Knowledge and Information Systems (2001) 3: 374–387

Knowledge and Information Systems

Short Paper

SAFE: An Efficient Feature Extraction Technique

Ujjwal Maulik[1], Sanghamitra Bandyopadhyay[2] and John C. Trinder[3]

[1]Department of Computer Science and Technology, Kalyani Government Engineering College, Kalyani, India
[2]Machine Intelligence Unit, Indian Statistical Institute, Calcutta, India
[3]School of Geomatic Engineering, University of New South Wales, Sydney, Australia

Abstract. This paper proposes an efficient window-based semi-automatic feature extraction technique which uses simulated annealing for minimizing the energy of an active contour within a specified image region. The energy is computed based on a chamfer image, in which pixel values are a function of distance to image edges. A user places a number of control points close to the feature of interest. B-spline fitted to these points provides an initial approximation of the contour. A window containing both the initial contour and the feature of interest is considered. The contour with minimum energy inside the window provides the final delineation. Comparison of the performance of the proposed algorithm with traditional *snake*, a popular feature extraction technique based on energy minimization, demonstrates the superiority of the SAFE technique.

Keywords: Active contour; B-spline; Edge detection; Energy minimization; Feature extraction; Simulated annealing

1. Introduction

Feature extraction (Barzohar and Copper, 1996; Merlet and Zerubia, 1996) from remotely sensed imagery has been an area of active research in computer vision and digital photogrammetry. The use of active contours, *snakes*, was introduced in Kass et al. (1988), to identify region boundaries based on the concept of energy minimization. An energy value associated with each contour was designed to be small when the contour was near the object. The energy had both extrinsic and intrinsic components. The extrinsic components were derived from underlying

Received 18 August 1999
Revised 25 October 2000
Accepted 8 December 2000

image characteristics (e.g, edge strength). Intrinsic components (e.g., elasticity) imposed regularity of the curve shape. Given an initial state, equations of motion were solved to find its final state. The main disadvantage of *snakes* is that the evolution is dependent upon its initial state, i.e., the user delineation, as it tends to gets stuck at a local energy minimum near its initial location in the search space. Moreover it also suffers from the computational complexity of deterministic dynamical systems with many degrees of freedom. Since the concept of energy minimization embedded in the active contour model is analogous to the principles of simulated annealing (SA) (Kirkpatrik et al., 1983), application of SA for feature extraction seems appropriate.

Simulated annealing, a well-known optimization tool, mimics the principles of the annealing procedure which is a physical process where a crystal is cooled down from the liquid to the solid phase. If the cooling is done slowly enough, the energy state of the crystal at the end will be very close to its minimum value. Simulation of this physical cooling may be done with the Metropolis algorithm. Simulated annealing has a wide range of applications in diverse areas like image segmentation, pattern classification, edge detection etc.

In this article we propose such a simulated annealing-based Semi-Automatic Feature Extraction technique called SAFE. It is called semi-automatic since the user, as in the *snake* model, is required to provide an initial rough estimate of the feature of interest. It is important to note that a given image may have several structures resembling the feature of interest (e.g., linear feature). It is therefore essential to approximately specify the structure in which the user is interested. Moreover, in order to restrict the search space of SA, we have incorporated a windowing approach in the SAFE algorithm. The proposed method has two phases. In the initial phase appropriate image-processing techniques are applied and the energy image is computed. Subsequently the operator is required to provide a number of control points which is a rough estimate of the feature of interest. Finally, B-splines are used to fit the initial, user-delineated, points and a window containing both the spline and the feature of interest is considered. In the second phase of SAFE, SA is used to minimize the energy within the window. The final curve with minimum energy thus delineates the feature. Since SA is known to converge to the optimal solution, it will be able to overcome the limitation of traditional implementation of *snakes* which often gets stuck at suboptimal solutions.

2. Contour Estimation By Energy Minimization in the *Snake* Model

2.1. Minimization of the Total Energy

Total energy of an active contour or *snake*, s, can be defined as the sum of *extrinsic* and *intrinsic* energies, and may be expressed by a parametric representation, $v(s) = (x(s), y(s))$, $x(s)$ and $y(s)$ being the coordinates of s, as

$$E = \int_{s_0}^{s_1} E(v(s))\mathrm{d}s = \int_{s_0}^{s_1} [E_g(v(s)) + E_p(v(s)) + E_c(v(s), v_0(s))]\mathrm{d}s \tag{1}$$

Normally, the intrinsic or geometric energy E_g, derived from the geometric

constraints of the object model is based on the first derivative (v_s) and the second derivative (v_{ss}) of the function $v(s)$. It is defined by $E_g = \gamma|v_s(s)|^2 + \beta|v_{ss}(s)|^2$, where γ and β are constants that control the influence of the geometric energy against the photometric energy. The extrinsic energy comprises photometric energy E_p, that constrains the contour to approach the feature of interest, and control energy E_c, which constrains the difference between the contour $v(s)$ and the initial curve $v_0(s)$. E_p, derived from the image, depends on the type of feature to be extracted. For a narrow linear feature, it can be the square of intensity values ($I(x, y)$) of the image, multiplied by a positive or negative constant (w) for lighter or darker features respectively, i.e., $E_{pl} = w|I(x, y)|^2$. For a step edge, it can be calculated as $E_{pe} = -|\delta I(x, y)|^2$, thus helping the contour to move towards the image points with high gradient values while minimizing the energy.

In this article, the features in the image are defined by morphological tools for narrow features and the Canny operator (Canny, 1986) for step functions. The *energy image* is defined by a Chamfer image, derived from the feature image, in which the pixel values relate to their closeness to any surrounding edge. Letting $E_{ext} = E_p + E_c$, equation (1) becomes

$$E = \int_{s_0}^{s_1} [\gamma|v_s(s)|^2 + \beta|v_{ss}(s)|^2 + E_{ext}(v_s(s)) = \int_{s_0}^{s_1} F(s, v, v_s, v_{ss})\mathrm{d}s \tag{2}$$

The computation requires the minimization of this energy.

2.2. Approximation by a Cubic Spline

The contours used to model features are cubic B-splines (Trinder and Li, 1996). The advantages associated with the splines are that they are smooth piecewise polynomials which maintain continuity between neighboring sections or domains. Moreover, they limit the influence of errors in the points used to locate the curve of order n to $n+1$ domains of the curve. Knowledge of both a set of polynomial functions, which determine the shape, and a given set of control points or a control polygon, specified by a number of fixed locations along the curve, defines the B-spline. The B-spline function adopted here is a third-order function, while image details representing the feature determine the control polygon. The spline is subject to internal forces, referred to as internal energy, brought about by the inherent piecewise smoothness of the B-spline, while the external forces that relate to the image information guide the SA-based process described in the next section towards the feature.

Suppose the points delineated by the user are approximated by a cubic spline. That is

$$x(s) = \sum_{i=1}^{n} N_i(s)X_i \tag{3}$$

$$y(s) = \sum_{i=1}^{n} N_i(s)Y_i \tag{4}$$

where X_i and Y_i are the parameters of the cubic spline in x and y directions in the image respectively, and $N_i(s)$ is the normalized cubic B-spline between knots

s_i and s_{i+4}. For equally spaced simple knots, with a spacing h,

$$\begin{aligned} N_i(s) &= \frac{s^3}{6h^3} \quad 0 \leqslant s - s_i \leqslant h \\ &= \frac{1 + 3s + 3s^2 - 3s^3}{6h^3} \quad h \leqslant s - s_i \leqslant 2h \\ &= \frac{4 - 6s^2 + 3s^3}{6h^3} \quad 2h \leqslant s - s_i \leqslant 3h \\ &= \frac{1 - 3s + 3s^2 - s^3}{6h^3} \quad 3h \leqslant s - s_i \leqslant 4h \end{aligned} \tag{5}$$

In the actual implementation, the user-delineated points are located at discrete values so that the integration of the B-spline in equation (1) will be replaced with a summation. Letting

$$N = [N_1(s)N_2(s)\ldots N_n(s)] \tag{6}$$

$$N_s = [N_1'(s)N_2'(s)\ldots N_n'(s)] \tag{7}$$

$$N = [N_1''(s)N_2''(s)\ldots N_n''(s)] \tag{8}$$

$$X = [X_1X_2\ldots X_n]^T \tag{9}$$

$$Y = [Y_1Y_2\ldots Y_n]^T \tag{10}$$

$$E_p = [f(x,y)]^T[f(x,y)] \tag{11}$$

the total energy can be written in matrix form as

$$\begin{aligned} E = \gamma X^T N_s^T N_s X + \beta X^T N_{ss}^T N_{ss} X + \gamma Y^T N_s^T N_s Y \\ + \beta Y^T N_{ss}^T N_{ss} Y + [f(x,y)]^T[f(x,y)] + E_c \end{aligned}$$

To minimize this energy the conditions are

$$\frac{\partial E}{\partial X} = \frac{\partial E}{\partial Y} = 0$$

Hence

$$(\gamma N_s^T N_s + \beta N_{ss}^T N_{ss})X + N^T[f_x(x,y)]^T[f(x,y)] + \frac{\partial E_c}{\partial X} = 0$$

and

$$(\gamma N_s^T N_s + \beta N_{ss}^T N_{ss})Y + N^T[f_y(x,y)]^T[f(x,y)] + \frac{\partial E_c}{\partial Y} = 0$$

In a semi-automatic feature extraction scheme, the operator initially places a curve near the image structure of interest. By this initial curve, the approximations X_0 and Y_0 of X and Y can be calculated. The goal is therefore to find the corrections δX and δY such that $X = X_0 + \delta X$ and $Y = Y_0 + \delta Y$. Letting $B = \gamma N_s^T N_s + \beta N_{ss}^T N_{ss}$ and following appropriate substitutions, based on the assumption that the initial curve is located on the feature, the final set of equations becomes

$$\begin{aligned} (B + N^T[f_x(x_0,y_0)]^T[f_x(x_0,y_0)]N + \lambda I)\delta X \\ + N^T[f_x(x_0,y_0)]^T[f(x_0,y_0)] = 0 \end{aligned} \tag{12}$$

$T = T_{max}$
Initialize configuration ($\mathscr{C}$) with energy $\mathscr{E}_{\mathscr{C}}$ as in equation. (17).
while($T > T_{min}$)
begin
for i=1 to N_T do /* N_T is the number of iterations at temperature T*/
begin
Evolve configuration $\mathscr{C}'$ with energy $\mathscr{E}_{\mathscr{C}'}$ from $\mathscr{C}$ by selecting positions (i,j)
such that $v_i = 1$ and $v_j = 0$ randomly and exchange v_i and v_j.
If $(\mathscr{E}_{C'} - \mathscr{E}_C \leqslant 0)$ $\mathscr{C} \leftarrow \mathscr{C}'$
Else $\mathscr{C} \leftarrow \mathscr{C}'$ with probability $\exp(-\frac{\mathscr{E}_{\mathscr{C}'} - \mathscr{E}_{\mathscr{C}}}{T})$
endfor
$T \leftarrow T \times \alpha$
endwhile

Fig. 1. Basic steps in SA.

$$(B + N^T[f_y(x_0, y_0)]^T[f_y(x_0, y_0)]N + \lambda I)\delta Y + N^T[f_y(x_0, y_0)]^T[f(x_0, y_0)] = 0 \quad (13)$$

The contour dynamics in *snake* is based on solution of coupled differential equations. In contrast, the proposed SA-based method, described below, requires the calculation of simple energy differences.

3. Feature Extraction Using Simulated Annealing

3.1. Simulated Annealing

The SA algorithm starts from a random initial configuration at high temperature. It then proceeds by generating new candidate states and accepting/rejecting them according to a probability which is a function of the current temperature and energy difference. The temperature is gradually decreased towards a minimum value, while the system settles down to a stable low energy state. The basic steps of SA which are also followed in SAFE are shown in Fig. 1.

3.2. SAFE Procedure

3.2.1. Basic Principle

The feature of interest $\mathscr{F}$ is first roughly delineated by providing control points along the boundary. Subsequently B-splines are used to model the curve formed from the control points. The advantages of the spline are that they are smooth piecewise polynomial which maintain continuity between neighboring domains. Let $\mathscr{S}$ represent the contour which the SAFE algorithm tries to fit around $\mathscr{F}$. $\mathscr{S}$ is initialized to the B-spline. A window $\mathscr{W}$ is considered across $\mathscr{S}$ in such a way that it includes $\mathscr{F}$. One may note that even if the separation between $\mathscr{F}$ and $\mathscr{S}$ increases, it is still possible to capture $\mathscr{F}$ by suitably increasing the size of $\mathscr{W}$. Therefore it is not necessary for the operator to provide a very accurate

delineation of $\mathscr{F}$. Considering the state of $\mathscr{W}$ as the current configuration, the SA based feature extraction algorithm (SAFE) attempts to provide $\mathscr{F}'$, its estimate of $\mathscr{F}$, by minimizing the energy within $\mathscr{W}$.

3.2.2. Configuration Representation

Let $\mathscr{S}$ consist of n points (pixels) and be represented by

$$\mathscr{S} = \{x_{s_1}, x_{s_2}, \ldots, x_{s_n}\} \tag{14}$$

Let the window $\mathscr{W}$ contain w points and be represented by

$$\mathscr{W} = \{x_1, x_2, \ldots, x_w\} \tag{15}$$

Note that $\mathscr{S}$ lies within $\mathscr{W}$. The configuration considered by the SAFE is represented as

$$\mathscr{C} = \{v_1, v_2, \ldots, v_w / v_i \text{ is the value at } x_i \text{ and } x_i \in \mathscr{W}\} \tag{16}$$

$\mathscr{C}$ is initialized as follows:

$$\begin{aligned} v_i &= 1 \text{ if } x_i \in \mathscr{S} \\ &= 0 \text{ if } x_i \in (\mathscr{W} - \mathscr{S}) \end{aligned} \tag{17}$$

In other words, a configuration represents the state of a window at an instant and initially it represents the window containing only the B-spline. Note that $\sum_{i=1}^{w} v_i = n$.

3.2.3. Energy Computation

Each configuration in the search space is assigned an energy which is dependent on the global characteristic of its corresponding window. The energy $\mathscr{E}_{\mathscr{C}}$ of configuration $\mathscr{C}$ is defined as follows:

$$\mathscr{E}_C = \sum_{i=1}^{w} v_i \times e_i \tag{18}$$

where e_i denotes the energy at point x_i in $\mathscr{W}$.

The SAFE algorithm redistributes the n points in $\mathscr{S}$ within $\mathscr{W}$ such that $\mathscr{E}_{\mathscr{C}}$ is gradually minimized while keeping $\sum_{i=1}^{w} v_i = n$; thereby providing closer and closer delineation of $\mathscr{F}$.

3.2.4. Location of Window

SAFE algorithm has been implemented using two types of windows. These are the following.

Rectangular window. Let (x_{min}, y_{min}) and (x_{max}, y_{max}) be the minimum and maximum in the two dimensions respectively of the B-spline drawn from the initial user delineation. A rectangular window is generated with the following coordinates: $(x_{min}, y_{min}), (x_{max}, y_{min}), (x_{min}, y_{max})$ and (x_{max}, y_{max}). Although this implementation (see Fig. 4) is intuitively very simple, it has the associated disadvantage that the size of the window may be very large depending on the length and shape of the B-spline. As the area of the window increases, the search space of SA also increases, thereby requiring more iterations to locate

the feature appropriately. Moreover, with the increase in the size of the window, the likelihood of including more than one feature in it increases. As a result, the feature of interest may not be properly delineated. To overcome these problems we also implement SAFE using piecewise linear window in the following way.

Piecewise linear window. Let $(x_1, y_1), (x_2, y_2), \ldots (x_n, y_n)$ be the n user-delineated points. For both the user-delineated points (x_1, y_1) and (x_n, y_n), generate two points (x'_i, y'_i) and (x''_i, y''_i) $i = 1$ and n at a distance d, specified by the user, from the corresponding point and perpendicular to the line joining the points (x_1, y_1) & (x_2, y_2) and (x_{n-1}, y_{n-1}) & (x_n, y_n) respectively (see Fig. 5). For each of the remaining points (x_i, y_i), $i = 2, \ldots, (n-1)$ generate two points (x'_i, y'_i) and (x''_i, y''_i) at a distance d from (x_i, y_i) and along the bisector of the angle between the lines joining (x_{i-1}, y_{i-1}) to (x_i, y_i) and (x_i, y_i) to (x_{i+1}, y_{i+1}). Using these $2n$ points (x'_i, y'_i) and (x''_i, y''_i) $i = 1, \ldots, n$, $(n-1)$ sub-windows are generated such that the ith sub-window $(i = 1, \ldots, n-1)$ has the following coordinates: (x'_i, y'_i), (x''_i, y''_i), $(x'_{(i+1)}, y'_{(i+1)})$ and $(x''_{(i+1)}, y''_{(i+1)})$. Note that d controls the size of the piecewise linear sub-windows. Thus by suitably tuning the value of d, the size of the sub-windows may be varied.

3.2.5. *Move for Generating New State ($\mathscr{S}'$) from Current State ($\mathscr{S}$)*

From $\mathscr{S}$, a pixel p_i is picked up randomly for mutation. This pixel is replaced by another pixel p_j from $\mathscr{W}$ which is not already in $\mathscr{S}$. The modified $\mathscr{S}(\mathscr{C})$ is then referred to as $\mathscr{S}'(\mathscr{C}')$. Its energy is computed as $\mathscr{E}_{e'} = \mathscr{E}_e - e_{p_i} + e_{p_j}$. Thus for computing the energy of the new state, two simple arithmetic operations are sufficient instead of solving coupled differential equations as in traditional *snake*. This is the key factor that makes the system very simple and fast.

3.2.6. *SAFE Algorithm*

The different steps used in the SAFE algorithm are as follows:

1. pre-processing (e.g., image stretching, contrast enhancement, noise reduction etc.);
2. edge detection (by Canny operator, morphological tools etc.);
3. derivation of a chamfer image (Trinder and Li, 1996) in which the pixel values relate to their closeness to any surrounding edge. For example, if an edge pixel has the gray value strength G_e, then for a non-edge pixel located x pixels away from it, its gray value $G_{non-edge}$ (in the chamfer image) should become $G_e - x$. Since there are many pixels in an image, the recorded value $G_{non-edge}$ is the maximum value derived from all edge pixels;
4. acquisition of control points (initial curve) from the user that roughly delineates the feature;
5. smoothing the initial curve using B-spline;
6. construction of a window (rectangle/piecewise linear) across the spline;
7. energy minimization in the window using SA;
8. fitting a curve through the pixels which minimizes the total energy in the window. This curve provides the resultant delineation of the feature.

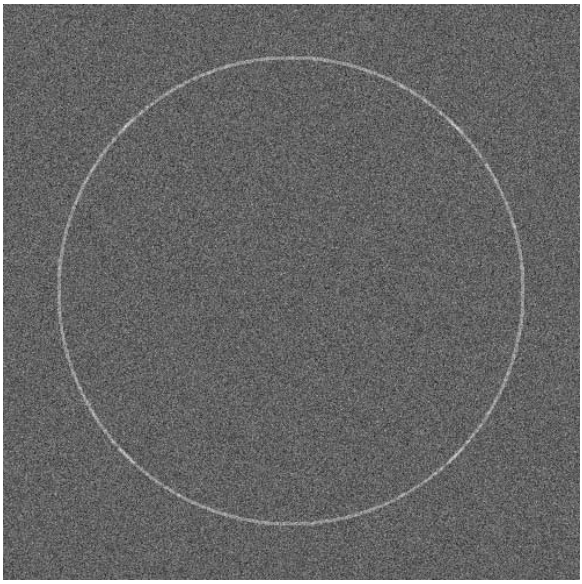

Fig. 2. Noisy image of a circle.

4. Implementation and Results

4.1. Temperature scheduling of SA

In this article SAFE algorithm has been implemented with the following parameters: temperature scheduling factor (α) = 0.8, $d = 15$, $T_{max} = 100$, $T_{min} = 0.00001$ and number of iterations at each temperature = 1000. The traditional *snake* algorithm was executed for both 50 and 100 iterations. Since the performance was found to be the same for both values, we are demonstrating the results obtained after 50 iterations only.

4.2. Results

In order to demonstrate the performance of SAFE we consider a synthetic image of a circle shown in Fig. 2. Note that this image is similar to an image considered in Ballerini (1999). As in Ballerini (1999), the intensity image is generated by setting a pixel value to 255 if it belongs to a boundary, and 100 otherwise. We smooth the boundary by convolving the image with a 3×3 window. A zero mean, white Gaussian noise is added to the image. Several noise levels were considered, the result reported here corresponding to standard deviation = 40. As can be found from Fig. 3, SAFE is able to successfully detect the boundary.

Figures 4–9 demonstrate the performance of traditional *snake* and SAFE respectively on both an aerial and a satellite image. The task is to extract the boundary of the road and the river on the aerial and satellite image respectively. These boundaries are characterized by step edges and therefore Canny edge detector has been used in the pre-processing phase. The thin white curves in the figures represent the B-spline drawn from the user-provided control points while

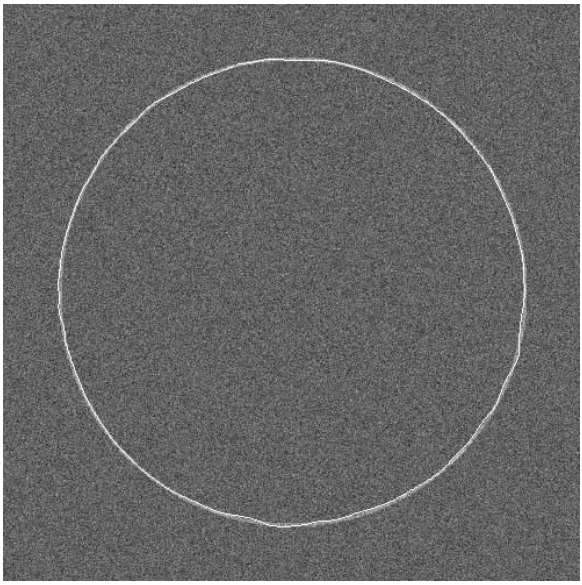

Fig. 3. Circle as detected by SAFE in Fig. 2.

Table 1. Time taken by traditional *snake* and SAFE in milliseconds.

Image	*snake*	SAFE
Aerial	216.658	266.656
Satellite	246.356	269.990

the black ones represent the final solution. Figures 4 and 5 demonstrate the use of rectangular and piecewise linear windows respectively for extracting the feature of interest by the SAFE algorithm. Figures 6 and 7 demonstrate the comparative performance of traditional *snake* and SAFE respectively, starting from the same initial configuration for the aerial image. As can be seen from the figures, while SAFE is able to provide a good estimate of the feature of interest (Fig. 7), the traditional implementation of *snake* fails to do so (Fig. 6). In this connection, it has been found in Trinder and Li (1996) that the pull-in range of traditional *snake* is normally about 5 pixels i.e., if the user delineated points are more than 5 pixels away from the feature of interest, then the *snake* will not be able to extract the feature. Similarly Figs 8 and 9 demonstrate the effectiveness of traditional *snake* and SAFE respectively, starting from the same initial configuration for the satellite image. Table 1 shows the CPU time taken by the two methods (*snake* and SAFE) for obtaining the results shown in Figs 6 and 7 for the aerial image and Figs 8 and 9 for the satellite image, when they are run on a DEC-Alpha machine. The values shown are averaged over 100 runs of the algorithms. As seen from the table, the time taken by the two methods is comparable although the performance of SAFE is found, from the figures, to be better.

The performance variation, with regard to energy minimization, of the SAFE algorithm with varying α and window sizes have been studied. As the value of α increases ($0 < \alpha < 1$), the cooling process becomes more gradual. As a result, the

Fig. 4. Feature extraction using SAFE with rectangular window for aerial image.

Fig. 5. Feature extraction using SAFE with piecewise linear window for aerial image.

Fig. 6. Feature extraction using *snake* for aerial image.

Fig. 7. Feature extraction using SAFE with piecewise linear window for aerial image.

Fig. 8. Feature extraction using *snake* for satellite image.

Fig. 9. Feature extraction using SAFE with piecewise linear window for satellite image.

SA-based algorithm stabilizes to lower energy values. This is evident from Fig. 10, which demonstrates the variation of energy values with the number of generations for $\alpha = 0.4, 0.5, 0.6, 0.7$ and 0.8. Regarding window size, it may be noted that as it increases, the corresponding search space also increases. Therefore for fixed α, T_{max}, T_{min} and number of iterations in each temperature, the performance of the algorithm improves (i.e., the final energy value is less) as the window is made smaller. This has been demonstrated in Table 2, where it is found that starting from the same initial configuration, i.e., same initial energy value, the SA-based algorithm is able to attain a lower final energy value for a smaller window. Note that the performance of SAFE may improve if it is allowed to run longer by suitably altering the temperature schedule parameters.

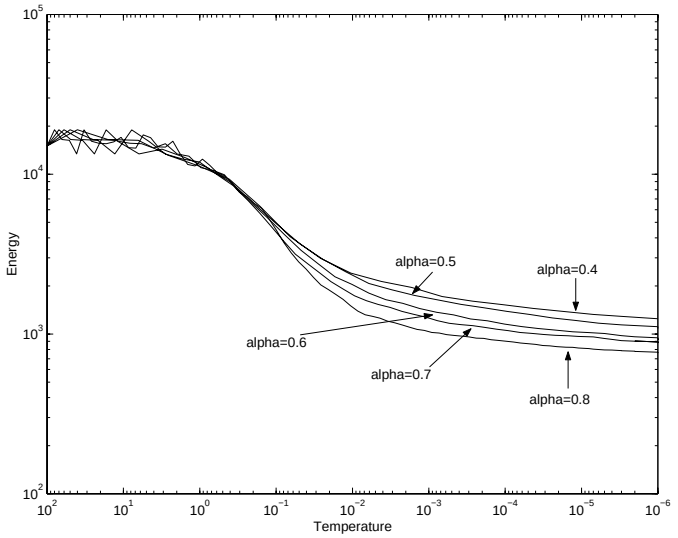

Fig. 10. Variation of energy values with temperature for different values of α ('alpha' in the figure).

Table 2. Performance variation of SAFE with window size. Energy of initial configuration is 1579.289, $T_{max} = 100$, $T_{min} = 0.00001$ and $\alpha = 0.8$.

Window size	Final energy value
40	759.128
35	749.614
30	743.222
25	719.196
20	696.812
15	680.495

5. Discussion and Conclusions

An efficient semi-automatic feature extraction technique called SAFE, which exploits the searching capability of simulated annealing in the energy minimization phase of the active contour model, is developed in this article. The significant superiority of SAFE over the traditional *snake* is demonstrated on an aerial and a satellite image. It is found that the execution time of the two methods are comparable. Note that *snake* may get trapped at a local energy minimum depending on the initial user delineation and repeated reinitializations are required each time this happens. Moreover, it is found to provide a good solution only when the initial user delineation is quite close to the feature of interest. SAFE does not suffer from either of these limitations, the only requirement being that the window size be large enough so as to include the feature of interest. This results in a lower and simpler level of user interaction for SAFE (which is an important step towards building a fully automatic system) as compared to the *snake* model.

In Grzeszczuk and Levin (1997), simulated annealing was also used for contour detection. However, they have defined a non-parametric energy function which is derived from statistical properties of previously segmented images,

thereby incorporating prior experience. Note that such *a priori* knowledge may not be available in all circumstances. Moreover, the moves used by them to generate new states are computationally complex. Storvik (1994) also applied simulated annealing to optimize a boundary modeled by a polygon whose vertices coincide with pixel corners. Since the method had little control over the move size distribution, a slow logarithmic annealing schedule was used which resulted in extremely slow performance. In contrast to the above two attempts, the technique developed in this article is computationally very simple. It requires no *a priori* knowledge about the image. The move generator is also very simple. Since we used geometric scheduling, instead of logarithmic scheduling, the time required by SAFE to converge to the solution is quite fast.

Acknowledgements. The authors acknowledge the Australian Research Council for providing the grant to carry out this research. They are also grateful to the anonymous reviewers for their valuable suggestions for improving the quality of the manuscript.

References

Ballerini L (1999) Genetic snakes for medical images segmentation, In Poli R (ed). Evolutionary image analysis, signal processing and telecommunications. Springer, Berlin, pp 59–73

Barzohar M, Copper D (1996) Automatic finding of main roads in aerial images by using geometric–stochastic models and estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence 18:707–721

Canny J (1986) A computational approach to edge detection. IEEE Transations on Pattern Analysis and Machine Intelligence 8:679–698

Grzeszczuk RP, Levin DN (1997) Brownian strings: segmenting images with stochastically deformable contours. IEEE Transactions on Pattern Analysis and Machine Intelligence 19:1100–1114

Kass M, Witkin A, Terzopoulos D (1988) Snakes: active contour models. International Journal of Computer Vision 1:321–331

Kirkpatrik S, Gelatt C, Vecchi M (1983) Optimization by simulated annealing. Science 220:671–680

Merlet N, Zerubia J (1996) New prospects in line detection by dynamic programming. IEEE Transactions on Pattern Analysis and Machine Intelligence 18:426–431

Storvik G (1994) A bayesian approach to dynamic contours through stochastic sampling and simulated annealing. IEEE Transactions on Pattern Analysis and Machine Intelligence 16:976–986

Trinder JC, Li H (1996) Extraction of man-made features by 3-d active contour models. International Archives of Photogrammetry and Remote Sensing 874–879

Correspondence and offprint requests to: S. Bandyopadhyay, Machine Intelligence Unit, Indian Statistical Unit, 203 BT Road, Calcutta 700 35, India. Email: sanghami@isical.ac.in