A fast and flexible multiresolution snake with a definite termination criterion

Nilanjan Ray*, Bhabatosh Chanda, Jyotirmay Das

Electronics and Communication Sciences Unit, Indian Statistical Institute, Calcutta 700035, India

Received 9 September 1999; accepted 4 April 2000

Abstract

This paper describes a fast process of parametric snake evolution with a multiresolution strategy. Conventional parametric evolution method relies on matrix inversion throughout the iteration intermittently, in contrast the proposed method relaxes the matrix inversion which is costly and time consuming in cases where the resulting snake is flexible. The proposed method also eliminates the input of snake rigidity parameters when the snake is flexible. Also, a robust and definite termination criterion for both conventional and proposed methods is demonstrated in this paper.

Keywords: Active contour; Parametric snake; Snake evolution; Snake rigidity parameters; Termination criterion; Multiresolution

1. Introduction

Snakes [1], or active contours, are curves defined within an image domain that can move under the influence of internal forces coming from within the curve itself and external forces computed from the image data. The internal and external forces are defined so that the snake will conform to an object boundary or other desired features within an image. Snakes are widely used in many applications, including edge detection [1], shape modeling [2,3], segmentation [4,5], and motion tracking [4,6].

There are, in general, two types of active contours presently: *parametric* [1] and *geometric* [7–9]. There have been several attempts to increase the capture range of parametric snakes through *balloon* model [10], through multiresolution scheme [11] and more recently through gradient vector flow method [12]. Comparatively only a few significant publications are there on the improvement of parametric snake evolution. Originally, Kass et al. introduced the method of finite difference [1] and subsequently Cohen et al. introduced finite element method [13]. This paper presents a fast snake evolution method coupled with a definite termination criterion for parametric snakes. A conventional evolution method of parametric snakes relies on inverting the rigidity matrix of the snake where one can apply any o(n) inversion method [4]. But during the whole evolution process this rigidity matrix changes as the number of snake points (*snaxels*) or the rigidity parameter changes [1]. So there is a need to invoke the matrix inversion routine quite frequently during the process [4]. This requirement naturally makes the process slow. Our proposed method bypasses this matrix inversion method in case of flexible snakes. In addition, we have proposed a multiresolution approach to speed up our algorithm further. Though multiresolution method as proposed by Leroy et al. [11] addresses the problem of initialization, it lacks the movement of snakes through different resolutions. In comparison, our proposed scheme explains the movement of snake through different resolutions successfuly.

Termination of the parametric snake evolution is a complicated process as the snake begins to oscillate at or r the solution [4]. We have also proposed a termination criterion that is based on the position of the snake and the external force at that position. This criterion looks for change of direction of the external force on the snaxels. The uniqueness of this termination condition is that it is well suited for the general snake equation (Euler equation) and also to the cases where external forces may not be specified in terms of gradient of a potential field [12].

The paper is organised as follows. We describe the necessary background briefly in Section 2. In Section 3, proposed multiresolution fast snake evolution process and the termination criterion are presented. Implementation of the algorithm and the experimental results are discussed in Section 4. Finally, concluding remarks are given in Section 5.

2. Background

A conventional snake is a parametric curve $C(s) = [x(s), y(s)]$, $s \in [0,1]$ that minimizes the energy functional [12]

$$E_{\text{snake}} = \int_0^1 \frac{1}{2}\{\alpha|C'(s)|^2 + \beta|C''(s)|^2\} + E_{\text{ext}}(C(s))\,\mathrm{d}s, \tag{1}$$

where the curly bracketed term of the integral represents internal energy of the snake and E_{ext} represents the external or image energy term. α and β of internal energy term represent, respectively, resistance to tension and that to the bending of the snake. Both of them are non-negative quantities. $C'(s)$ and $C''(s)$ denote the first and second derivatives of the curve $C(s)$ with respect to the parameter s, respectively. The external energy typically arises out of the lines/boundaries in a binary image and the edges in a gray level image. They may be expressed by $\pm G_\sigma(x,y)f(x,y)$ and $-|\nabla G_\sigma(x,y)f(x,y)|$, respectively [12], where $f(x,y)$ is a two-dimensional image and $G_\sigma(x,y)$ is a two-dimensional Gaussian function with standard deviation σ, and ∇ is the gradient operator.

The energy functional E_{snake} is minimised by the variational approach to obtain the following Euler equation [12]

$$\alpha C''(s) - \beta C''''(s) - \nabla E_{\text{ext}} = 0 \tag{2}$$

under the assumption that α and β do not vary over the curve.

To find a solution to Eq. (2), the snake is made dynamic by treating C as a function of time t as well as s, i.e., $C(s,t)$. Then the partial derivative of C with respect to t is set equal to the left-hand side of Eq. (2) to obtain

$$C_t(s,t) = \alpha C''(s) - \beta C''''(s) - \nabla E_{\text{ext}}. \tag{3}$$

When the solution $C(s,t)$ stabilizes, the term $C_t(s,t)$ vanishes and we achieve a solution to Eq. (2). If we take $[u(x(k),y(k)), v(x(k),y(k))] = -\nabla E_{\text{ext}}$, k being discretised over s, Eq. (3) essentially becomes

$$\begin{aligned}\frac{x^{t+\delta t}(k) - x^t(k)}{\delta t} &= \alpha[x(k\oplus 1) - 2x(k) + x(k\ominus 1)] \\ &\quad - \beta[(x(k\oplus 2) - 2x(k\oplus 1) + x(k)) \\ &\quad - 2(x(k\oplus 1) - 2x(k) + x(k\ominus 1)) \\ &\quad + (x(k) - 2x(k\ominus 1) + x(k\ominus 2))] \\ &\quad + u(x(k), y(k))\end{aligned} \tag{4}$$

and

$$\begin{aligned}\frac{y^{t+\delta t}(k) - y^t(k)}{\delta t} &= \alpha[y(k\oplus 1) - 2y(k) + y(k\ominus 1)] \\ &\quad - \beta[(y(k\oplus 2) - 2y(k\oplus 1) + y(k)) \\ &\quad - 2(y(k\oplus 1) - 2y(k) + y(k\ominus 1)) \\ &\quad + (y(k) - 2y(k\ominus 1) + y(k\ominus 2))] \\ &\quad + v(x(k), y(k))\end{aligned} \tag{5}$$

where $\oplus$ and $\ominus$ are modulo n addition and subtraction respectively, n being the total number of snaxels. These modulo operations or wrap up operations are introduced to take care of closed snakes. These two equations can be written using matrix–vector notation in the following compact forms taking the time step $\delta t = 1$ and taking $\mathbf{x}$ and $\mathbf{y}$ as n-dimensional column vectors (denoting snaxels' positions)

$$\mathbf{x}^{t+1} = \mathbf{x}^t - A\mathbf{x}^t + u(\mathbf{x}^t, \mathbf{y}^t) \tag{6}$$

and

$$\mathbf{y}^{t+1} = \mathbf{y}^t - A\mathbf{y}^t + v(\mathbf{x}^t, \mathbf{y}^t), \tag{7}$$

where

$$A = \begin{bmatrix} c & b & a & & & a & b \\ b & c & b & a & & & a \\ a & b & c & b & a & & \\ & & & \ddots & & & \\ & & a & b & c & b & a \\ a & & & a & b & c & b \\ b & a & & & a & b & c \end{bmatrix}$$

is an $n \times n$ symmetric pentadiagonal matrix with $a = \beta$, $b = -(4\beta + \alpha)$ and $c = 6\beta + 2\alpha$. For the sake of numerical stability, Eqs. (6) and (7) are written as [1]

$$\mathbf{x}^{t+1} = \mathbf{x}^t - A\mathbf{x}^{t+1} + u(\mathbf{x}^t, \mathbf{y}^t) \tag{8}$$

and

$$\mathbf{y}^{t+1} = \mathbf{y}^t - A\mathbf{y}^{t+1} + v(\mathbf{x}^t, \mathbf{y}^t), \tag{9}$$

so that we obtain $\mathbf{x}$ and $\mathbf{y}$ by inverting the positive-definite matrix $(A + I)$, I being the identity matrix. Iterations are carried on until the two solutions stabilize. Throughout the iteration process the matrix A changes quite often and this change may take place or in its dimension or in its elements or both. In fact, A changes while

(1) α and β are allowed to vary keeping number of snake points, n, constant.
(2) n is allowed to vary keeping α and β constant.

The second situation occurs when maximum distance between two adjacent snaxels is not allowed to exceed a prespecified limit, say, 0.5–1.5 units [12]. Naturally, the drawback of Eqs. (8) and (9) is the inherent slowness owing to frequent matrix inversion. The slowness is more imposed by the fact that updations of $\mathbf{x}$ and $\mathbf{y}$ require multiplications with all the elements of the matrix $(A + I)^{-1}$. For a flexible snake, i.e., for α and β having very small values, the matrix $(A + I)^{-1}$ becomes sparse. So multiplications with all the elements of the matrix are definitely wastage of time.

3. Proposed method

From the foregoing discussion, it is evident that the existing conventional snake evolution method is inherently slow. Here we propose a fast method for flexible snake evolution coupled with a definite criterion for its termination.

A closer look at Eqs. (6) and (7) reveals that they are force balance equations. The terms $-A\mathbf{x}^t$ and $-A\mathbf{y}^t$ are the x and y components of internal or snake force and the terms $u(\mathbf{x}^t, \mathbf{y}^t)$ and $v(\mathbf{x}^t, \mathbf{y}^t)$ are the corresponding components of external force at iteration t. They are added vectorially to give a net unbalanced force which pushes the snake along its direction. When these two forces balance each other there is no further movement or evolution of snake. We first discuss our proposed method of snake evolution and subsequently discuss the termination criterion.

3.1. Fast evolution

One cannot directly iterate for $\mathbf{x}^{t+1}$ and $\mathbf{y}^{t+1}$ from Eqs. (6) and (7) because they are numerically unstable. On the other hand, working with them helps us to avoid $o(n)$ matrix inversion and $o(n^2)$ multiplications at each iteration. In order to make the process numerically stable, we make the external force $[u, v]$ unity at each grid point and we apply the same treatment for the internal force $[-A\mathbf{x}^t, -A\mathbf{y}^t]$ as well. By doing this we put a limit on the net unbalanced force.

A further analysis of the internal force has helped us to get rid of the snake parameters α and β. A slight manipulation of terms shows that the orthogonal components of the internal force on the snaxel $[x_k, y_k]$ are $(4\beta + \alpha)((x_{k\ominus 1} - x_k) + (x_{k\oplus 1} - x_k)) + ((x_k - x_{k\ominus 2}) + (x_k - x_{k\oplus 2}))$ and $(4\beta + \alpha)((y_{k\ominus 1} - y_k) + (y_{k\oplus 1} - y_k)) + \beta((y_k - y_{k\ominus 2}) + (y_k - y_{k\oplus 2}))$, respectively. So the internal force essentially depends on the Euclidean distance between adjacent snaxels which lie in the range 0.5–1.5. These internal forces are shown in Fig. 1. In order to get rid of the trouble of supplying appropriate value for α and β we take for the kth snaxel: $4\beta + \alpha = 1/\sqrt{(d^2_{k,k\ominus 1} + d^2_{k,k\oplus 1})}$ and $\beta = 1/\sqrt{(d^2_{k,k\ominus 2} + d^2_{k,k\oplus 2})}$, where $d_{i,j}$ is the distance between ith and jth snaxels. These denominators will never be zero as we impose the distance constraint on the snaxels as mentioned before. Taking the parameters this way enables us to get rid of the user inputs for snake parameters, viz., α and β.

3.2. Multiresolution approach

We further speed up the evolution process by a multiresolution method. Suppose we scale down the original image by half in both directions. Then we let the snake evolve on this reduced image with the foregoing method. Once the snake stabilizes at the current resolution, we go back to the original scale and let the snake evolve until it stabilizes at this resolution. This proposed multiresolution method can be cascaded for more than two stages. Naturally at any resolution, the snake starts evolving with the snaxels obtained from the immediate lower resolution mapped to appropriate coordinate positions. The reasons behind the further speed up of multiscale method are enumerated below:

- In the coarse resolution number of snaxels is low and, hence, number of operations is less.
- In the coarse resolutions intricate details are smoothed out and, hence, it requires fewer iterations for termination.

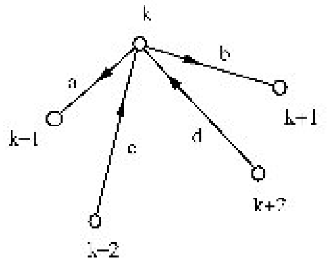

Fig. 1. a, b, c, d are internal forces on kth snaxel. Arrowheads are showing force directions. a, b are proportional to $4\beta + \alpha$ and c, d are proportional to β.

- As one goes from coarse to finer grids, in the finer grid the snake is already very close to the solution, and it requires only a few iterations to stabilize. Moreover, at times it requires no further iterations if the region boundary sought for is sufficiently smooth at its original resolution.

The process is indeed stable. It cannot generate spurious details after the snake settles. This is because we are going from coarse-to-fine grids. In coarse grids the finer details are smoothed out, so the evolved and settled snake will be sufficiently smooth at this stage. As one goes to the final resolution, one can think that the snake (sufficiently smooth) is initialized very close to the solution, as if the previous steps did not exist. It should be noted that when there is a thin line segment in an image as in Fig. 2, it may be difficult to get the same at half scale. To surmount this problem we develop the external guiding force (u, v) on the original image itself, and whenever we go to lower resolution external guiding force (u, v) at that resolution is obtained by means of bi-linear interpolation of scaled down external guiding force of the original image.

3.3. *Definite termination*

Eqs. (6) and (7) suggest that the snake stabilizes or, in other words, the solution is found when $x^{t+1} = x^t$ and $y^{t+1} = y^t$. Unfortunately, we observe that as a snake reaches very close to a solution in the discrete domain it starts oscillating. The reason may be described as follows. The external force pushes the snake to the region boundary from, say, outside. So a snaxel already arrived at the vicinity of the solution may cross the boundary due to push as well as discrete coordinate value. As it crosses the boundary it experiences an external force opposite to its direction of movement so far and hence it goes back. The internal force is always playing and is trying to make it smooth. Absolute equilibrium between the internal and the external forces is a rare situation. However as we always force the internal and the external forces to be unit vectors, most of the snaxels at the solution crosses the region boundary with to and fro motion. And if at the solution the snake is further allowed to evolve then the number of oscillating snaxels increases as our experiment suggests. Based on this observation we suggest a termination criterion as stated below.

Apparently our task is to count number of snaxels moving back and forth to terminate the snake evolution process. However, if we just observe the to and fro movement of a snaxel, we may come up with a wrong conclusion that this snaxel has reached the solution. This is because a snaxel can well move back and forth due to its internal force even if it has not reached the solution. So we observe whether or not a snaxel is experiencing external force with alternating sign in a sequence of iterations. This is a more robust criterion as the direction of the external force changes only across the boundary. For this purpose in our implementation we observe each snaxel for previous m (in our experiment $m = 3$) iterations, and if any of u and v components changes its sign at the snaxel locations, we conclude that this snaxel has reached the solution. In order to take care of possibly a very small number of snaxels in equilibrium, we also look for the snaxels at which the internal and external forces exactly balance each other and include them while counting oscillating snaxels. Finally, if this count exceeds P% of total number of snaxels then the snake evolution process is terminated. Naturally P is a user-specified parameter.

4. Results and discussion

We have tested our proposed algorithm on a large number of synthetic and real images. Here we present only a few results obtained through the proposed multiresolution method. We also present the results of conventional evolution method as a basis of comparison with our method.

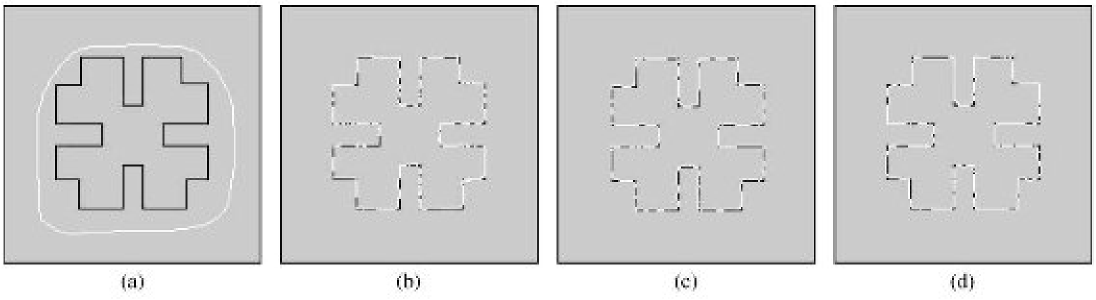

Fig. 2. (a) Initialized snake on polygon, (b) snake evolved with inverse method, (c) snake evolved with our proposed method at single resolution, (d) snake evolved with our proposed method in multiresolution scheme. $P = 99$ in all three cases.

Table 1
Comparative results of conventional and proposed method of snake evolution on polygon image with $P = 99$, HS = Half Scale, FS = Full Scale

	Conventional method	Our proposed method	
		Single resolution scheme	Multiresolution scheme
Iterations	174	186	111 (HS) + 6 (FS)
Time (in s)	476	2.2	1.815 (HS) + 0.2 (FS)
Oscillation	99.5%	99.2%	99.3% (HS) + 99.3% (FS)

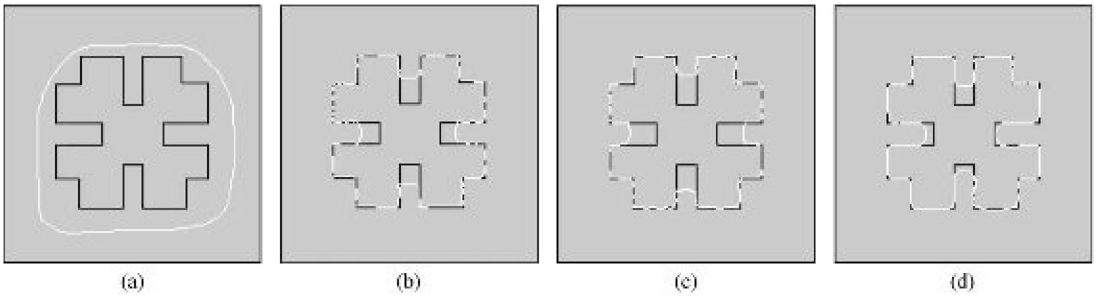
Fig. 3. (a) Initialized snake on polygon, (b) snake evolved with inverse method, (c) snake evolved with our proposed method at original resolution, (d) snake evolved with our proposed method at half-resolution. Number of iterations is 90 in all three cases.

Table 2
Comparative results of conventional and proposed methods of snake evolution on polygon image with 90 iterations

	Conventional method	Our proposed method	
		At original resolution	At half resolution
Iterations	90	90	90
Time (s)	127.1	1.082	0.926
Oscillation	91.1%	88.9%	92.8%

In all these experiments we have used GGVF as the guiding force for snake, with $K = 0.0005$ [14] and 500 iterations. For the conventional snake evolution we have used $\alpha = 0.6$ and $\beta = 0.2$ in all cases. We have implemented all our programs using JDK1.2 with Java Advanced Imaging and Java Media Framework.

As a test object we have taken here a polygon having a number of deep concavities and sharp corners as shown in Fig. 2(a). The original object with initialized snake and results of three different methods, namely conventional method, our proposed method and proposed multiresolution method, are shown in Fig. 2. In Table 1 we have given a comparison of these three methods of snake evolution using the proposed termination criterion with $P = 99\%$. It is clear from the figures that our proposed termination criterion has worked successfully (i.e., terminated the evolution of snake at an acceptable solution) with all three methods. From this result we also observe that the total time as well as number of iterations are the least in multiresolution scheme. Secondly, though number of iterations required by our proposed method in single resolution scheme is more than that of conventional method, time requirement of our method is remarkably less compared to that of conventional method because of the absence of large matrix inversion.

To compare the behaviour of snake evolution at half scale with that at the original scale and with that in the conventional method we have kept a number of iterations constant in all three cases and let the snake evolve on the same polygon object with the same snake initialization. Resulting images along with initial condition are shown in Fig. 3. Comparison of the three methods is given in Table 2. As is seen from the figures the snake in the proposed multiresolution scheme has approached closest to the solution among the three methods. Percentage of oscillation is also the highest in multiresolution scheme, as expected. Obviously, the time required is much less at half resolution, because at half resolution the evolution process deals with fewer snaxels.

We have carried out the above experiments with a smooth object as shown in the Fig. 4(a). Resulting images are shown in the Fig. 4 with $P = 90\%$.

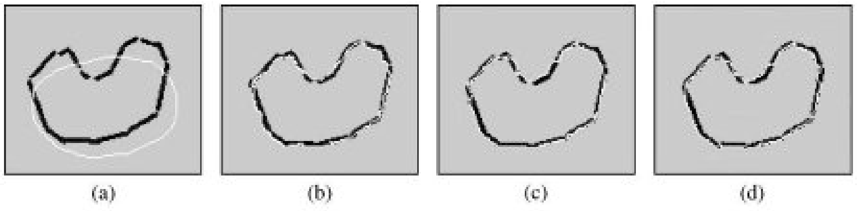

Fig. 4. (a) Initialized snake on polygon, (b) snake evolved with inverse method, (c) snake evolved with our proposed method at single resolution, and (d) snake evolved with our proposed method in multiresolution scheme. $P = 90$ in all three cases.

Table 3
Comparative results of conventional and proposed method of snake evolution on a smooth object with $P = 90$, HS = Half Scale, FS = Full Scale

	Conventional method	Our proposed method	
		Single resolution scheme	Multiresolution scheme
Iterations	24	45	12 (HS) + 3 (FS)
Time (s)	7.15	0.227	0.07 (HS) + 0.1 (FS)
Oscillation	91.9%	91.6%	90.0% (HS) + 91.1% (FS)

Table 4
Comparative results of conventional and proposed method of snake evolution on a smooth object with 15 iterations

	Conventional method	Our proposed method	
		At original resolution	At half resolution
Iterations	15	15	15
Time (s)	4.15	0.1	0.3
Oscillation	85.10%	30.83%	100%

Comparison among the results is presented in Table 3 which is similar to Table 1. As expected the number of iterations and the total time of evolution are the least in multiresolution scheme. Once again this result establishes that the termination criterion has behaved uniformly with all three methods. Next, the behaviour of snake evolution on this smooth object due to conventional method as well as our proposed methods are presented where the number of iterations is fixed at 15. Table 4 shows these results and Fig. 5 shows corresponding images with initial position of the snake. We have achieved 100% oscillation with half resolution using the proposed method.

The proposed multiresolution method is used to detect object's boundary in real images — especially detecting cloud contours in satellite images. An example follows. On a Meteosat IR band image we have run our method to capture a cloud patch as shown in Fig. 6(a) along with the initialized snake. Result of our proposed method using multiresolution scheme is shown in Fig. 6(b) with $P = 50$. Note that only two different resolutions: the original and its half are used. For comparison we also present the result of the conventional method in Fig. 6(c). Corresponding results are given in Table 5.

It is evident from the results that our proposed multiresolution method performs much better than conventional snake evolution method. The results also demonstrate that the proposed termination criterion works both in conventional as well as in proposed multiresolution paradigm.

5. Conclusion

This paper presents a fast implementation of parametric snake evolution coupled with a robust termination criterion. Our principal effort is to eliminate the $o(n)$ matrix inversion and $o(n)$ multiplications in flexible parametric snake evolution. This fast process may help in time critical processes like tracking cyclones from satellite images, tracking movement and change in cells in biomedical images, where flexibility of the active contour is a desirable property.

Termination criterion is based on the image force that guides the snake. As a result a good quality snake can always be achieved for a wide range of parameter values. Secondly, the parameter P ensures the termination of

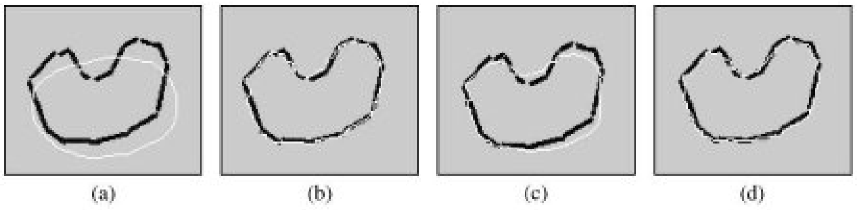

Fig. 5. (a) Initialized snake on polygon, (b) snake evolved with conventional method, (c) snake evolved with our proposed method at original resolution, (d) snake evolved with our proposed method at half-resolution. Number of iterations is 15 in all three cases.

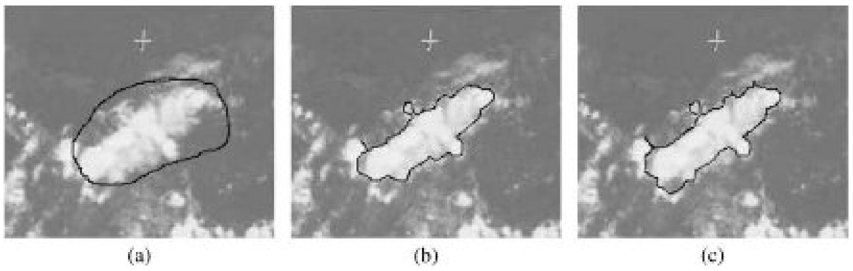

Fig. 6. (a) Initialized snake on Meteosat image, (b) snake evolved with our proposed method, (c) snake evolved with conventional method.

Table 5
Comparison between conventional snake evolution method and our proposed scheme on Meteosat cloud image

	Conventional method	Multiresolution scheme
Iterations	45	30 (half scale) + 15 (full scale)
Time (s)	46.91	0.31 (half scale) + 0.17 (full scale)
Oscillation	52.0 %	69.3 % (half scale) + 52.40 % (full scale)

evolution process within a reasonably small time. However, value of P plays a critical role when the object boundary to be captured is discontinuous. These shortcomings may be removed if total energy of the snake is also monitored along with the proposed criterion.

Acknowledgements

The authors acknowledge the Indian Space Research Organisation (ISRO) for providing funds for carrying out the research and the Director, Indian Statistical Institute for providing the necessary infrastructure. The authors also acknowledge Prof. N.R. Pal and Dr. D.P. Muhkerjee for their valuable comments and suggestions.

References

[1] M. Kass, A. Witkin, D. Terzopoulos, Snakes: Active contour models, Int. J. Comput. Vis. 1 (1987) 321–331.

[2] D. Terzopoulos, K. Fleischer, Deformable models, Vis. Comput. 4 (1988) 306–331.

[3] T. McInerney, D. Terzopoulos, A dynamic finite element surface model for segmentaion and tracking in multidimensional medical images with application to cardiac 4D image analysis, Comput. Med. Imag. Graph. 19 (1995) 69–83.

[4] F. Leymarie, M.D. Levine, Tracking deformable objects in the plane using an active contour model, IEEE Trans. Pattern Anal. Machine Intell. 15 (1993) 617–634.

[5] R. Durikovic, K. Kaneda, H. Yamashita, Dynamic contour: A texture approach and contour operations, Vis. Comput. 11 (1995) 277–289.

[6] D. Terzopoulos, R. Szeliski, Tracking with Kalman snakes, in: A. Blake, A. Yuille (Eds.), Active Vision, MIT Press, Cambridge, MA, 1992, pp. 3–20.

[7] V. Caselles, F. Catte, T. Coll, F. Dibos, A geometric models for active contours, Numer. Math. 66 (1993) 1–31.

[8] R. Malladi, J.A. Sethian, B.C. Vemuri, Shape modelling with front propagation: A level set approach, IEEE Trans. Pattern Anal. Machine Intell. 17 (1995) 158–175.

[9] V. Caselles, R. Kimmel, G. Sapiro, Geodesic active contours, Proceedings of the 5th International Conference on Computer Vision, 1995, pp. 694–699.

[10] L.D. Cohen, On active contour models and balloons, CVGIP: Image Understanding 53 (2) (1991) 211–218.

[11] B. Leroy, I. Herlin, L.D. Cohen, Multi-resolution algorithms for active contour models, Proceedings of the 12th Internal Conference on Analysis and Optimization of Systems, 1996, pp. 58–65.

[12] C. Xu, J.L. Prince, Snakes, Shapes, and Gradient Vector Flow, IEEE Trans. Image Processing 7 (1998) 359–369.

[13] L.D. Cohen, I. Cohen, Finite element methods for active contour models and balloons for 2-D and 3-D images, IEEE Trans. Pattern Anal. Machine Intell. 15 (1993) 1131–1147.

[14] C. Xu, J.L. Prince, Generalized gradient vector flow external forces for active contours, Signal Processing 71 (1998) 131–139.

About the Author—NILANJAN RAY received the B.E. degree in mechanical engineering in 1995 from Jadavpur University, Calcutta, India and the M.Tech. degree in computer science in 1997 from Indian Statistical Institute, Calcutta, India.

Currently, he is working as a research assistant in Oklahoma Imaging Laboratory, Oklahoma State University, USA, under the supervision of Dr. S.T. Acton. His research interests include image processing through anisotropic diffusion, level set methods, snake evolution techniques and PDEs.

About the Author—BHABATOSH CHANDA Born in 1957. Received B.E. in Electronics and Telecommunication Engineering and Ph.D. in Electrical Engineering from University of Calcutta in 1979 and 1988, respectively. Received "Young Scientist Medal" of Indian National Science Academy in 1989 and "Computer Engineering Division Medal" of the Institution of Engineers (India) in 1998. He is also the recepient of UN fellowship, UNESCO-INRIA fellowship and fellowship of National Academy of Science, India during his career. He worked at Intelligent System lab, University of Washington, Seattle, USA as a visiting faculty from 1995 to 1996. He has published more than 50 technical articles. His research interest includes image processing, pattern recognition, computer vision and mathematical morphology. Currently he is working as a professor in Indian Statistical Institute, Calcutta, India.

About the Author—JYOTIRMAY DAS received the M. Tech. degree from the Institute of Radio Physics and Electronics and Ph.D. degree both from the University of Calcutta.

His current research interests include Atmospheric pattern analysis and remote sensing applications. He is currently the professor and head of the Electronics and Communication Sciences Unit of Indian Statistical Institute, Calcutta.