DETECTION AND TREATMENT OF THE SELF_CROSS IN
THE SHRINK PROCESSES OF PARAMETRIC ACTIVE
CONTOUR MODEL

1LIU HONGSHEN,2WANG NAN,3RUAN YUE
1Prof., School of Computer Science, Anhui University of Technology, Ma’anshan Anhui, 243002
2Master Degree Candidate, School of Computer Science, Anhui University of Technology, Ma’anshan
Anhui, 243002
3Assoc. Prof., School of Computer Science, Anhui University of Technology, Ma’anshan Anhui, 243002
E-mail: 1lhslhs1@ahut.edu.cn, 2247401443@qq.com, 3878374858@qq.com

ABSTRACT

The research object of this paper is the self_crosses of contour in parameter active contour model with
shrink strategy (PACMS). For studying the object, parameter active contour model is improved to rapidly
converge and reduce number of the self_crosses of contour. The concept of self_cross in PACMS is
introduced and the influence of it on the evolution of PACMS is presented. With the improved model, the
existence of self_cross in PACMS is shown. The treatments of self_cross on contours in the current
literatures are analyzed. A model of describing the self_crosses is set up and an improved method of
detecting and treating the self_crosses in PACMS is put forward. The experiment results of the new method
are given and the comparison among the new method and other methods of detecting and treating the
self_crosses are made. The result shows that the new method is more efficient than the other method of
detecting self_crosses on contours.

Keywords: Parametric Active Contour Model, Parameter Active Contour Model With Shrink Strategy, The
Self_Cross Of Contour, Detection, Treatment

1. INTRODUCTION

Parametric active contour model\([\text{1}]\) (also known as snake, PACM) is the primary method of
segmenting images and tracking videos. Its shortcomings are that the contour is hard to reach
the depression area of the objects in images and the initial contour is required near to the objects as
possible. These limit its applications. Because of those, many researchers have done some works to
improve PACM and obtained some meaningful results\([2-10]\). These works promote the application of
PACM. But they still can not solve all problems of applying PACM to segment objects of images and
track videos. In fact, the shapes of objects are random and irregular and it is difficult to require
contours evolution to the edges of objects under the restriction of tension and curvature of the
contours. Because this restriction of tension and curvature makes the contour short and smooth, the
restriction is of positive role in the beginning of evolution and negative effect in the late, which it
prevents the contour moving towards the hollow
parts of objects. If the restriction is eliminated completely, the contours generate more much self_crosses in the evolution. That implied that the influences of tension and curvature on the contours must be modified along with the evolution of contours by changing their weights. But it is difficult how to change their weights in order to make the influence in a moderate degree. If a contour is set randomly, there are self_crosses of the contour, which they are inevitable by modifying the weights of tension and curvature. The self_crosses must exist in the evolution of contours when multiple objects are segmented from images with PACM and the time when it appears is that time when many objects are segmented. So it is meaningful to study the self_crosses of contour in PACM\[7-11\].

For studying the self_crosses of contour, we present an improvement of PACM in Section 2, describe how the self_crosses generate in the shrinking progress of contours and they influence on their application in Section 3. Detection and treatment of self-crosses are described in Section 4. In Section 5, we present some results and some discussion of our algorithm.

2. IMPROVEMENT ON PACM FOR RESEARCHING THE SELF_CROSSES OF CONTOURS

In order to study the self_crosses of contours, PACM must be improved. The goal of improving PACM keeps the contour evolution inward and reduces the number of self_crosses as possible. The discrete evolution equation in PACM is shown as follows:

\[ E_{int}(i) = \frac{1}{2} \left( \alpha \left( v_i^2 - v_{i-1}^2 \right) + \beta \left( v_{i-1} - 2v_i + v_{i+1} \right) \right) \]

\[ (i=0,1,2,...,n) \quad (1) \]

The progress of iteration evolution makes the sum of the energy \( E_{int}(i) \) minimum. The first order derivative of formula (1) is the elastic energy of contours and makes contours short. The second derivative is the rigid energy and makes contours smooth. These two parts contain no directivity information and don’t keep contours evolution inward\[5\]. Distribution of a part of pixels around one contour is shown as figure 1 and the black pixels form a segment of the contour. If contours are defined as rings with the clockwise or anti-clockwise directivity, the inner positions of the contour in figure 1 are exactly the opposite when passing the segment in figure 1 on two different directions. If the segment is searched in the sequence a and the contour is of the clockwise, the pixels in the contour are left. If the segments are searched in the sequence b and the contour is of the clockwise, the pixels in the contour are right. For the parameter active model with shrink strategy (PACMS), the energy of pixels in the contour and outside the contour must be different to ensure shrinking in the evolution of the contour. But the pixel energy calculated with the formula (1) is same. So the improvement of PACM is that one direction coefficient \( C \) is added at second derivative and the energy formula improved is shown as follows:

\[ E_{int}(i) = \frac{1}{2} \left( \alpha \left( v_i^2 - v_{i-1}^2 \right) + \beta \left( v_{i-1} - 2v_i + v_{i+1} \right) + c \cdot \left( 2v_i - v_{i-1} - v_{i+1} \right) \right) \]

\[ (i=0,1,2,...,n) \quad (2) \]

Figure 1: Directivity Of Searching Contours
Figure 2: Definition of Directive Number

Value of C is -1 for pixels in contours and +1 outside contours. In greedy algorithm, the candidate pixel of the pixel researched is located in its 8-neighbourhood, in which some pixels is in contours and the other is outside contours. C of a pixel in the 8-neighbourhood is related with its position i, j and calculated by the searching path. The searching path is described by two parameters, which one is called in_edge directive number and another is called out_edge directive number. The directive number is defined in figure 2. Let r be in_edge directive number and k be out_edge directive number, C is presented as such:

$$c_{ij} = \chi(r, k) \quad r, k \in \{0, 1, 2, 3, 4, 5, 6, 7\}$$

r, k are determined by the quadrant which edges locate in and whether the absolute value of $$\frac{y_j - y_i}{x_j - x_i}$$ is greater than 1, which $$x_i, y_i, x_j, y_j$$ are the coordinates of two endpoints in the edge. For example, the quadrant which in_edge locates in figure 2 is 1 and the absolute value of $$\frac{y_j - y_i}{x_j - x_i}$$ of in_edge is greater than 1, the quadrant which out_edge locates in figure 2 is 4 and the absolute value of $$\frac{y_j - y_i}{x_j - x_i}$$ of out_edge is also greater than 1, so those red pixels C are -1 and the other are +1 (see fig.2). Weight α, β are separately set 2.0 and 0.6 in the later experiment. The result shows that direction coefficient C prevents contours' evolution outside and along tangential direction, and indeed reduces the number of self_crosses in contours.

3. Generation of the Self_crosses in the Shrinking Progress of Contours and Their Influence on Application

There are two types of evolution strategy in PACM. One evolution strategy keeps contours shrinking and the other makes contours expanding. The self_crosses of contours happen on the parameter active models with shrinking strategy (PACMS). The factors of producing self_crosses in PACMS have many. To sum up them has the following several aspects. @degree of defining the inner force of contours correctly and setting weights of α, β, γ accurately. @irregular shape of the initial contour. @density of vertexes on the initial contour. @distribution of data in images, such as multiple objects. Figure 3 is an example which the irregular shape of contour causes the self_crosses of contours (the double edges of the object in the figure are produced by Sobel operator). Figure 12 b) is an example which the high density of vertexes on the contour causes the self_crosses of contours. Figure 4 is an example which the distribution of data in images causes the self_crosses of contours. The four factors affect the self_crosses of contours in different degree. The distribution of data in images certainly causes the self_crosses of contours and the others may cause the self_crosses of contours. Improvement on defining the inner force of contours may decrease the number of self_crosses in evolution, and do not completely eliminate the self_crosses (just as our work). For the irregular shape and the vertex density of initial contour, although the number of self_crosses can be decreased by artificially setting the initial contour, this require reduces the application arrange of
PACMS because the demand for the regular shape limits the arbitrary of initial contour. In order to keep the shape of area segmented same as the shape of objects; the high density of vertexes on contours is required.

4. DETECTION AND TREATMENT OF THE SELF_CROSSES IN THE EVOLUTION OF PACMS

4.1 Detection of self_crosses in the evolution of PACMS

The tasks of detecting self_crosses in the evolution of PACMS contain two parts. One determines whether the self_cross happens on in the evolution of PACMS and the other locates the self_cross if a self_cross exists in contours. So detecting self_crosses in the evolution of PACMS
includes two steps: the first step looks for self_crosses and the second step locates the self_cross which is found\textsuperscript{[14]}. Now there have been three methods for detecting the self_crosses. The first method is based on T_Snake model\textsuperscript{[11]}. The second method is detecting the self_crosses of contours by linear interpolation and sorting\textsuperscript{[12]}. The third is based on modern differential geometry\textsuperscript{[14]}. These methods have some shortcomings. For example, the method based on T_Snake model can not detect all self_crosses and the self_crosses which their size are less the size of the grid in T_Snake model are not able to be found. The method based on T_Snake model can not obtained also the position of intersection. The second method determines the self_crosses of contours by calculating the coordinate of intersection in two lines, and is complicated and unnecessary. The third method detects self_crosses by calculating the turn number\textsuperscript{[14]} and is sophisticated to treat when there is even number of self_crosses in contours.

The time of detecting the self_crosses is between two turns of evolution. Number of times which the detecting algorithm is executing is same as the number of evolution turns. So the efficiency of the detecting algorithm greatly affects the whole progress of evolution of PACMS and it is essential to design a high performance algorithm of detecting the self_crosses of contours. PACMS is used for segmenting images and tracking video by scanning the whole image area in the evolution of contours. The characteristic data of images are investigated and determined whether the contours reach to the edges of objects in the progress of scanning. The intersection of two disconnect edges in a contour predicates that there is no characteristic data in the area which the two edges have scanned and the area must not be focused. Therefore, the task of locating self_crosses determines which two edges happen on intersection without calculating the coordinate of their intersection. The traditional method estimating intersection of two line segments is the projection method. This method determines intersection of two line segments by means of the overlap situation that the projections onto X and Y axis of two lines. The traditional method is simply and not efficient. The method of the cross product of vectors is efficient in detecting self_crosses of contours\textsuperscript{[15]}. Here an improvement method of the cross product of vectors is used to detect the self_crosses on contours. Let two edges be $P_1P_2$ and $P_3P_4$. We regard the two edges as two vectors $\overrightarrow{P_1P_2}$ and $\overrightarrow{P_3P_4}$. Two new vectors $\overrightarrow{P_1P_3}$ and $\overrightarrow{P_2P_4}$ are defined as follows (see fig.6);

\begin{center}
\includegraphics[width=0.5\textwidth]{figure6.png}
\end{center}

**Figure 6**: Intersection of two vectors

The intersection conditions of $\overrightarrow{P_1P_2}$ and $\overrightarrow{P_3P_4}$ include:

1. $\overrightarrow{P_1P_3}$ and $\overrightarrow{P_2P_4}$ locate on two sides of $\overrightarrow{P_3P_4}$, the direction of $\overrightarrow{P_3P_4} \times \overrightarrow{P_1P_3}$ is opposite from the direction of $\overrightarrow{P_3P_4} \times \overrightarrow{P_2P_4}$, that is, the signs of the cross product of $\overrightarrow{P_3P_4} \times \overrightarrow{P_1P_3}$ and $\overrightarrow{P_3P_4} \times \overrightarrow{P_2P_4}$ are different.

2. $\overrightarrow{P_1P_3}$ and $\overrightarrow{P_2P_4}$ locate on two sides.
of $p_1p_2$; the direction of $p_1p_2 \times p_1p_3$ is opposite from the direction of $p_1p_2 \times p_2p_4$, that is, the signs of the cross product of $p_1p_2 \times p_1p_3$ and $p_1p_2 \times p_2p_4$ are different.

Two vectors are collinear when the cross product of them is zero. Let the coordination of the point $P_i$ be $(X_i, Y_i)$. Let

$$F_1 = p_1p_2 \times p_1p_3 = (X_4 - X_3)(Y_3 - Y_1) - (Y_4 - Y_3)(X_3 - X_1)$$
$$F_2 = p_1p_2 \times p_1p_4 = (X_4 - X_3)(Y_4 - Y_2) - (Y_4 - Y_3)(X_4 - X_2)$$
$$F_3 = p_1p_3 \times p_1p_2 = (X_2 - X_1)(Y_3 - Y_1) - (Y_2 - Y_1)(X_3 - X_1)$$
$$F_4 = p_1p_4 \times p_1p_2 = (X_2 - X_1)(Y_4 - Y_2) - (Y_2 - Y_1)(X_4 - X_2)$$

The intersection condition of $p_1p_2$ and $p_1p_3$ is described as follows:

$$F_1 \cdot F_2 < 0 \land F_3 \cdot F_4 \leq 0 \lor F_1 \cdot F_2 \leq 0 \land F_3 \cdot F_4 < 0 \quad (3)$$

4.2 Treatment of self_cross in the evolution of PACMS

The self_cross must be treated on time after it is found, or else it affects the next turn of evolution in PACMS. The treatment ways of self_crosses are related to the reasons which cause the self_crosses. The self_cross caused by multiple objects is treated by splitting the original contour into two contours. The self_cross caused by the others reasons are treated by discarding the segment between the two intersecting edges. So the first step of treating self_crosses determines which reason causes the self_cross in the evolution of PACMS.

There is only a contour at the beginning of the evolution of PACMS and more sub contours will be generated along with the evolution of PACMS. These new contours may produce more sub contours again. Some of these new contours may be preserved and the others may be discarded. The method distinguishing them must be researched. Let's start with the easiest situation which one initial contour and one sub contour exist. In that situation, the judgment changes along with the evolution of PACMS: at the initial stage of evolution, which all vertexes of the contours are not on the edges of the objects and the self_crosses are caused by the shape irregularity or the high vertexes density of contours, the length of redundant part is much shorter than the segment preserved; in the middle and late stage of the evolution of PACMS, which some vertexes may be on the edges of the objects, if a vertex of contours is on the edges of objects, the segment of the vertex is preserved (see fig.7). The judgment of vertexes on the edges of objects is based on the characteristic data in the vertex $P_{image}$. If $P_{image}$ of a vertex is greater than a threshold value $P_{level}$, the vertex is on the edges of objects.

Figure 7: The segment discarded and the segment preserved when happening on self_crosses

The general situation in the evolution of PACMS is that parent contour evolutes into multiple child contours (see fig.12 b)). The method of judging that a new contour is or is not preserved need be further studied in the general situation. This situation is described as follows:

Let a contour $C = \{x(i), y(i)\}, i \in [0, M]$. Here $M$ is the number of vertexes on the contour $C$, also is the number of edges. $i$ is the order number of vertexes or edges on the contour $C$. Let self_crosses at the
moment of evolution be \( J_i(k_i, l_i), i = [0, n] \). Here \( n \) is number of self_crosses (number of pairs of edges), \( k_i \) and \( l_i \) respectively are the order number of edges in self_crosses, \( i \) is the order number of self_crosses and \( J_i \) is the name of self_cross. This hints that \( M \) edges of the contour \( C \) are split into \( 2n \) segments. The treatment method can distinguish which segment is the discarding segment in the \( 2n \) segments. Assume that the distribution of the \( n \) self_crosses and \( 2n \) segments is shown as figure 8. The dotted lines indicate the redundancy segments and the dashed lines indicate the preserved segments.

Studying the \( 2n \) segments need search the graphical structure data in fig.8 and the order of studying the \( 2n \) segments must be determined. In fact, the relations among these self_crosses or segments are of tree structure characteristic. Segments which are corresponding with the parent nodes of the tree contain the segments corresponding with the child nodes of the tree.

Tree structure of figure 8 is shown in figure 9. Integer pairs in brackets of figure 9 are the order numbers of edges happening on self_cross.

The self_cross information of contours can be stored in a 2-dimensional array or a list. The root node which is corresponding to the first segment locates on the beginning of the array or list. The order of items in the 2-dimensional array is same as the order of self_crosses detected. If \( J_1(k_1, l_1) \) is the parent node of \( J_2(k_2, l_2) \), \( J_1(k_1, l_1) \) contains \( J_2(k_2, l_2) \), that is, \( k_1 \leq k_2 \) and \( l_1 \geq l_2 \). If \( J_1(k_1, l_1) \) and \( J_2(k_2, l_2) \) are leaf nodes at the same level, \( [k_1, l_1] \cap [k_2, l_2] = \emptyset \), that is, \( [k_1, l_1] \) and \( [k_2, l_2] \) are not overlap.

Treatment of the self_crosses need traverse the
whole tree and the traversal order is from the beginning of the leaf nodes at lowest level. The treatment procedure is that the leaf node is cutted, judged and marked. While all leaf nodes of a parent node have been treated and cut, the parent node becomes a leaf node. This progress continues until the root node is left. The treatment algorithm is described as follows:

All items in the array storing the tree are investigated according to the following way from the beginning of the latest item:

1) pick out one node or one integer pair (self_cross), Let the integer pair be (k, l). If it is root node, the section investigated is \([l, M] \cup [0, k]\), or else the section investigated is \([k, l]\). Go to next step;

2) examine all vertexes in the investigated section as follows:
   a) If the current vertex has been visited, the flag of leaf node is set to false and go to 2);
   b) Examine the characteristic data of vertexes in the section.
      b1) if the characteristic data \(P_{image}\) of the current vertex in the image is greater than \(P_{level}\), then the segment is preserved and is recorded, go to 5);
      b2) go to 2);

3) If the flag of leaf node is true and \(|k-l|<<M-|k-l|\), then the segment between k and l is discarded and make the discarding flag

4) If the flag of leaf node is false and the preserved flag is false, the segment between k and l is discarded and make the discarding flag except those segments corresponding to the child nodes it contains

5) If the discarding flag is true, a new contour produces and all vertexes are marked preserved (not including those segments corresponding to the child nodes it contains); Or else all vertexes are marked discarded (not including those segments corresponding too the child nodes it contains)

5. THE EXPERIMENTAL RESULTS AND THE DISCUSSION OF DETECTING AND TREATING ALGORITHM OF SELF_CROSSES

The above detecting and treating algorithm is applied to figure 3 b) and figure 4 b) and the results are shown in figure 10. Figure 10 shows that the algorithm is effective for treating self_crosses. The algorithm is used to segment multiple objects and the result is shown in figure 11. Figure 11 illustrates the algorithm success in segmenting multiple objects.

![Image](image1)

(a) The treating result of Figure 3b)

![Image](image2)

(b) The treating result of figure 4b)

Figure 10 the treating results of the algorithm

Figure 12 is same as figure 11 except the initial contour. The initial contour in figure 12 is produced automatically by a program. Figure 12
shows that high density of vertexes very easily lead to self_crosses of contour. In figure 12, one initial contour produces four sub contours. Figure 13 shows the treatment result for a actual image.

Figure 11 treating self_cross and segmenting multiple objects

Figure 12 the self_cross caused by the high vertex density of contour
d) The final result

Figure 13 segmenting the actual image of Chinese characters

Here we make the comparison among the traditional projection algorithm, the algorithms of literature\textsuperscript{[15,16]} and the above algorithm in detecting self_crosses. We respectively apply the traditional projection algorithm, the algorithm of literature\textsuperscript{[15]}, the algorithm of literature \textsuperscript{[16]} and the above algorithm in detecting self_crosses in Figure 11 b) and Figure 12 b) and the result is shown in Table 1. The Unit of data in Table 1 is the time interval of the counter in author's pc and the data are the time of detecting self_crosses, not including the treating time. The data in Table 1 shows that our algorithm is most effective in detecting self_crosses.

<table>
<thead>
<tr>
<th>No.</th>
<th>L</th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
<th>A4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20</td>
<td>316782</td>
<td>17329</td>
<td>277836</td>
<td>16835</td>
</tr>
<tr>
<td>2</td>
<td>78</td>
<td>435637</td>
<td>285909</td>
<td>371072</td>
<td>283621</td>
</tr>
</tbody>
</table>

In Table 1, “No” column is No of the contour studied, No 1 and No 2 are the contours in figure 11b) and figure 12b). “L” column is length of the contour studied, “Ai” column are the spending time of the following algorithms:

A1: traditional project algorithm
A4: Our algorithm

As mentioned above, self_crosses of contours in PACMS appear inevitably and the study about them is of significance. The algorithm of detecting and treating self_crosses in contours makes the restriction condition of inner force in PACMS much more relaxed, even unnecessary. It solves not only the problem of the convergence of the depressed area, but also eliminates the influence of self_crosses on further evolution of PACMS. Detection and treatment of self_crosses are meaningful for segmenting multiple objects and it is the key which one contour becomes multiple contours\textsuperscript{[10,11]}. The study on self_crosses of contours finds also some problems to be solved. The main problem is that treatment of self_crosses may cause the distribution density of vertexes in contours uneven. The problem will affects segmentation of multiple objects.

REFERENCES


