AN ADAPTIVE SNAKE ALGORITHM FOR CONTOUR DETECTION

Qin Zhongyuan, Mou Xuanqin, Wang Ping, Cai Yuanlong

Institute of Image Processing, Xi'an Jiaotong Univ., Xi'an, Shaanxi, 710049 China

Email: zyqinljl@hotmail.com

Abstract: An adaptive Snake algorithm is presented in this paper. To every point in the initial position, the energy of the point in its neighbors is calculated by Greedy algorithm. If the target contour is not included in its neighbors, we can increase the radius of its neighbors and calculate the energy of all the points again until target contour is included. The target contour can be got by iterating once. In addition, the convergent radius is increased. It can also be applied to objects of high convexity. Comparative experiments indicate the validity of this method.

Key Words: Active contour model Snake adaptive

1. Introduction

Contour based technique has been proven to be an effective approach in object recognition. Most of these recognition schemes assume that a closed object boundary is obtained and the recognition algorithms are developed based on the detected closed boundary. Accordingly, closed contour detection is a crucial and first step in object recognition [1].

Active Contour Models (also called snakes), which optimize/minimize an energy function, has become quite popular for boundary detection in the past few years. The energy function is generally obtained from internal and external sources with different weighting coefficients [2]. Snakes have also been used for edge and curve detection, segmentation, shape modeling, and visual tracking.

There are two key difficulties with Active Contour Models. First, the initial contour must, in general, be close to the true boundary or else it will likely converge to the wrong result. The second problem is that active contours have difficulties progressing into *boundary concavities and convexities.*

After the original snake model was proposed, greedy algorithms that searched local neighborhoods instead of performing global optimization were developed to reduce the computational complexity of the snake models[3]. However, traditional snake models are highly sensitive to initial contour conditions and are unable to approach the concave parts of an object boundary. The gradient vector flow (GVF) was proposed to overcome these deficiencies [4, 5]. The formation of the GVF field is driven by image gradients that are diffused into the whole picture after iteration.

However snake is still confused to high convex boundary. We present a novel adaptive algorithm based on Greedy algorithm to solve this problem. It changes the size of neighborhoods adaptively in iteration. Satisfactory results are obtained. And it is easy to be implemented and fast in computation.

2. Snake Formulation

A snake is an elastic curve ($v(s)$) that evolves from its initial shape and position as a result of combined action of external and internal forces. The internal forces model the elasticity of the curve, whereas external forces push the snake towards features of the image. The conventional energy function in the snake is defined as follows:

$$E^*_{snake}(v) = \int_0^1 E_{snake}(v(s))ds = \int_0^1 [E_{\text{int}}(v(s)) + E_{ext}(v(s))]ds \quad (1)$$

where E_{int} and E_{ext} are the internal spline energy and the external image energy respectively, which can be defined as

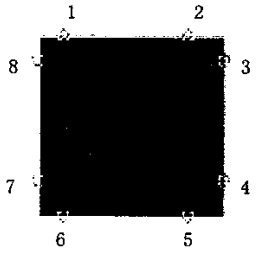
$$E_{\text{int}}(v(s)) = (\alpha(s)|v_s(s)|^2 + \beta(s)|v_{ss}(s)|^2)/2 \quad (2)$$

$$E_{ext} = w_{line}E_{line} + w_{edge}E_{edge} + w_{term}E_{term} \quad (3)$$

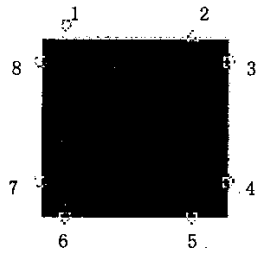
where $E_{line} = I(x,y)$, $E_{edge} = -|\nabla I(x,y)|^2$, E_{term} is the curvature of the contour in the image smoothed by the Gause function.

Equation (2) contains a first-order term which will have large values when there is a gap in the contour, and a second-order continuity term which will be larger where the contour is bending rapidly. The value of α and β at a point determines the extent to which the contour is allowed to stretch or bend at that point. The relative sizes of α and β can be chosen to control the influence of the corresponding constraints. If α is 0 at a point, a discontinuity can occur at that point, while if β is 0, a corner can develop, because large value of these terms would not be included in the total would not be included in the total.

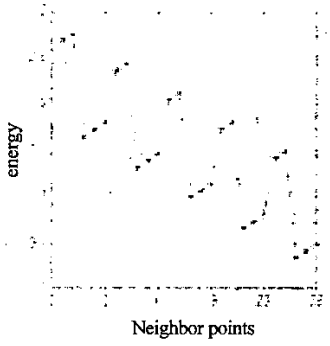
0-7803-7488-6/02/$17.00 © 2002 IEEE.



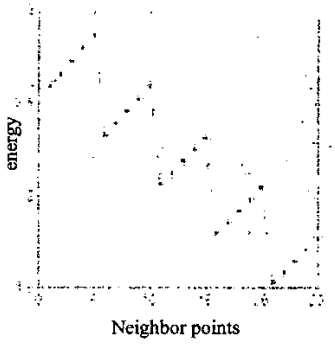
(a) Initial contour 1



(b) Initial contour 2



(c) The energy of the neighborhood of point 1 in contour 1



(d) The energy of the neighborhood of point 1 in contour 2

1	6	11	16	21
2	7	12	17	22
3	8	(13)	18	23
4	9	14	19	24
5	10	15	20	25

(e) The distribution of neighborhoods of point 1. The point with circle is the snake point. The last line is the boundary.

Figure1 The change of energy of some point's neighborhood in Greedy algorithm.

Till now, the size of neighborhoods in snakes is fixed, in most cases it is 3*3 or 5*5. When the initial snake point is far from the object contour, E_{ext} has less change in the neighborhoods. It will change its position mostly on the internal forces, which makes it difficult to converge to object contour. From this point we present an adaptive algorithm based on Greedy algorithm, the radius of the neighbor changes adaptively during the iteration and satisfactory results are obtained.

3. Adaptive Snake Algorithm

The Energy function in Greedy algorithm is defined as:

$$E = \int(\alpha(s)E_{cont} + \beta(s)E_{curv} + \gamma(s)E_{image})ds. \quad (4)$$

where $E_{cont} = \bar{d} - |v_i - v_{i-1}|$, $E_{curv} = |v_{i-1} - 2v_i + v_{i+1}|^2$, $\bar{d}$ is the average distance between the snake points, E_{image} is the gradient magnitude of the image.

The energy of every point in the neighborhoods is computed from (4) in Greedy algorithm during each iteration. The point with minimal energy is chosen as the new position in the next iteration. After deep research of the energy of every point in the neighborhoods, we find that the energy is lower (at least, there is one point whose energy is lower than 1.0) if the boundary is included in the neighborhood, otherwise they are all higher than 1. Just as shown in figure1.

From the above discussion we present an adaptive snake algorithm which adjusts the radius of

neighborhoods. Its basic principle is as follows: To every snake point, we compute the energy of its neighbor points according to Greedy algorithm. If the boundary is included in the neighborhoods, there must be some points whose energy is quite low. The point with the lowest energy is the position in the next iteration. Otherwise, the radius is increased and the energy is computed again until the boundary is included. After all points are processed in this way, we can get the boundary needed.

In the implementation of the algorithm we make the following consideration:

(1) The choice of neighbor shape

The traditional neighbor shape is square, but more than one boundary point may be included which reduces the accuracy of the result. Therefore, we choose round shape, for it will intersect the boundary in one point. The detail is: First choose a square neighbor, then compute the distance of every point in the neighborhood to the center, if it is less than some threshold (the radius), we can regard it as inside the round neighbor.

(2) Judgment of whether a boundary point is included in the neighbor

This is the key point in our algorithm. It is observed that when a snake point is inside the neighbor, the minimal energy will be lower than 0.5, otherwise the minimal energy will be higher than 1.0. So 1.0 can be chosen as the threshold of whether a boundary point is included in the neighbor. A simple explanation is as follows: First the E_{image} of the boundary is nearly 0, and other points' is nearly 1.0. Second E_{cont} and E_{curv} are both normalized to [0,1], so $E_{cont} + E_{curv}$ will be in [0,2]. When the boundary is contained in the neighbor and the size of neighbor is large enough, there must exist some points in the neighbor whose $E_{cont} + E_{curv}$ <1/2. So this point's energy will be lower than 1.0 and the energy in other position will be higher than 1.0.

The pseudo-code for the adaptive algorithm is as follows:

Initialize the snake point set $(1,2,\cdots N)$, the neighbor size is r, the neighbor space is $(1,2,\cdots M), M = r^2$, $\alpha = 1$, $\beta = 1$, $\gamma = 1.2$.

```
for (I=1,I<=N;I++)
{
    Set initial neighbor space of point I;
    E min =Max;
    while (E min >1)
    {
        for (j=1,j<=M,j++)
        {
          Compute Econt, Ecurv and Egrad;
          E(j)=α Econt+ β Ecurv+ γ Egrad;
        }
        for (j=1,j<=M,j++)
        {
            if (E(j) < E min)
            {
                E min =E(j);
            }
        }
        Increase the neighbor radius;
        Reset the neighbor space;
    }
}
```

Only one iteration is needed to get the object boundary by the above algorithm. The *capture range* of the external force fields is increased and high convexity can be processed efficiently.

4. Experimental Results and Analysis

Experiments are carried out to the three algorithms using Matlab 6.1. The PC used is Pentium 4 1.5G HZ. Object of high convexity is selected as the model in the three algorithms and results are shown in Figure 2. The time used is described in table 1.

From the above experiment we can see that to high convex object neither Greedy nor GVF can get satisfactory results, while adaptive algorithm shows rather satisfactory result.

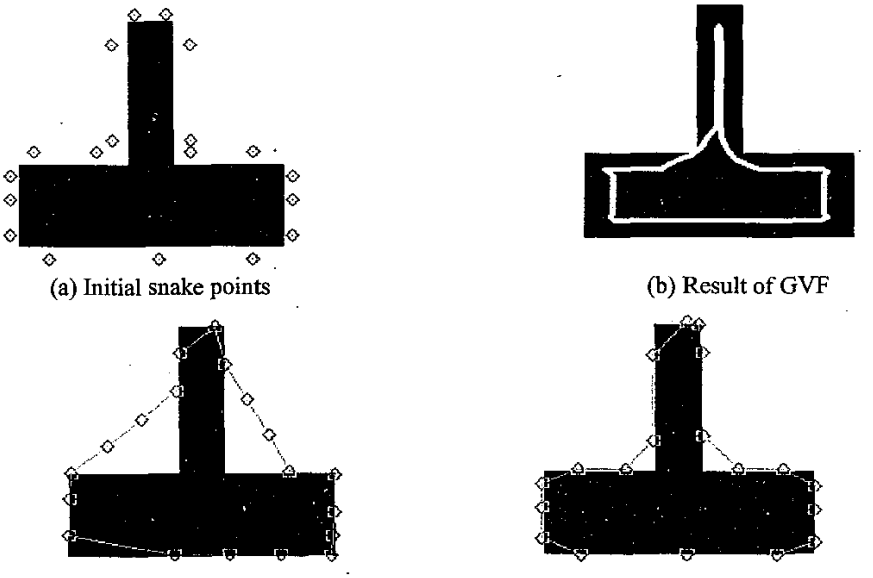
Experiments to round corner rectangle and inner boundary of a cirque are shown in figure 3.

5. Conclusion

Based on the analysis of GVF and Greedy, an adaptive algorithm for active contours has been presented to cope with objects of high convexity. By computing the neighbor's energy and increasing the radius, we thus enlarge the capture range and improve the robustness of Snakes. Experiments have shown the efficiency and accuracy of this method. The proposed algorithm can be easily adopted to existing recognition scheme to widen the practicability of existing computer vision and recognition systems.

References

1. P.H. Li and T.W. Zhang, "Review on Active Contour Model (Snake Model)", *Journal of Software*, Vol. 11(6), pp: 751~757,2000.
2. M. Kass, W. Within and D. Tetzopoulos, "Snake:active contour model" *Int J. Comput Vision*, Vol. 1 (4). pp: 321~33, 1987.
3. D.J. Williams and M. Shah, "A fast algorithm for active contours and curvature estimation", *Computer Vision and Image Understanding*, Vol. 55(1). pp: 14~26, 1992.
4. C. Xu and J.L. Prince, "Gradient Vector Flow Deformable Models", *Handbook of Medical Imaging*, edited by Isaac Bankman, Academic Press, September, 2000.
5. C. Xu and J.L. Prince, "Snakes, shapes and gradient vector flow", *IEEE Transactions on Image Processing*, Vol.7(3), pp: 359~369. 1998.



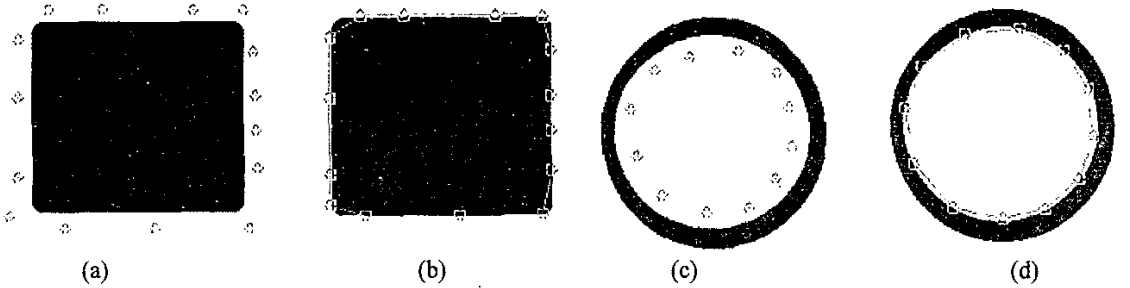
(a) Initial snake points (b) Result of GVF

(c) Result of Greedy (d) Result of adaptive algorithm

Figure 2 Results of the three algorithms to high convex object

Table 1: Time used for the three algorithms (second)

algorithm	GVF algorithm	Greedy algorithm	Adaptive algorithm
time	37.57	6.57	2.45



(a) (b) (c) (d)

Figure 3 Experiments to round corner rectangle and inner boundary of a cirque using the adaptive algorithm
(a) initial snake points when object is a round corner rectangle (b) the result using adaptive algorithm
(c) initial snake points when object is a inner boundary of a cirque (d) the result using adaptive algorithm