A New Flood-Fill Algorithm for Closed Contour

Roman Khudeev

Alparysoft R&D, Tomsk, Russia

***Abstract* — In this article, a new flood fill method is described. The method is fast and can be applied to both simple and self-crossing contours. It also can be used to fill the area outside the given contour.**

***Index Terms* — filling of contour**

There are several ways to make flood-fill for closed contour [1-5], but these variants demand the large calculations near contour and as a result because of low processing speed. More existent algorithms do not allow to fill complicated contours which contain self-crossings.

The suggested variant is characterized by simple working logic, simple realization and high processing speed. The given method does not use trigonometric functions, multiplication and division operations and recursive algorithms. In contrast to recursive algorithms in which the quantity of verification reaches 4 or even 8 (depend upon algorithm and contour type) for each dot of the contour the quantity of verification in our algorithm in average does not exceed 2 for each dot of the contour. This quickens lot the process of contour flood fill for the cases in which the quantity of dots inside the contour greatly exceeds the quantity of the contour dots. More than that our method allows to fill self-crossing contour both the whole one and any of its parts.

So, the task is:

- There is an array of dots X,Y size.
- There is closed 4 or 8 connected contour. The color of its dots differs from the background (fig. 1).
- It is necessary to fill all the areas which are located inside or outside the given contour.

The description of the algorithm is following.

0-7803-9219-1/05/$20.00 © 2005 IEEE

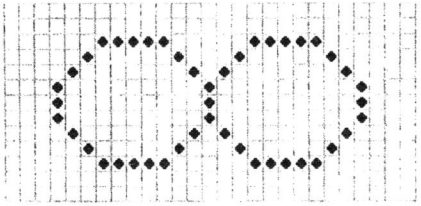

Fig. 1.

The essence of the algorithm is that it is necessary to build other contour (external contour) over the given one (internal contour). And then it is necessary to fill the areas per line in which between two neighboring dots of the external contour there is at least one dot of the internal contour. For doing this it is necessary to make the following steps:

1. It is necessary to find any dot of external contour. This dot can be set or it is necessary to scan an image per line until get to one of the external contour dots (fig. 2).

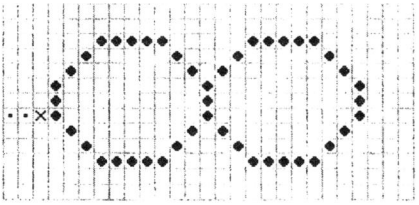

Fig. 2.

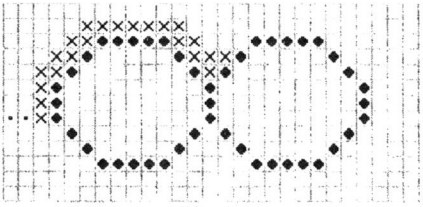

Fig. 3.

2. Then we build external contour. For doing this we should go around the given contour in series and any direction from the point

which we had found in item 1. During this we should mark the dots of the external contour (fig. 3).

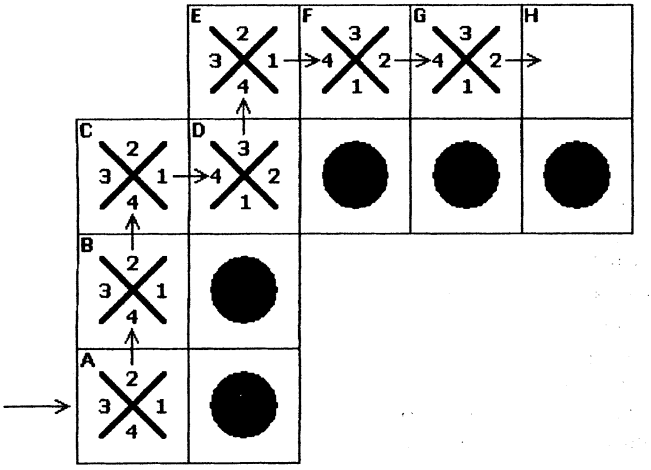

Fig. 4.

The algorithm of external contour building can be as follows: we have the first dot of the external contour (marked with letter A on the fig. 4). It was found during scan per line. For this dot and the following dots it is necessary to determine any "free" direction and make a step in this direction and mark this dot as a next dot of the external contour. Then this should be repeated until the external contour will be completed. The "free" direction is one of four directions which does not lead to a dot of the internal contour when we made a step. The direction is "free" also when we get to a dot of the external contour when we make a step. The examination is directions must be done either clockwise or anticlockwise always. The sequence of the examination of directions for A-G dots of the external contour marked with digits on fig. 4. At first the direction 1 must be examined then 2, 3 and 4. The rule for choosing the direction 1 is the following: the direction is the same as the direction to the closest dot of the internal contour for the first dot; for all the rest of the dots the direction 1 is the next direction anticlockwise from the direction we

are coming from, i.e. the direction we are coming from always is number 4. To clarify the process fig. 4 shows the sequence of the examination of directions for several dots of the external contour. The rules described above allow to build the external contour with great speed, i.e. with minimal quantity of examinations of "free" directions.

The process of the external contour building will be finished when the current dot coordinates is the same as the first dot coordinates (fig. 5).

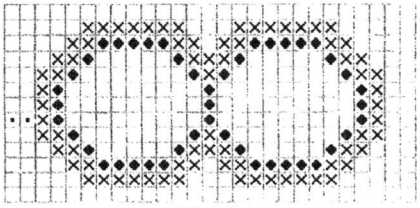

Fig. 5.

3. Now it is time to scan per line the rectangle in which the contour is inscribed. When at least one dot of the internal contour is between two neighboring dots of the external contour we should fill a raw of dots part which is located between two neighboring dots of the external contour. If it is necessary (e.g. the flood fill color is not the same as the contour color) it is possible to exclude from filling the internal contour dots (fig. 6, 7).

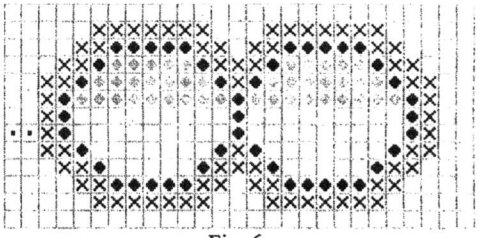

Fig. 6.

The same principle is for filling all the dots outside the given contour. For doing this after the external contour building it is neces-

sary to scan per line and if there are no dots of the internal contour between the edge of the image and the dot of the external contour or if where are no dots of the internal contour between two neighboring dots of the external contour then all the dots can be filled with the filling color.

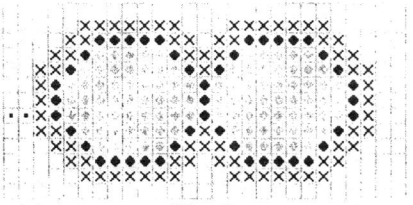

Fig. 7.

If it is necessary to fill the area inside the contour it is possible to do it upon the same principle. But in this case the given contour should be considered as the external one. Then the internal contour is built and if there are no dots of the external contour between two dots of the internal contour then all the dots between the dots of the internal contour can be filled with the color of filling.

So the given algorithm in spite of other ones allows to fill all the internal areas both the simple contour and self-crossing one of any configuration. Also it is possible to fill any area of a self-crossing contour. More than that our algorithm works faster for the contours which have much more dots inside the contour than in the contour.

REFERENCES

[1] J. Encarnacao, „Einfurung in die Graphische Datenverarbeiterung". Eurographics '89. Tutorial Notes 1. Hamburg, FRG, September 4–8, 1989. 122 p.

[2] D.F. Rogers, "Procedural Elements for Computer Graphics", McGraw-Hill, Boston, MA, 1998.

[3] P. Shirley, "Fundamentals of Computer Graphics", AK Peters Ltd, 2002.

[4] http://alglib.sources.ru/graphics/fillarea.php

[5] http://algolist.manual.ru/graphics/fill.php