Journal of Advances in Engineering Science
Section C (4), January - June 2010, PP 57-70

VECTOR ALGEBRA BASED TRACING OF EXTERNAL AND INTERNAL BOUNDARY OF AN OBJECT IN BINARY IMAGES

Nitin L. Narappanawar[1]*, B. Madhusudan Rao[1],Srikanth T.[1] and M. A. Joshi[2] (FIET)

1 - International Institute of Information Technology, P-14 Rajiv Gandhi Infotech Park, Hinjewadi, Pune-411057, India
2 - Collage of Engineering, Wellesely Rd, Shivajinagar, Pune, -411005, India

ABSTRACT

The present work details an algorithm using vector algebra for tracing a boundary of a binary pattern. More specifically, the paper provides a theoretical framework for boundary tracing and provides a basis for boundary tracing task. Core concept in the present work is the idea of seeking minimum angle between a vector formed by present and previous boundary locations, and the vectors formed by present and next possible boundary locations, using vector algebra. The proposed algorithm uses both cross and dot product for the choice of the minimum angle. This algorithm overcomes the drawbacks of heuristic approaches and provides a complete solution to the tracing problem. These include solutions taking into account a variety of contour possibilities external or internal on one hand and open or closed on the other, and the combinations thereof.

Use of sound mathematical basis in this tracing algorithm can avoid the drawbacks inherent in heuristic approaches. The paper includes proof for this algorithm providing 100% sensitivity and completeness. The trial runs performed on more than 1300 test images yielded a set of ordered boundary pixels 100% of the time.

Keywords: *Boundary following; Boundary traversing; Contour tracing; External boundary tracing; Internal boundary tracing.*

I. INTRODUCTION

Humans beings use shape of an object as one of the key distinguishing factors to identify or classify an object. The shape of an object is essentially captured in its boundary, as boundary is the place where the object ends and the background starts. Hence, it is natural to use the descriptors that describe the boundary as an object classifier or identifier. Based on this theme many boundary descriptors have been developed. A few boundary descriptors have been summarized by Ashbrook and Thacker [1]. Some of these are Fourier descriptors [2] [3], Hough transform based descriptors [4] [5] and Chain Code descriptors [6] [7]. A relatively new boundary based descriptor can be found at [8]. Boundary descriptors are used in wide range of applications related to automatic interpretation of images containing segmentation results, printed and hand written documents, maps, drawings etc. The boundary descriptors require an ordered set of connected pixels (in clockwise or anticlockwise sense) which define the boundary of the object. The process of obtaining an ordered set of boundary pixels is known as boundary tracing or following.

In a digital binary image, there are only two types of pixels, the region pixels and the background pixels. The region pixels essentially capture the object information. The boundary pixels are region pixels, which are defined by their connectivity with the background pixels [9]. In an image the number of boundary pixels is far less as compared to the region pixels. However, the boundary pixels concisely capture the extent and geometry of the region. Further, it is also more efficient to manipulate region (and thus the object) information using boundary. These properties make boundaries an attractive and economical representative of regions in image analysis.

* nitin_nl@rediffmail.com

The paper uses two types of connectivity; 4-way connectivity and 8-way connectivity as defined at [9]. In order to avoid connectivity paradox, the boundary pixels are assumed to be 8-way connected and the background pixels are 4-way connected [9]. This assumption implies that the boundary pixels are those region pixels which are 4-connected with the background. To trace a boundary is to locate an 8-connected path [10] of boundary pixels in a particular order (direction of tracing).

Refer to Figure 1, we propose the following definitions:

External Background: The external background pixels are 4-connected among themselves and they can be connected to an edge of the given image, using a 4-connected path of background pixels.

External Boundary: An external boundary is an 8-connected path (open or closed) of region pixels which are 4-connected with the external background pixels. It is possible that there may be disjoint clusters of external background pixels in an image. These clusters will become connected if it is assumed, without loss of generality, that there are external background pixels outside the image as well (i.e. padding the image by background pixels). In such situations, the external boundary may coincide with an edge of the image at certain locations. Further the situation may give rise to a case of multiple objects in an image, which can be addressed by taking up one object at a time. As a generalization if the image has multiple unconnected objects, the processing can be done considering one object at a time.

Internal Background: The internal background pixels are 4-connected among themselves and they can not be connected to any edge of the given image, using a 4-connected path of background pixels.

Internal Boundary: An internal boundary is an 8-connected path (open or closed) of region pixels which are 4-connected with the internal background pixels.

Section II reviews the existing methods (algorithms) for boundary tracing while highlighting their limitations. Section III presents the vector algebra based algorithm for tracing a closed external boundary and a closed internal boundary. The justification for the procedure used for locating the starting pixel and the direction of tracing in the case of internal boundary is presented in Appendix-I. Section IV gives the modifications required in the algorithm presented in Section 3, to trace an open boundary. It is proved in Appendix-II that the proposed algorithm provides 100% sensitivity and completeness.

II. REVIEW

Algorithms such as Square Tracing algorithm, Moore-Neighbor Tracing algorithm and the Theo Pavlidis' algorithm [11] have been used for boundary tracing. Theo Pavlidis algorithm results in erroneous results in some cases. An unsuccessful attempt to modify Theo Pavlidis algorithm can be found at [12]. The Moore-Neighbor tracing algorithm is principally

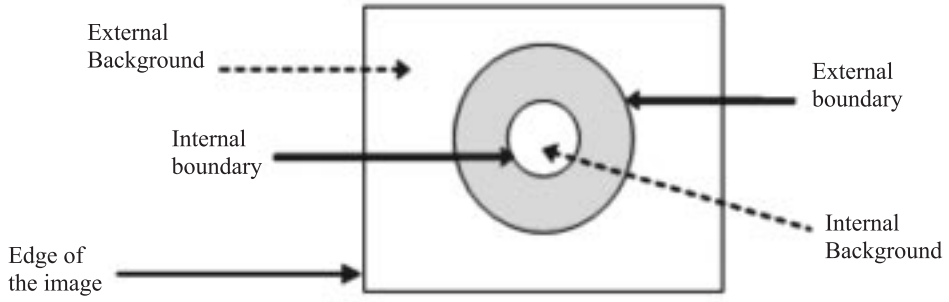


Figure 1 : Figure showing "Internal boundary", "External boundary", "Internal background", "External background" and "Edge of the image".

suited for tracing external boundary of a closed contour. In MATLAB® version 7.1, a boundary tracing algorithm is implemented. The MATLAB® documentation explicitly states the ambiguity in boundary tracing depending on starting pixel and the direction chosen for first step [13].

Consider Figure 2 (a). This figure is input to boundary tracing algorithm. Let the input "direction of tracing" be anticlockwise. The starting pixel for the tracing is indicated in Figures 2 (b) and (c), by S. For both Figures, 2 (b) and (c) the starting pixels are identical. However, direction chosen for the first step is "North" for Figure 2 (b) and it is "South" for Figure 2 (c). The traced boundaries are indicated using a gray shade. It can be observed that in Figure 2 (b) external boundary is traced, where as in Figure 2 (c) internal boundary is traced. Also actual direction of tracing the boundary is different in these figures (though the direction of search in the neighborhood is same, anticlockwise). In Figure 2 (d) the starting pixel as indicated by S, is different from that in Figure 2 (c). However, direction chosen for first step is same and is "South". It is observed that in Figure 2 (d) external boundary is traced as against the internal in Figure 2 (c). Also the actual direction of tracing is different in these two figures.

With reference to the discussions in the preceding paragraphs, following are the requirements of a boundary tracing algorithm:

1. The traced boundary: For identical inputs the traced boundary should be same in terms of content (pixel locations) and direction. This condition enables the next step which is boundary description.
2. Starting pixel: There may be a requirement for identifying a starting pixel which satisfies a certain specific condition. The specific condition satisfied by the starting pixel should be useful in initiating the boundary tracing.

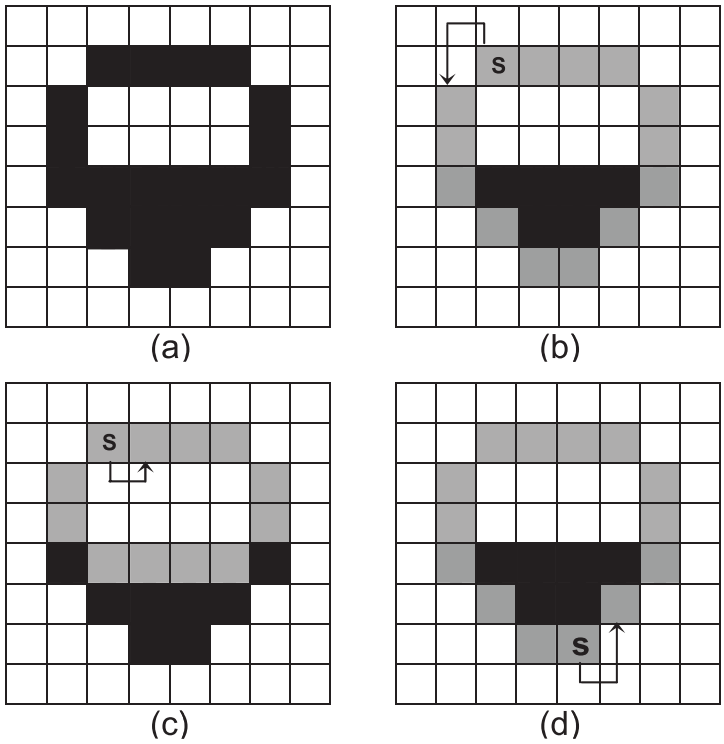


Figure 2 : Impact of "starting pixel" and "direction of first step" on boundary tracing. The shaded pixels indicate the traced boundary pixels where as black pixels indicate the remaining region pixels.

There may be multiple pixels on the same boundary that satisfy the condition. If the inputs to the tracing are identical except for the starting pixel, the traced output boundary should be same in terms of contents as well as direction, except for the changes in the output on account of change in starting pixel.

3. Locating a starting pixel: There should be a method to locate a starting pixel. There should be adequate justification that the method indeed locates the starting pixel.
4. Internal and external boundary: There are going to be binary images with internal and external boundaries. It is possible that there are common pixels between both these boundaries. It is also possible that a starting pixel is one of these common pixels. In such a case one can trace both the boundaries from the same starting pixel. Hence it should be possible to specify which of these boundaries is to be traced.

A. Preliminaries

Let $\hat{i}, \hat{j}$ and $\hat{k}$ denote unit vectors in the direction of X-axis, Y-axis and Z-axis respectively. Let O be the origin. It is assumed that the boundary that is being traced lies totally in XOY plane.

Using basic vector representation concepts, let

$$\overline{V_1} = V_{1x}\hat{i} + V_{1y}\hat{j} \text{ and } \overline{V_2} = V_{2x}\hat{i} + V_{2y}\hat{j}$$

Let θ be the directed angle from $\overline{V_1}$ to $\overline{V_2}$

Let $p = V_{1x}V_{2y} - V_{1y}V_{2x}$ and $q = V_{1x}V_{2x} + V_{1y}V_{2y}$.

$$\text{Let } \mathbf{J} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \quad \mathbf{V}_1 = \begin{bmatrix} V_{1x} \\ V_{1y} \end{bmatrix} \quad \mathbf{V}_2 = \begin{bmatrix} V_{2x} \\ V_{2y} \end{bmatrix}$$

Then in matrix notation $p = V_2^T J V_1$ and $q = V_2^T V_1 = V_1^T V_2$. Also, $\| \overline{\mathbf{V}_1} \| = \sqrt{V_1^T V_1}$.

Now, $\overline{V_1} \times \overline{V_2} = \| \overline{V_1} \| \, \| \overline{V_2} \| \sin(\theta) \, \hat{k} = p\,\hat{k}$ and

$\overline{V_1} \cdot \overline{V_2} = \| \overline{V_1} \| \, \| \overline{V_2} \| \cos(\theta) = q.$

In the present work, in the case of $\overline{V_1} \times \overline{V_2}$ the scalar multiplier of the unit vector $\hat{k}$ is of interest. Therefore, henceforth $S(\overline{V_1} \times \overline{V_2})$ would mean only the scalar multiplier, $\| \overline{V_1} \| \, \| \overline{V_2} \| \sin(\theta)$ or p. Hence $S(\overline{V_1} \times \overline{V_2}) > 0$ would mean $p > 0$.

The operator $\exp(\theta J) = I \cos(\theta) + J \sin(\theta)$, rotates a vector through an angle θ, in anticlockwise direction [14]. ("I" is the identity matrix).

If $\overline{V_1}$ represents position vector of point A and $\overline{V_2}$ represents position vector of point B, then (from basic vector algebra), $\overline{AB} = \overline{V_2} - \overline{V_1}$. Therefore, if A is the current pixel and B is the next pixel, $\overline{AB}$ = (next pixel) - (current pixel).

III. THE PROPOSED ALGORITHM FOR TRACING A CLOSED EXTERNAL BOUNDARY

The scanning is started from a corner of a rectangular image. The pixels adjoining the edges of the image are 4-connected with the edges. The scanning visits the pixels in a 4-connected manner. Therefore, all the background pixels visited till the first region pixel is encountered, are external background pixels. The first region pixel that is encountered (starting pixel) is 4-connected with the external background pixel(s) and hence, is a pixel on the external boundary of the object. Therefore, the boundary that would be traced from the located starting pixel is an external boundary.

For tracing contours, the algorithm for boundary tracing presented in this paper requires a geometrical condition that a starting pixel should satisfy, that it should be a convex vertex. The necessary mathematical backing for locating a convex vertex is provided at [15]. The vectors drawn from the current pixel to the neighboring boundary pixels are known as the connectivity (or current to next) vectors (CNV). Vector drawn from the current pixel to the traced previous pixel can be called as CPV. If the current pixel is a starting pixel CPV will not exist. But, as the starting pixel is always a convex vertex, the decision based on cross product will be appropriate. The algorithm essentially chooses the best of the CNVs to establish next connection in the anticlockwise or clockwise direction, as specified by the user. For explanation in this paper anticlockwise direction has been followed.

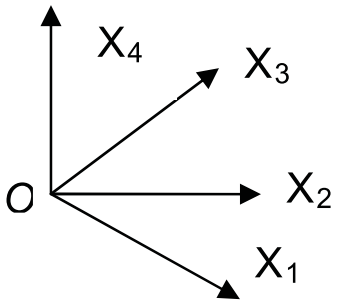


Figure 3 : Distribution of CNVs in a plane at a convex vertex

Consider Figure 3. Consider O as a current pixel. The set of CNVs is $A = \{ \overline{OX_1}, \overline{OX_2}, \overline{OX_3}, \overline{OX_4} \}$.

It can be observed that pixels X_2, X_3 and X_4 are in the same half plane created by the division of the plane by the line OX_1. Hence, the angular span of CNVs is less than π (a convex vertex). Since $S(\overline{OX_1} \times \overline{OX_2}) > 0$, $S(\overline{OX_1} \times \overline{OX_3}) > 0$ and $S(\overline{OX_1} \times \overline{OX_4}) > 0$, hence, $\overline{OX_1}$ will be chosen while following an anticlockwise direction of tracing. Thus X_1 will be chosen as next pixel. It is also observed that since the angles between the vectors is being considered; orientation of $\overline{OX_1}$ can be arbitrary.

Consider that a particular pixel is reached in the tracing process. Let the pixel be designated as current pixel. Let the current pixel be at the center of a 3x3 neighborhood. A vector from current pixel to the pixel from which current pixel is reached is the CPV. Now, compute cross product and dot product between CPV and CNVs. Using the sign of computed cross product and dot product decision for next boundary pixel can be made successfully. This process can be justified as follows:

(a) Let unit CPV be UCPV and unit CNV be UCNV.

(b) For each UCNV, find θ in $(0, 2\pi]$ such that, UCNV = exp (θ J) UCPV.

(c) The UCPV and UCNV can not be equal when a closed boundary is being traced. But, in case the boundary is open, UCPV can be equal to UCNV when the neighborhood of the current pixel has just one region pixel, the previous one. Such a case can be addressed by considering a semi open interval $(0, 2\pi]$ for assigning angles to CNVs.

(d) Choose UCNV with minimum θ to trace an external boundary in anticlockwise direction. The pixels visited starting from CPV till minimum θ is reached are external background pixels. Hence the chosen pixel is an external boundary pixel as it is 4-connected with external background pixel. For further justification refer to Appendix–II. If clockwise direction of tracing is desired, choose minimum angle in clockwise direction.

(e) For 3x3 neighborhood, since θ has discrete values from $\{\pi/4, \pi/2, 3\pi/4, \pi, 5\pi/4, 6\pi/4, 7\pi/4, 2\pi\}$, {+ve, 0, -ve} states of cos (θ) and sin (θ), are adequate for decision making. The correspondence between each of the states and angle in anticlockwise sense is shown in Figure 4 (a). To choose minimum angle, choose the number with minimum value. For finding minimum angle in clockwise sense refer to Figure 4 (b).

(f) {+ve, 0, -ve} for cos (θ) is given by dot product. {+ve, 0, -ve} and for sin (θ) it is given by cross product. Hence, for a 3x3 neighborhood the actual computation of UCNV and UCPV is not required, but just sign of cross and dot products is adequate for choosing the CNV.

(g) Using the above discussion, full range $(0, 2\theta]$ for θ, can be covered. The use of semi-open interval $(0, 2\theta]$ assigns minimum priority for the UCNV in the direction of UCPV.

It can be observed that absolute orientation of CPV is immaterial as relative angle between CPV and CNV is considered. Hence, the logic would give errorless result for all the orientations of CPV and all the orientations of CNVs with respect to CPV. This makes the logic absolutely robust.

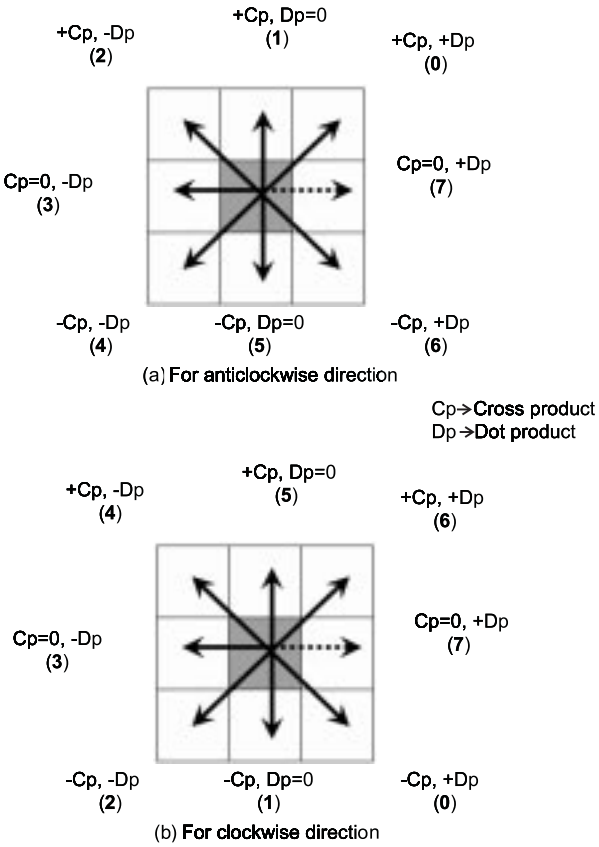


Figure 4 : Priority assignment of pixels using sign of cross product and dot product between CNVs and CPV. The shaded pixel at the center is the current pixel. The solid arrows indicate CNVS and dotted arrow indicates CPV. The numbers indicate the priority assignment, with the lowest number having the highest priority.

In line with the logic given above, the algorithm that can be used for tracing closed external boundaries (TRC_EXT_BNDRY), in a 3x3 neighborhood, is as follows.

1) Locate starting pixel using scanning.
2) Current pixel = starting pixel.
3) Compute CNVs. If set of CNVs is empty, exit. Locate next pixel using only cross product.
4) ECNV = selected CNV.
5) Previous pixel = current pixel. Current pixel = next pixel.
6) Compute CPV and CNVs.
7) Compute CPVxCNVs and CPV.CNVs.
8) In accordance with Figure 4, choose next pixel. The choice of next pixel is done by selecting the sub figure of Figure 4 corresponding to the same direction as that of boundary tracing.
9) Repeat 5 to 8 till current pixel = starting pixel and selected CNV = ECNV.

A. The Proposed Algorithm for tracing a closed internal boundary

The internal boundary, if it exists, will be within the bounds of a closed external boundary. Therefore, the starting pixel of the internal boundary will also be within bounds of the closed external boundary. Also, since the external boundary is closed these will be at least one closed subsection of the internal boundary. For internal boundary to exist there has to be a set of 4-connected background pixels surrounded by region pixels. These background pixels are the internal background pixels. To trace an internal boundary is to find an 8-connected path of those region pixels which are 4-conneted with the internal background pixels.

It is possible that a given image has disjoint clusters of internal background pixels. These may give rise to internal boundaries that are independent. The processing can be done considering one internal boundary at a time. It is also possible that an image may have disjoint regions such that one is placed inside the other. Since the regions are disjoint they are separable, hence the processing should treat it as a multiple object situation and take up one object at a time.

The first step in the tracing is to locate a starting pixel on the internal boundary. The scan procedure to be used needs a modification as the region to be scanned is the one that is contained within the closed external boundary. Also, the scan now looks for the

first internal background pixel. The starting pixel and the next pixel will be derived from the location of the located internal background pixel. The scanning is the process similar to polygon fill. The polygon fill algorithm is give at [16] [17]. A novel algorithm has been derived and is given below. The presented algorithm assumes that the external boundary is traced in an anticlockwise manner.

Algorithm STRT_INT_BNDRY:

1. Find binding rectangle (xmin, ymin) and (xmax, ymax) of the closed external boundary.
2. Found=0. Mark all the pixels in the traced external boundary as "not processed".
3. Loop y_current = ymin to ymax
4. Parity=0.
5. Loop x_current = xmin to xmax
6. If current pixel (x_current, y_current) = region
   a. For all occurrences of current pixel in traced boundary which are marked "not processed" execute:
      i. Starting from (x_current, y_current), traverse the traced boundary pixels in clockwise direction to locate y-coordinate of a previous pixel, y_prev, such that y_prev is not equal to y_current. In the traced boundary, mark the traversed pixels except for the located previous one, as "processed".
      ii. Starting from (x_current, y_current), traverse the traced boundary pixels in anticlockwise direction to locate y-coordinate of a next pixel, y_next, such that y_next is not equal to y_current. In the traced boundary mark the traversed pixels except for the located next one, as "processed".
      iii. If (y_prev - y_current)*(y_next - y_current) < 0 then invert parity.
7. Else (i.e. current pixel = background)
   a. If parity =1 then found=1; break;
8. Loop x_current end.
9. If found break;
10. Loop y_current end
11. If found
    a. starting pixel = (x_current, y_current-1) next pixel = (x_current - 1, y_current)
    b. If clockwise direction of tracing is required for tracing the internal boundary, interchange starting pixel and next pixel.
12. Exit.

Appendix-I gives the supportive justification for the operation of the algorithm STRT_INT_BNDRY. Now the algorithm to trace internal boundary (TRC_INT_BNDRY) is as follows:

1) Trace external boundary using TRC_EXT_BNDRY.
2) Locate starting pixel and next pixel using STRT_INT_BNDRY.
3) Current pixel = starting pixel.
4) ECNV = (next pixel) – (starting pixel).
5) Previous pixel = current pixel. Current pixel = next pixel.
6) Compute CPV and CNVs.
7) Compute CPVxCNVs and CPV.CNVs.
8) In accordance with Figure 4, choose next pixel. The choice of next pixel is done by selecting the sub figure of Figure 4 corresponding to the opposite direction to that of boundary tracing.

Repeat 5 to 8 till current pixel = starting pixel and selected CNV = ECNV.

Figure 5 gives the input image and the traced external and internal boundaries.

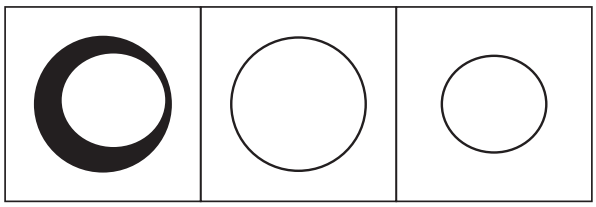


Figure 5 : Results of obtaining external and internal boundary. (a) Input image. (b) Traced external boundary. (c) Traced internal boundary.

IV. TRACING THE "OPEN" BOUNDARY

While tracing a boundary it is not known whether it is “open” or “closed”. This information can be derived only after the boundary is traced. Further, when UCPV = UCNV, the tracing is at dead end and it has to retreat. If in the process of tracing, it is found that the tracing has come to a previously visited pixel and also the next choice of the pixel is same, then that should be taken as the terminating condition. This essentially amounts to checking “Getting same CNV at starting pixel”. The same terminating condition can be used in the case of open or closed boundaries. Further, the choice of next pixel can be made by locating a CNV at minimum angle from the CPV in clockwise or anticlockwise sense irrespective of whether the boundary is open or closed. Hence the algorithm TRC_EXT_BNDRY with some modifications can be used to trace any boundary. After tracing the external boundary, the internal boundary, if it exists, can be located and traced.

Table 1 : First region pixel "O" located by scanning and its neighbors {A, B, C, D}. bk = back ground pixel r=region pixel.

bk	bk	bk
bk	O(r)	A(r, bk)
B(r, bk)	C(r, bk)	D(r, bk)

The modification required in TRC_EXT_BNDRY is limited to addressing a situation when the starting pixel is a dead end of an open boundary. Refer to Table 1. The pixel O is the first region pixel located by the scanning process. Consider 3x3 neighborhood of pixel O. The pixels A, B, C and D can be either region pixels or background pixels. There are four variables each having two states therefore there are 16 possibilities. Following statements discuss all the possibilities:

1. All {A, B, C, D} are background pixels (one possibility). In this case there is nothing further to trace.
2. Only one of {A, B, C, D} is a region pixel (four possibilities).
   a. Current pixel = starting pixel = O. Next pixel = One of {A, B, C, D}.
   b. ECNV= (next pixel) – (starting pixel).
   c. Enter TRC_EXT_BNDRY at statement number 5.
   d. Multiple of {A, B, C, D} are region pixels (eleven possibilities). In this case O is a vertex of an angle. TRC_EXT_BNDRY can be used as it is.

V. RESULTS AND CONCLUSIONS

Vector based boundary tracing is shown to provide a complete solution to an important step in image segmentation and feature extraction process. Novel algorithms for external boundary tracing, locating internal boundary and tracing internal boundary have been presented in this paper. Strong mathematical approach as shown in the present work compared to heuristic approaches can survive and overcome exceptions found in heuristic solutions. Several cases are presented to highlight this issue.

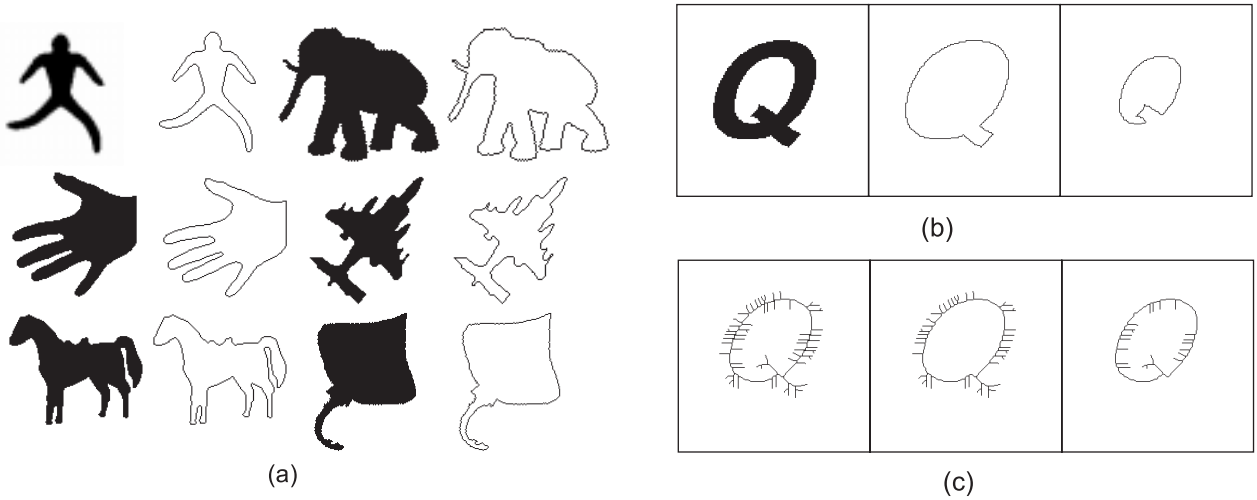


Figure 6 : (a) A few of the test images and the obtained results of boundary tracing. (b) An object with its traced external and internal boundary. (c) Image of morphologically thinned object in (b) and its traced external and internal boundary

More than 1300 images from the database given at reference [18] [19] were subjected to present algorithm for boundary tracing and all the images resulted in reproducible and ordered set of boundary pixels. Figure 6 indicates a few sample results from the image set. Results with these sample images indicate possible boundary tracing applications in multiple problem domains, with regular and irregular shaped objects as well as natural and synthetic images. Appendices prove the robustness of the presented algorithms by giving them the supportive mathematical backing.

APPENDIX-I : JUSTIFICATION FOR STRT_INT_BNDRY.

The justification for: The algorithm (STRT_INT_BNDRY), which locates the first two pixels of the internal boundary and thus the corresponding direction of tracing.

Let:

ibk = internal background pixel.

ebk = external background pixel.

r = region pixel.

Consider Table 2

1. The ebk and ibk type pixels can not be 4-connected with each other, as this will make existence of internal boundary null and void.
2. The pixel O is the background pixel that is inside the closed external boundary. It is also the first one of its kind that has been encountered in the scanning process. The scanning procedure ensures:
   a. There are no ibk type pixels on and above line AB and on ray HJ.

Table 2 : The located internal background pixel O and its neighbors.

L (ebk, r)	M (ebk, r)	N (ebk, r)	P (ebk, r)
K (ebk, r)	A (ebk, r)	B (r)	C (ebk, r)
J (ebk, r)	H (r)	O (ibk)	D (ibk, r)
I (ebk, ibk, r)	G (ebk, ibk, r)	F (ibk, r)	E (ebk, ibk, r)

b. Hence, pixels L, M, N, P, K, A, B, C, J, H are either r or ebk.

c. Pixels on and above line MN and on Ray AK can not be on internal boundary as they can not be 4-connected to ibk. The pixels are shaded gray in Table 2.

3. Further pixels B and H are r. They cannot be ebk as it will 4-connect ibk and ebk, which is not possible. (Statements 1 and 2).

4. Pixels D and F are either r or ibk. They can not be ebk as it will 4-connect ebk and ibk which is not possible (Statement 1). Pixels E, G and I can be r, ebk or ibk.

5. Pixels B and H are 4-connected with ibk and they are region pixels (Statement 3). Hence they are pixels on the internal boundary.

6. Pixels A, K and P cannot be on internal boundary as they can not be 4-connected to ibk. (Statement 2(c)).

7. All the possibilities worked out in above statements, are summarized in Table 2.

8. Using statements 2, 4 and 6, pixel B can be connected only through C, D or H. (As the other pixels in the neighborhood of B, namely A, M, N and P can not be on the internal boundary).

9. Using statement 2, 4 and 6, pixel H can be connected only through B, F, G, I or J. (As the other pixels in the neighborhood of H, namely A, J and K can not be on the internal boundary).

10. Considering all the possibilities posed by statement 8 and 9, and noting ibk should be in the interior of the internal boundary, it can be concluded that B to H is an anti-clockwise direction of tracing and H to B is clockwise direction of tracing.

11. It can be observed that B is a non-convex vertex of the object formed by the region pixels which also implies that the starting point for tracing the internal boundary can not be a dead end of an open boundary.

APPENDIX-II : PROOF FOR 100% SENSITIVITY AND COMPLETENESS.

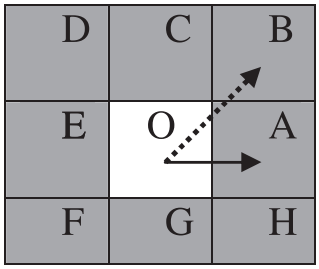
The von Neumann neighborhood of range 'r' is defined by equation (1) [20].

$$N^{v}_{(x_0,y_0)} = \{(x,y) : | x - x_0 | + | y - y_0 | \le r\} \quad (1)$$

The Moore neighborhood of range 'r' is defined by equation (2) [21].

$$N^{M}_{(x_0,y_0)} = \{(x,y) : | x - x_0 | \le r, | y - y_0 | \le r\} \quad (2)$$

Table 3 : The 8 connected pixels of a pixel.



In the discussion below the range "r" in equations (1) and (2) is taken as one, i.e. a 3x3 neighborhood is considered for all the pixels. Refer to Table 3. Pixels {A, B, C, D, E, F, G, H} are all 8-connected to pixel O. Pixels {A, C, E, G} are 4-connected to pixel O. Also pixel A is 4-connected to pixel B, pixel B is 4-connected to pixel C and so on.

With reference to above discussion the following proposition can be proved:

Proposition 1: Given distinct pixels x, y and O such that both the pixels x and y are 8-connected to pixel O, also unit Vector (Ox) = exp ((nπ/4) J) (unit Vector (Oy)), where n = 1 or -1 then pixels x and y are 4-connected.

Table 4 : Sub - image giving possible states of the pixels at the end of scanning and locating first region pixel "G". r = region pixel, ebk = external background pixel, ibk = internal background pixel.

A (ebk)	B (ebk)	C (ebk)	D (ebk)
E (ebk)	F (ebk)	G (r)	H (ebk, r)
I (ebk, r)	K (ebk, r)	L (ebk, ibk, r)	M (ebk, ibk, r)
N (ebk, ibk, r)	O (ebk, ibk, r)	P (ebk, ibk, r)	Q (ebk, ibk, r)

Proposition 2: While tracing the external boundary the proposed algorithm always chooses an external boundary pixel (sensitivity) and all possible boundary pixels are included in the traced boundary (completeness).

Proof:

Assume an anticlockwise direction of seeking minimum angle. Refer to Table 4. The pixel G is located using scanning from top left corner of a binary image. Therefore, G is the starting pixel. The possible states of the pixels are given in Table 4. For a vertex at G, depending on the states of pixels H, K, L and M, the next pixel will be K, L or M. Hence, there are following exhaustive cases:

1. K = r, L = ibk or r, M = ebk or ibk or r, H = ebk or r, F = ebk. Pixel K is chosen as next pixel.
2. K = ebk, L = r, M = ibk or r, H = ebk or r, F = ebk. Pixel L is chosen as next pixel.
3. K = ebk, L = ebk, M = r, H = r. Pixel M is chosen as next pixel.

I. Case 1:

   a. UCPV is in the direction of $\overline{KG}$.

   b. The minimum angle θ is found from the sequence a_seq = {$\pi/4$, $\pi/2$, $3\pi/4$, π, $5\pi/4$, $6\pi/4$, $7\pi/4$, 2π}, such that chosen UCNV = exp (θJ) UCPV. Let the variable in the sequence a_seq, be designated as φ.

   c. The previous selection process has established that the pixel at $\varphi = \pi/4$ is always ebk.

   d. The successive angles in the sequence a_seq have a difference of $\pi/4$. All the pixels pointed to by (exp (φJ) UCPV) are in the 3x3 neighborhood of pixel K and are thus 8-connected with pixel K. Hence by preposition 1, the pixels at successive angles in the sequence a_seq are 4-connected.

   e. Hence, the rejected background pixels are all ebk since they are sequentially 4-connected with pixel at $\varphi = \pi/4$ (statement c).

   f. The region pixel pointed to by the chosen UCNV and the one corresponding to the previous angle in the sequence a_seq (let it be called as pixel LR), are also 4-connected. The pixel LR is ebk (statement e). Hence, the region pixel pointed to by the chosen UCNV is an external boundary pixel.

   g. In the next iteration one of the rejected ebk pixels as elaborated in statement (e) will be at $\varphi = \pi/4$, thus justifying statement (c).

II. Cases 2 and 3 can be proved on similar lines to above.

III. Finally, since the external boundary tracing process is iterative, all the pixels chosen using the criteria in I (b) above are external boundary pixels.

IV. Further implication of statement I (e) is that the rejected ebk pixels form a 4-connected open path (ebk-path) neighboring the current external boundary pixel. This path and a similar path formed around next boundary

pixel, share common pixel(s) (from statements I (g), I (e) and I (c)). Also, each ebk-path has 4-connected boundary pixel as a terminator at both the ends (statements I (f) and I (c)). These statements imply that all ebk pixels that are 4-connected with the current boundary pixel are also members of the ebk-path neighboring the current boundary pixel.

V. Statements at IV imply that if the two distinct boundary pixels are 8-connected then the ebk-paths neighboring them are also connected. The connection is such that the union of the individual ebk-paths is also an ebk-path. Hence, the ebk-paths for head and tail of CNV generate a unified ebk-path.

VI. All the ebk-paths put together will form a closed path of 4-connected ebk pixels (ebk-loop). Since at the terminating condition, the starting pixel and the first chosen boundary pixel form a CNV. Hence when the termination condition is reached, not only the 8-connected path of boundary pixels is traced but also the closed 4-connected path of ebk pixels around the whole region is traced.

VII. Statements IV and VI imply that all possible 4-way connections between the region pixels and the ebk pixels are visited, hence all the boundary pixels are included in the traced boundary.

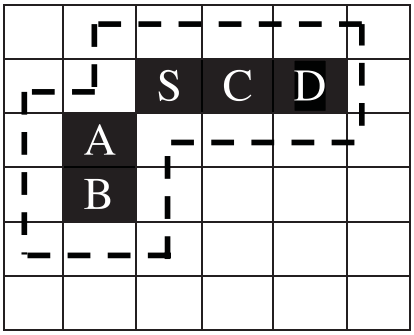


Figure 7 : Image showing the closed 4-connected path of background pixels for the case of open boundary. The path is shown by dotted lines.

A similar proof can be given in the case of open boundaries. Consider Figure 7. Let the direction of tracing be qualified with the words “Forward” and “Reverse” for the open boundary.

1. When the current pixel is B, for CPV at 0 redians the CNV is at 2π radians. Hence the direction of tracing is changed from forward to reverse at B. Also, all the rejected seven background pixels are members of the background pixel path (bk-path). Similar situation exists when pixel D is the current pixel.
2. The open boundaries will have repeated pixels in its traced list.
3. The open boundaries are always one pixel wide. (The place at which a width of a single pixel wide open boundary is increased to more than one pixel, the edge elements in forward and reverse tracing list will differ. Thus generating a closed sub-section).
4. Statement 3 implies that at every selection point there is only one choice which is always a boundary pixel.
5. Consider pixel C. It would be current pixel with CPV as $\overline{CD}$ (forward tracing) and with CPV as $\overline{CS}$ (reverse tracing). The bk-path generated during forward tracing and reverse tracing will together have all the background neighbors of C. In general all the background pixels in the neighborhood of an open boundary pixel are part of the unified bk-path.
6. For the open boundaries the bk-loop will be formed on reaching the termination condition. All possible 4-connected background pixels for each of the boundary pixels will be members of the bk-loop.

Figure 8 shows the ebk-loop with dotted lines, for a region that has an open and a closed sub-section of the boundary.

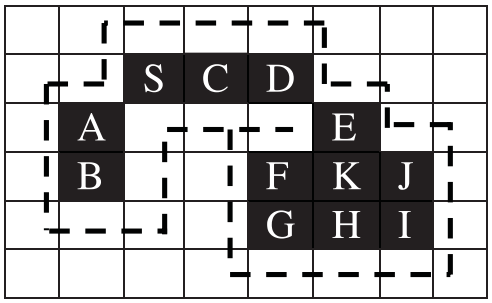


Figure 8 : Image showing the closed 4-connected path of ebk pixels. The path is shown by dotted lines.

ACKNOWLEDGMENT

The authors wish to acknowledge the anonymous reviewers for their constructive criticism and also thank the team members of Embedded Systems Department at I2IT, Pune for their assistance and encouragement.

REFERENCES

1. A. Ashbrook, N.A. Thacker.: Algorithms For 2-Dimensional Object Recognition, Tina Memo No.1996-003, 1998
2. Cosgriff, R. L.: Identification of Shape, Rep. 820-11, ASTIA AD 254792, Ohio State Univ. Research Foundation, Columbus, Ohio USA, 1960
3. Zahn, C. T., Roskies, R. Z.: Fourier Descriptors for Plane Closed Curves, IEEE Transactions on Computers, Vol. 21, pp. 269-281, 1972.
4. Ballard, D.H.: Generalizing the Hough Transform to Detect Arbitrary Shapes, Pattern Recognition 13(2), 111-122.
5. Hough, P. V. C.: Method and Means for Recognizing Complex Patterns, US Patent 3969654, 1962.
6. H. Freeman.: On the Encoding of Arbitrary Geometric Configurations, IRE Trans. Electronic Computers, ED-10, 2, 260-268, June 1961.
7. H. Freeman.: Boundary Encoding and Processing, in Picture Processing and Psychopictorics, B. S. Lipkin and A. Rosenfeld, Eds., Academic Press, New York, 241-266, 1970.
8. Xiaodong Kong, Qingshan Luo, Ying Guo and Guihua Zeng.: A New Boundary Descriptor Based on the Directional Distance Histogram, Communications, 2006. APCC '06. Asia-Pacific Conference on , vol., no., pp.1-4, Aug. 2006
9. A. K. Jain.: Fundamentals of Digital Image Processing, Prentice Hall, Englewood Cliffs, 1989.
10. Rafael C. Gonzalez, Richard E. Woods.: Digital Image Processing, Second Edition, Pearson Education, 2003. pp 66-68.
11. T. Pavlidis.: Algorithms for Graphics and Image Processing, Computer Science Press, Rockville, Maryland, 1982.
12. L. Yingqiang, W Lide.: Improvement of Regional Contour Tracing Algorithm, Pattern Recognition and Artificial Intelligence, 215-226, 1993.
13. MATLAB®, Image Processing Toolbox User's Guide, Version 5.01, 10-14 to 10-18, 2005.
14. W. Kahan.: Cross-Products and Rotations in 2 and 3-Dimensional Euclidean Spaces, Mathematics and Computer Science Depts. University of California Berkeley CA 94720. [Online] Available: www.cs.berkeley.edu/~wkahan/MathH110/Cross.pdf
15. Nitin L Narappanawar, R Athale: Locating a Convex Vertex in Binary Images, ICSCIS07, Jabalpur Engineering college Jabalpur, December, 2007.
16. Donald Hearn and M Pauline Baker.: Computer Graphics C Version, Second edition, Pearson Education Inc. and Dorling Kindersley Publishing Inc., First Impression 2006. pp 137-142.

17. James D Foley et al.: Computer Graphics Principles & Practice, Second edition in C, Pearson Education Inc. and Dorling Kindersley Publishing Inc., First Impression, 2006. pp 116-119.

18. Thomas B. Sebastian, Philip N. Klein and Benjamin B . Kimia.: Shock-based Indexing into Large Shape Databases. ECCV, 2002, [Online] Available:http://www.lems.brown.edu/vision/publications/conference/SebastianECCV02.pdf

19. Sebastian, T.B., P.N. Klein, and B.B. Kimia: Recognition of shapes by editing their shock graphs; T-PAMI May 04 550-571.

20. Weisstein, Eric W.: von Neumann Neighborhood, From MathWorld-A Wolfarm Web Resource, http://mathworld.wolfram.com/vonNeumann Neighborhood.html

21. Weisstein, Eric W.: Moore Neighborhood, From MathWorld - A Wolfarm Web Resource, http://mathworld.wolfram.com/Moore Neighborhood.html

■