well documented in literature [21]. Of course, other representations can also be used as the method proposed is quite independent of the choice of representation. We discretised equation(5) simply by using backward differences.

We observed that a initial smooth contour degenerates when there is a rapid motion of the target or there is a rapid shape change or a combination of both. We first implemented the proposed method on the same sequence as shown in figure 1 for the purpose of comparison. In this sequence the shape change is slow but the object moves fast. Not only at frame 31(figure 2) is the curve stabilised but also remains so till frame 40, figure(2(b)).

We next take a more difficult sequence where there is a combination of shape change and motion. Figures 3(a)and 3(b) shows one such sequence. We again note that the curve remains stable despite this combination of motion and shape change.

VII. Conclusions and Future Work

Parametric active contours are simpler to implement and much faster than level set methods; however their stability is always suspect. In this paper, we presented a simple method to stabilise parametric active contours. This can be used with any representation of contours. In one of the future works, we present a rigorous theoretical proof of the proposed equation.

References

[1] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *IJCV*, pp. 321–331, 1988.

[2] Menet, Saint-Marc, and Medioni, "Active contour models: Overview, implementation and application," pp. 194–199, 1990.

[3] L. Cohen and I. Cohen, "Finite-element methods for active contour models and balloons for 2-d and 3-d images," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 15, no. 11, pp. 1131–1147, November 1993.

[4] S.Osher and R.Fedkiw, *Level Set Method and Dynamic Implicit Surfaces*. Springer, 2003.

[5] J. A. Sethian, *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999.

[6] C. Xu, A. Yezzi, and J. L.Prince, "On the relationship between parametric and geometric active contours," in *Proc. 34th Asilomar Conference on Signals,Systems, and Computers*, Asilomar, October 2000, pp. 483–489.

[7] R. Ronfard, "Region based strategies for active contour models." *IJCV*, vol. 13, no. 2, pp. 229–251, October 1994.

[8] R. Malladi, J. A.Sethian, and B. C.Vemuri, "Shape modeling with front propagation: A level set approach," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 17, no. 2, pp. 158–175, 1995.

[9] V. Caselles, R. Kimmel, and G. Sapiro, "Geodesic active contours," *IJCV*, vol. 22, no. 1, pp. 61–79, 1997.

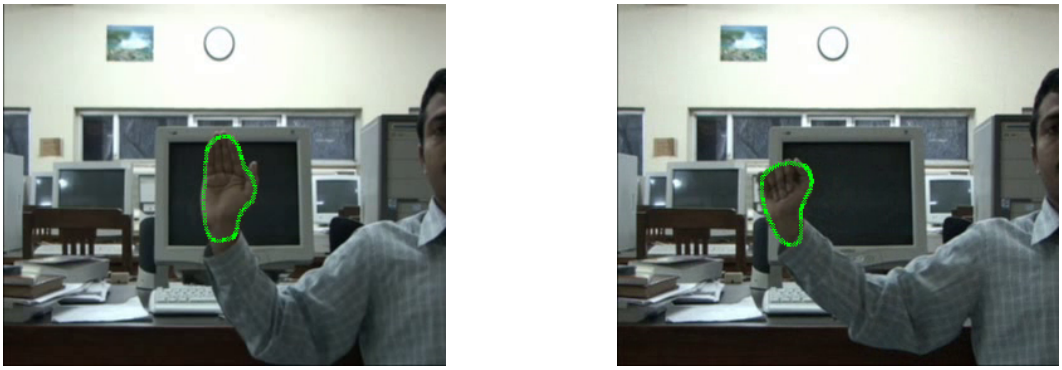
[10] C.Xu and J.L.Prince, "Snakes, shapes, and gradient vector flow," *IEEE Trans. Image Processing*, vol. 7, no. 3, pp. 359–369, 1998.

[11] S. C. Zhu and A. L. Yuille, "Region competition: Unifying snakes, region growing, and bayes/mdl for multiband image segmentation." *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, no. 9, pp. 884–900, 1996.

[12] T. F. Chan and L. A. Vese, "Active contours without edges," *IEEE Trans. Image Processing*, vol. 10, no. 2, pp. 266–277, 2001.

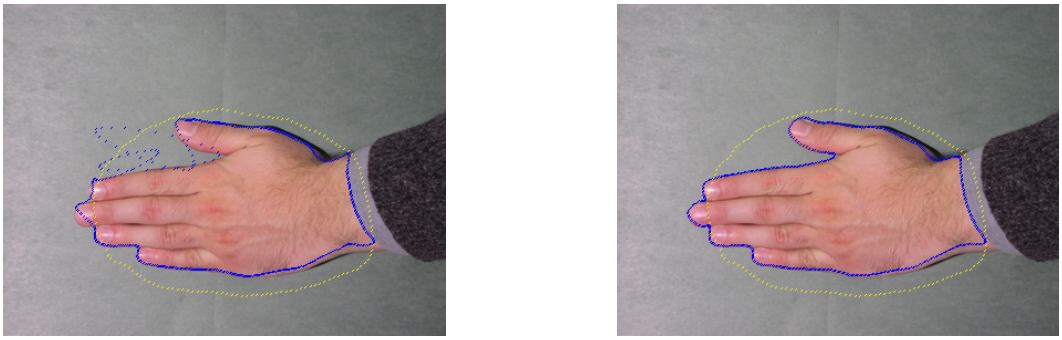
[13] N. Paragios and R. Deriche, "A PDE-based level-set approach for detection and tracking of moving objects," in *ICCV*, 1998, pp. 1139–1145.

[14] H. Delingette, "On smoothness measures of active contours and surfaces," in *IEEE VLSM 2001*, Vancouver, Canada, July 2001, pp. 43–50.



(a) Frame 76 (b) Frame 120

Fig. 3: Curve Stabilisation using proposed method. Even for rapid motions the curve remains stable.



(a) (b)

(c) (d)

Fig. 4: As the number of control point increases, the looping occurs. Using our term, the looping disappears and shape is segmented prperly.

[15] H. Delingette and J. Montagnat, "Shape and topology constraints on parametric active contours," *CVIU*, vol. 83, no. 2, pp. 140–171, 2001.

[16] K. Mikula and D. Sevcovic, "Computational and qualitative aspects of evolution of curves driven by curvature and external force," *Computing and Visualization in Science*, vol. 6, no. 4, pp. 211–225, 2004.

[17] B. B. Kimia, A. R. Tannenbaum, and S. W. Zucker, "On the evolution of curves via a function of curvature, I: The classical case," *JMAA*, vol. 163, no. 2, pp. 438–458, January 1992.

[18] M. Jacob, T. Blu, and M. Unser, "Efficient energies and algorithms for parametric snakes," *IEEE Trans. Image Processing*, vol. 13, no. 9, pp. 1231–1244, September 2004.

[19] K. Siddiqi, S. W. Zucker, Y. B. Lauzière, and A. Tannenbaum, "Area and length minimizing flows for shape segmentation." in *IEEE Conf. on CVPR*, 1997, pp. 621–627.

[20] D. F. Rogers and J. A. Adams, *Mathematical elements for computer graphics (2nd ed.)*. New York, NY, USA: McGraw-Hill, Inc., 1990.

[21] P. Brigger, J. Hoeg, and M. Unser, "B-spline snakes: a flexible tool for parametric contour detection." *IEEE Trans. Image Processing*, vol. 9, no. 9, pp. 1484–1496, 2000.