International Journal of Signal Processing, Image Processing and Pattern Recognition
Vol.9, No.12, (2016), pp.19-28
http://dx.doi.org/10.14257/ijsip.2016.9.12.03

Segment of Multiple Objects Based on Parameter Active Contour Model

Liu Hongshen,Wang Nan and Zhang Xuefeng

Anhui University of Technology, Hudong road 59, Maanshan, Anhui, China
857025058@qq.com

Abstract

The subject of this paper is the segmentation of multiple objects from images based on 6the parameter active contour model (PACM). After analyzing application of the parameter active model to segment multiple objects, the evolution strategies and disadvantages of existing methods are presented. This paper proposes that the key points are two parts in detecting multiple objects with the PACM in the shrinking strategy. One key point includes the split time where contours appear as self-crosses, and the split algorithm of contours. The other key point is to maintain the uniform distribution of sampling points on contours in order to match the shapes of objects in segmenting. A new algorithm for detecting self-crosses is presented, and the results show that the new algorithm is faster than the other algorithm. The problem where vertexes on contours are sampled to match the shapes of objects in segmenting is studied, and its solution is presented.

Keywords: *parameter active contour model; segment of multiple objects; the shape of contour; self-cross of contours*

1. Introduction

The active contour model (ACM) has been applied in segmenting objects in images and tracking objects in videos since being presented [1]. One reason is that image segments use not only image information, but also user consciousness with ACM. Therefore, ACM research has been popular [4-5]. ACM technology has developed rapidly, and the geometric active contour model (GACM) developed after the parameter active contour model (PACM) has been presented [5]. PACM technology has also been developed. Cohen [2] applied finite element analysis to PACM and proposed a balloon model that evolves in the inflation strategy. Xu [3] improved the external force in the model and introduced the gradient vector flow (GVF) to promote the convergence of sunken areas in objects. However, these studies mostly consider the case where there is a single object in the images, although generally there are multiple objects in an image. ACM application for segmenting multiple image objects should be researched. In the segment of multiple of objects with PACM, there are some problems that need to be solved. One is determining how to best ensure that the contour shape remain the same as the object shape; that is, the key with which the objects can be segmented completely. Another is ascertaining how the split time is determined at which initial contour split into multiple contours. Every object corresponds to one contour after a series of splits. The subject of this paper is the segmentation of multiple objects in images with PACM. Our goal is for all objects to be segmented from the image with PACM in the normal evolution strategy under the condition where there is one initial contour that contains all objects in the image studied at beginning.

ISSN: 2005-4254 IJSIP

2. Current Studies on Segmentation of Multiple Objects with Parameter Active Contour Model

Segmenting the objects in an image with ACM demands that each object be encircled by one contour. Such demand is easily satisfied for the segment of a single object. However, for multiple objects, this is a complicated problem because there is one initial contour at the start of the evolution. When segmenting multiple objects, there are two evolution strategies in ACM: an expansion strategy with GVC [2], and a shrink strategy. For the expansion strategy, some contours, known as the seed of evolution, must be determined at the start. In [7], the initial contours are determined by two phases. In the first phase, a grid is set in the image's global area, and the grid points are evenly distributed in the image area. Then the grid points move toward the object edges under the evolution and form some point clusters. The points in such cluster are connected with lines and a polygon is formed. These polygons are used as the initial contours for the next phase.

In the second phase, segmentation of multiple objects in the image is executed by GVF. In [8-9], a seed point corresponds to one object, and this is the initial contour at the start of evolution. Then these seed points evolve into contours that contain objects under the expansion strategy. Reference [8] randomly sows the seed points in the image's global area, and each seed point evolves to form a contour with GVC and the expansion strategy. When the contour stops evolving, a pattern image is formed. Subsequently, each pattern image is analyzed by the principal component analysis (PCA) and the reconstruction error is calculated. The contour might contain an object if its reconstruction error is less than a given threshold. Reference [9] does not sow randomly in the image's global area, and determines the area in which the objects probably exist at first. Second, the seed points are sowed in the area.

The limitations of the aforementioned studies are that it is difficult to reasonably sow the seed points in the image area in order to omit no objects. For the ACM shrink strategy, the initial contour is produced by users. However, the problem for how contours correspond to objects is not solved easily because there is only one initial contour and more than one object. Contour splitting is the only method for solving this problem. In addition, the problem for how contour shapes remain as the object shapes is to be researched. This paper's subject is the segmentation of multiple objects with PACM under the shrink strategy of evolution.

3. Time and Treatment of Contour Split in Parameter Active Contour Model

3.1. Contour Self-cross and Application in Splitting Contours

There is only one initial contour in PACM under the shrink strategy of evolution. Segmentation of multiple objects with PACM demands that one object be enclosed by a single contour. Along with PACM evolution, the initial contour encloses the objects and is split into multiple sub-contours in order to satisfy the demand of multiple objects. Let C_0 be a contour composed of a series of edges so that C_0 is the set of edges:

$$\boldsymbol{C}_0 = \{e_{00}, e_{01}, \cdots, e_{0n}\} \tag{1}$$

C_0 is then split such that all C_0 edges are divided into two groups. For example, C_0 is split into sub-contours C_1 and C_2. The two sets of edges are described as follows:

$$\boldsymbol{C}_1 = \{e_{10}, e_{11}, \cdots, e_{1m}\}$$

$$\boldsymbol{C}_2 = \{e_{20}, e_{21}, \cdots, e_{2k}\} \tag{2}$$

where e_{1x} and e_{2x} are the edges of C_0. The manner in which these C_0 edges are divided into two groups must be studied. A set of edges in a contour is an ordered sequence of edges. Every edge has three properties: two endpoints and a serial number. To group the edges, we must seek two boundary points that are the serial numbers of two edges. Those edges whose serial number is between the serial numbers of two boundary edges are divided into one group, and all others are placed in the other group. Therefore, splitting a contour becomes finding the two boundary edges. The results of our research show that contour self-crosses in PACM are when contours should be split, and this always happens. The results also show that some self-crosses are related to multiple objects. Contour self-crosses in PACM refer to two contour edges regarded as the two boundary edges.

In PACM with the shrink strategy of evolution, the time for splitting contours occurs when the contours arise contour self-crosses. In order to split contours correctly, contour self-crosses must be detected. The reason for PACM being able to segment image objects and track video is that the image's global area is scanned by the contour edges when the contour evolves and the characteristic data in the image are inspected. When a contour edge moves over the area of an image where there is some characteristic data, the contour edge stops shifting. The fact that there is an intersection between two nonadjacent edges of a contour indicates that there are no characteristic data in the area where these two nonadjacent edges are scanned. All image objects are separated by some areas where there is no characteristic data, and the initial contour edges produce self-crosses among the image objects. All self-crosses are produced because of areas where there are no characteristic data. These self-crosses provide the time for splitting the contour.

3.2. Self-Cross Detection

The appearance of self-crosses is dynamic and unpredictable in the PACM evolution with the shrink strategy of evolution. Checking contour self-crosses involves two steps. The first is to determine whether the self-crosses exist in the contours in the procedure for evolution. If a self-cross exists in the contours, the second step calculates the intersection position in the self-cross in order to split the contours [10]. The intersection position information in a self-cross is always the coordinates of the cross point. However, splitting contours requires the serial number of the cross edges in the contours. Although such serial numbers can be obtained by calculating the coordinates of the cross point, such calculation is complex. Based on this analysis, detecting contour self-crosses is only required to determine whether the intersection of two nonadjacent edges exists, but it is not required to calculate the intersection coordinates.

Among all methods that determine whether one line intersects another, the simplest determines the projection overlap of those lines along the X or Y-axis. However, this method, called the projection method is this paper, is not efficient. A more efficient method is introduced here and illustrated in Figure 1.

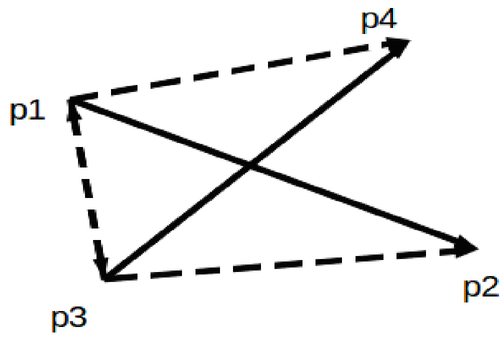


Figure 1. Intersection Determination for Two Lines on Same Surface

The intersection conditions of $\overrightarrow{P1P2}$ and $\overrightarrow{P3P4}$ include:

(1) $\overrightarrow{P3P1}$ and $\overrightarrow{P3P2}$ are located on the two sides of $\overrightarrow{P3P4}$; the direction of $\overrightarrow{P3P4} \times \overrightarrow{P3P2}$ is opposite to the direction of $\overrightarrow{P3P4} \times \overrightarrow{P3P1}$; that is, the sign of the cross product of $\overrightarrow{P3P4} \times \overrightarrow{P3P2}$ and $\overrightarrow{P3P4} \times \overrightarrow{P3P1}$ is different.

(2) $\overrightarrow{P1P3}$ and $\overrightarrow{P1P4}$ are located on the two sides of $\overrightarrow{P1P2}$; the direction of $\overrightarrow{P1P2} \times \overrightarrow{P1P4}$ is opposite to the direction of $\overrightarrow{P1P2} \times \overrightarrow{P1P3}$; that is, the sign of the cross product of $\overrightarrow{P1P2} \times \overrightarrow{P1P4}$ and $\overrightarrow{P1P2} \times \overrightarrow{P1P3}$ is different.

Two vectors are collinear when their cross product is zero. Let (Xi,Yi) be the coordinates of Pi. Let

$$F1 = \overrightarrow{P3P4} \times \overrightarrow{P3P2} = (X4-X3)\cdot(Y2-Y3)-(Y4-Y3)\cdot(X2-X3)$$

$$F2 = \overrightarrow{P3P4} \times \overrightarrow{P3P1} = (X4-X3)\cdot(Y1-Y3)-(Y4-Y3)\cdot(X1-X3)$$

$$F3 = \overrightarrow{P1P2} \times \overrightarrow{P1P4} = (X2-X1)\cdot(Y4-Y1)-(Y2-Y1)\cdot(X4-X1)$$

$$F4 = \overrightarrow{P1P2} \times \overrightarrow{P1P3} = (X2-X1)\cdot(Y3-Y1)-(Y2-Y1)\cdot(X3-X1) \quad (3)$$

The intersecting conditions of $\overrightarrow{P1P2}$ and $\overrightarrow{P3P4}$ are described as follows:

$$F1\cdot F2 < 0 \wedge F3\cdot F4 \le 0 \text{ or } F1\cdot F2 \le 0 \wedge F3\cdot F4 < 0 \quad (4)$$

Checking edge intersection is performed after every turn of the contour evolution. Table 1, indicates the time when three algorithms run. As can seen from this table, our algorithm is the fastest. The data unit in Table 1, is the time interval of the counter from the authors' PC. The time in Table 1, is that for checking the intersection in the contour; it does not contain the treatment time for self-crosses.

Table 1. Running Time for Three Algorithms

Contour No.	Contour Length	Algorithm in [10]	Proposed algorithm	Projection Algorithm
1	86	927	849	937
2	64	797	485	521
3	17	668	86	92

3.3. Splitting Contours

Once the edge intersection is found, the contours are split. There are two types of self-crosses when PACM evolves with the shrink strategy of evolution. One is caused by multiple objects, such as self-crosses A and B shown in Figure 2. The other is caused by contour redundancy, such as self-crosses C and D. The negative effect of self-crosses is that they alter the PACM evolution strategy from shrink to expansion. Therefore, PACM self-crosses must be treated on time.

Along with PACM evolution, self-crosses appear randomly. The extreme situation where several self-crosses appear on one contour simultaneously is possible. Therefore, a study of splitting contours must illustrate such situation as an example. Figure 2, shows that there are four self-crosses on one contour at one moment.

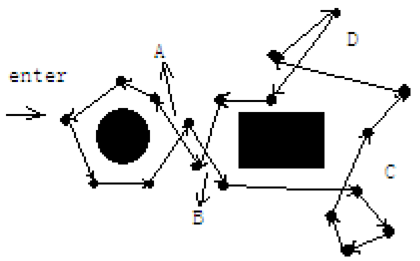


Figure 2. Case of Self-crosses on Contour

In Figure 2, self-crosses A and B are caused by the objects. Self-crosses C and D are caused by contour redundancy. The above detection algorithm for self-crosses outputs a pair of integers that represent the order number of the cross edges for one self-cross. There are four pairs of integers in Figure 2. The question is, in what order are they treated? Assume that numbering all contour edges starts and ends at the enter point, and the four pairs of cross edge numbers are (iA,jA), (iB,jB), (iC,jC), and (iD,jD). The following relationship exists between these edge numbers:

$$iA < iB < iC < jC < iD < jD < jB < jA \tag{5}$$

There are two relationships between two self-crosses. One contains the relationship, and the other is the parallel relationship. We define self-cross B as being contained in self-cross A if iA < iB and jA > jB. Similarly, we define self-crosses A and B as being parallel to each other if jA < iB or jB < iA. In Figure 2, self-crosses C and D are parallel to each other and both are contained in B. Self-cross B is contained in A. If a self-cross corresponds to the node of a tree, and the containing relationship between the self-crosses corresponds to the father-son relationship between nodes, the tree for Figure 2, is as shown in Figure 3.

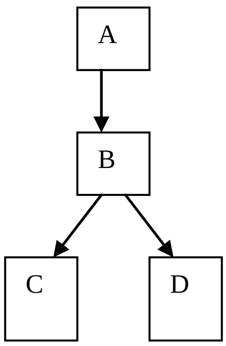


Figure 3. Tree that Corresponds to Figure 2

The detection algorithm for self-crosses produces a tree based on the order of coding edges. The treatment for self-crosses C and D must occur prior to self-cross B because the treatment for the latter relies on the treatment for the former. Similarly, the treatment for self-cross B must occur prior to A because treatment for the latter relies on the treatment for the former. Therefore, the treatment order for self-crosses is opposite to the order of traversing the tree using the width-first strategy.

The splitting treatment for self-crosses involves two parts: determining whether a new contour is produced based on the length of the new contour, and correcting the length of the original contour. Assume that the node that corresponds to the self-cross studied in the tree is P and its serial number pair is (iP,jP). Let the length of the new contour produced by self-cross P be Lp, which is equal to jP − iP. Assume that a son of P is S if P is a parent node. Let the length of the new contour produced by self-cross S be Ls. The rules for the splitting treatment for self-crosses are presented as follows:

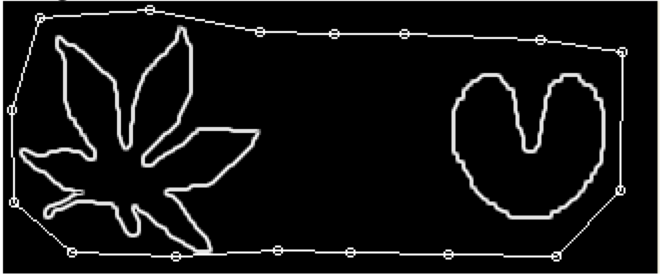
1) If P is a leaf node and Lp > 2, one new contour is produced;

2) If P is a leaf node and Lp =< 2, self-cross P and the edges composed of it are eliminated;

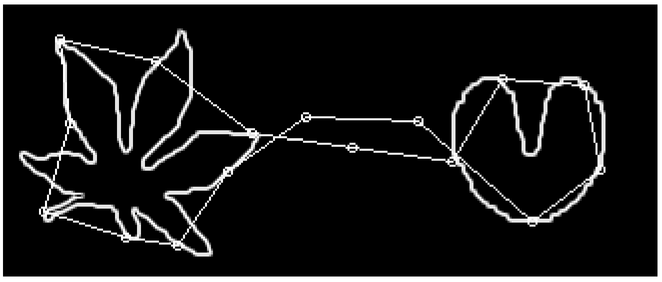
3) If P is a parent node and Lp − Ls > 2, one new contour is produced and Lp is replaced by Lp-Ls;

4) If P is a leaf node and Lp − Ls =< 2, self-cross P and the edges composed of it are eliminated;

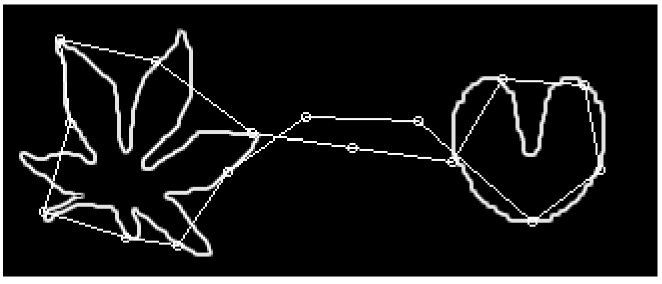
According to the previous rules, treatment of the contour in Figure 2, generates three sub-contours. Self-cross D satisfies rule 2 and does not generate a new contour. The sub-contour generated by self-cross C is eliminated along with PACM revolution in the end. The algorithm designed based on the previous rules is applied as shown in Figure 4, which shows that the algorithm is correct and effective.



(a)



(b)



(c)

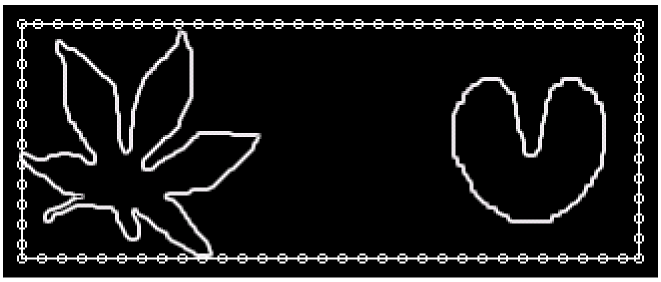
Figure 4. Self-cross Treatment: (a) Initial Contour to be Researched; (b) Contour Self-crosses After Evolution; (c) Treatment Results for Self-Crosses

After solving the time and treatment for splitting contours in PACM, the next step is to study how the objects are segmented completely from images. Although all the points of the left contour are located on the border of the flower shown in Figure 4, segmentation of the flower object from the image is not successful at the time of splitting the contour. The reason is that the points on the left contour are not evenly distributed, and the point interval is too large. For segmenting objects in images with PACM, we must study the way in which the distribution of points on contours is consistent with the shape of the object to be segmented.

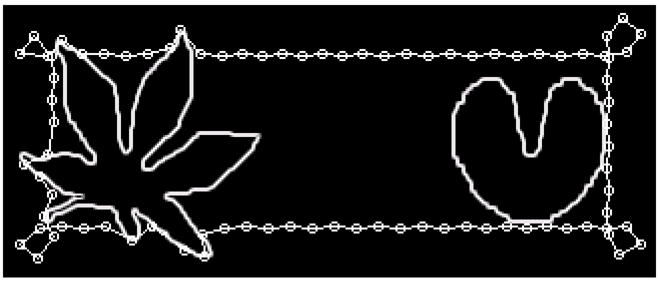
4. Discussion on Approach for Object Shape with Contours

Segmenting image objects with PACM demands for the contours to be close to the image objects, and for the contour shapes to be identical to the object shapes. Satisfying such demands is difficult because the shape and size of the objects are unknown before the objects are segmented. The interval between two points on a contour cannot be controlled to satisfy object shape because the interval between the points on the contour changes in the procedure for the evolution of the contour. Because the intervals among the points on a contour are shorter, and the contour is more suitable to the object shape, a practical method is for the intervals among the points on a contour to be maintained as short as possible. This is achieved by high-frequency sampling of the contour at the start, and by interpolation during evolution. After a round of evolution, an examination of whether the interval between two points on the contour is larger than a given threshold is performed. If the interval between the two points is larger than the threshold, interpolation is conducted between the two points. This method sacrifices evolution speed for segmentation precision.

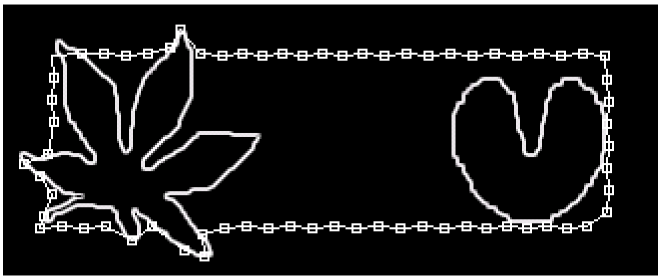
We find that the aforementioned method causes some new problems. When a contour is sampled by high frequency at the start, some points on the contour are redundant and cause more new self-crosses during the evolution. Figure 5(b), illustrates such situation. The self-cross caused by the redundant points must be treated, which increases the time for treating the self-crosses again.



(a)



(b)



(c)

Figure 5. Self-cross Caused by Redundant Points on Contour: (a) Initial Contour from High-frequency Sampling; (b) Self-cross Caused by Redundant Points; (c) Results After Treatment

The treatment of those self-crosses causes the interval between two points to exceed the threshold. Then, the distribution uniformity for points on the contour is changed in the model evolution because of the self-cross treatment. Figure 6, shows that contour non-uniformity is caused by the self-cross treatment.

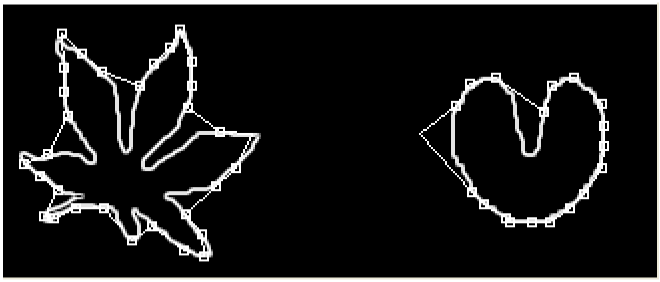


Figure 6. Contour Non-uniformity Caused by Self-cross Treatment

It is not possible for contour non-uniformity to be permanently solved after interpolation time. In order to permanently maintain the distribution uniformity of the contour points, the interval in each pair of points on the contour must be examined after a round of evolution. This procedure is described in the flowchart shown in Figure 7. The condition for exiting the cycle shown in Figure 7, is the problem to be researched. This is not same as the original exiting condition for PACM where all vertexes are located on the contours, because some new vertexes can appear after interpolation. Therefore, the condition for exiting the cycle shown in Figure 7, is that no new vertex appears and all intervals are less than the threshold.

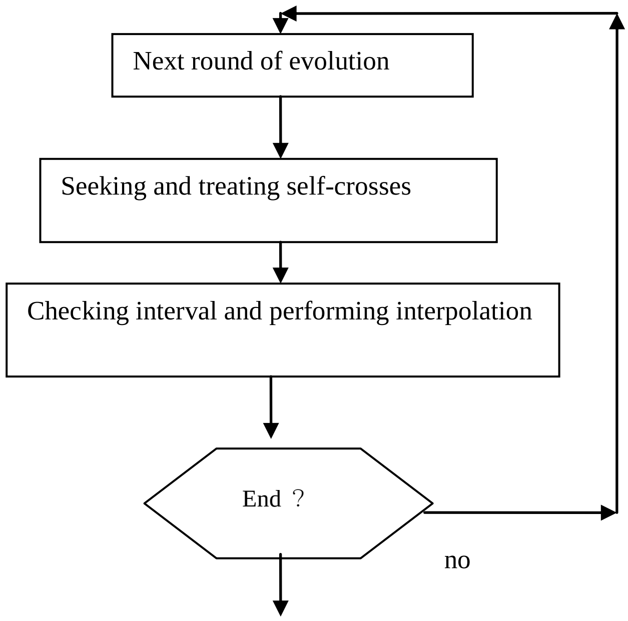


Figure 7. Interval Examination Flowchart and Self-cross Treatment

The contours shown in Figure 8, are formed from the contour shown in Figure 6, through interpolation. Figure 9, shows the evolution results after interpolation. In Figure 9, every point on the contour is located on the object edges.

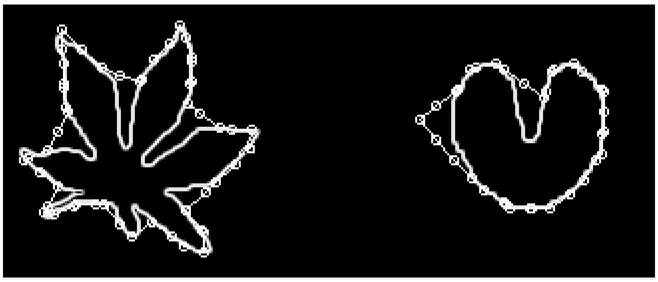


Figure 8. Contour after Interpolation of Contour from Figure 6

If all intervals are less than the threshold, evolution ends. We believe that if the contour shapes are not the same as the object shapes at the moment shown in Figure 9, or the distribution of the points on the contour is not uniform, contour interpolation continues. After interpolation, if there are some new points not on the object edges, contour evolution continues. This forms a treatment cycle that continues until no new points appear and all intervals are less than the threshold.

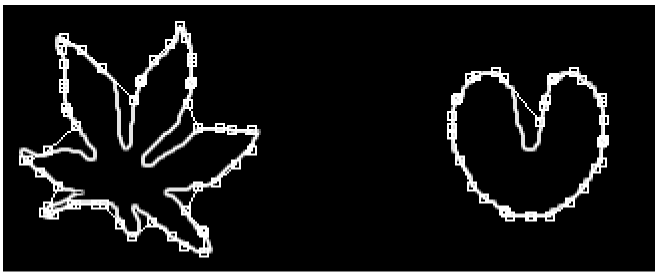


Figure 9. Results of Continuing Evolution after Figure 6

The threshold is related to object similarity to the contours. The threshold is reduced and object similarity to the contours increases. Currently, there is no practical method for determining the threshold because of edge irregularity in the objects and unknown object size. This method will be proposed in future study.

5. Conclusion

This paper is a dissertation on the topic of segmenting multiple objects from images with PACM in the shrink strategy. The paper started with current works on the segmentation of multiple objects, which mainly focus on PACM in the expansion strategy. These methods are very difficult to apply when enclosing one object with one contour. Segmenting multiple objects from images with PACM in the shrink strategy also achieves enclosing one object with one contour, and encounters different issues with PACM in the expansion strategy. These issues include the detection of self-crosses and splitting contours. This paper proposed a detection algorithm for self-crosses and an algorithm for splitting contours. The problem of approaching object shapes to contours was also discussed.

Acknowledgments

The work of this paper is supported by the nature funds of Anhui province (code:1308085QF113).

References

[1] M. Kass, A. Witkin, and D. Terzopoulos, “Snake: Active Contour Models,” International Journal of Computer Vision, **(1988)**, pp. 321-331.
[2] L. D. Cohen and I. Cohen, “Finite Element Methods for Active Contour Models and Balloons for 2D and 3D Images,” IEEE Transactions on Pattern Analysis and Machine Intelligence, **(1993)**, pp. 1131-1147.
[3] C. Xu and J. L. Prince, “Snakes, Shapes and Gradient Vector Flow,” IEEE Transactions on Image Processing, **(1998)**, pp. 359-369.
[4] P.-h. Li and T.-w. Zhang, “Review on Active Contour Model (Snake Model),” Journal of Software(Chinese), **(2000)**, pp. 751-757.
[5] C. Bo and L. Jianhuang, “Active Contour Models on Image Segmentation: A Survey,” Journal of Image and Graphics, **(2007)**, pp. 11-19.
[6] T. Srinark and C. Kambhamettu, “A Framework for Multiple Snakes and its Applications,” Pattern Recognition, **(2006)**, pp. 1555-1565.
[7] C. H. Chuang and W. Nunglie, “Automatic Multi-object Segmentation by Two-phase Snake Processing,” Asian Journal of Health and Information Sciences, **(2006)**, pp. 296-320.
[8] B. N. Saha, N. Ray, and H. Zhang, “Snake Validation: A PCA-based Outlier Detection Method,” IEEE Signal Processing Letters, **(2009)**, pp. 549-552.
[9] B. N. Saha and N. Ray, “Automating Snakes for Multiple Objects Detection,” Computer Vision—ACCV 2010 (10th Asian Conference on Computer Vision), **(2011)**, pp. 39-51.
[10] A. Nakhmani and A. Tannenbaum, “Self-Crossing Detection and Location for Parametric Active Contours,” IEEE Transactions on Image Processing, **(2012)**, pp. 3150-3156.

Author

Liu Hongshen, received his BS degree in physics from the Anhui Normal University in 1988 and MS degree in Physical chemistry from Dalian Institute of Chemical Physics, Chinese Academy of Sciences in 1991. He received his qualification certificate of senior programmer from Ministry of Personnel and Ministry of Electrics Industry of People’s Republic of China in 1994. His current research interests include image processing, pattern recognition and design of software.