Proceedings of International Conference on Image
Processing and Character Recognition

# A Fast Algorithm to Euclidean Distance Transformation by Modeling Ripple Effect

Ti-Chiun Chang, Yung-Nien Sun, and [1]Yuh-Hwan Liu

Institute of Information Engineering National Cheng-Kung University
[1] Department of Electrical Engineering National Cheng-Kung University

**Abstract** — *A novel approach to accurate Euclidean distance transformation (EDT) is presented in this paper. Distance transformation (DT) is an operation of converting a binary image, consisting of object and non-object pixels into a distance map (or image) where the value of each pixel represents the minimum distance from the corresponding pixel to the object surface in the input image. DT has a wide variety of applications in digital image processing. Most of the existing methods of EDT are based on mathematical morphology, which, in nature, is not appropriate to perform the task with enough accuracy. To overcome this difficulty, the rippling model, which simulates the ripple effect in the real world, is proposed. Based on this model, accurate EDT can be achieved efficiently as the corresponding Voronoi diagram of the input image is formed. That is, all the points that have the minimum distances to one of the predefined points are, bounded by the boundary defined by Voronoi diagram, in the same region. The minimum distance at each pixel with respect to the uniquely defined surface points is thus encoded as the exact Euclidean distance. We have shown that the digital Voronoi diagram as well as the Euclidean distance map can be obtained by using the proposed algorithm.*

**Index terms** — Distance transformation, Morphology, Euclidean distance, Voronoi diagram, Digital image processing.

## I. Introduction

In many applications, it is often desirable to know the minimum distance from any given point to the surface of an object in an image. This task can be accomplished by the so-called distance transformation (DT), which generates a distance map (or image) after performing on an input binary image (in our work, the object surface in the binary image is first extracted). The value of each pixel, called distance code, in the derived distance map is the minimum distance from the corresponding pixel to the object surface in the input image. There are some types of distance measure introduced in [1], among them, Euclidean, city block, and chessboard distances are the most commonly used. City block and chessboard distances are based on the 4-neighborhood and 8-neighborhood computation respectively. They are not exact distance, however, are easy to compute and widely developed. In this paper, we focus our attention on Euclidean distance transformation (EDT).

DT has a wide variety of applications in digital image processing[1-2]. One example is the skeletonization for feature extraction and representation[3]. Another popular application is for registration between images, acquired from different imaging modalities, where corresponding point patterns are difficult to extract[4]. The brain images registration is one of such examples, where images of two modalities are first segmented and the brain surfaces are extracted. Then, DT is performed on the surface images of one modality to derive the distance maps. Subsequently, the registration parameters are determined by fitting the other surface image on the distance maps to obtain the minimum surface distance. This technique is called surface fitting and has been widely adopted for registration for multimodality medical images[5, 6]. DT can also be used to obtain the digital Voronoi diagram (or Dirichlet tessellation), which can be extended to more interesting applications. One can easily generate a digital Voronoi diagram by following our proposed algorithm. A large number of algorithms to DTs have been proposed so far. Some of the works are dedicated to city block or chessboard distance measure[7, 8], which are not the main concern in this paper. For EDT, Danielsson[12] proposed sequential algorithms which uses four passes of $3 \times 3$ mask operation over the image. Though the error is quite small, true EDT is still not achieved. Yamada proposed a parallel EDT algorithm that always yields correct results[13]. Unfortunately, parallel operation required an expensive architecture so that not

everyone can afford it. Another popular technique for EDT is the mathematical morphology[1, 2, 7, 9, 10]. But as Frank and Owen claimed in their paper [10], an accurate sequential Euclidean distance transformation algorithm had not yet been presented until their work was published. In their work, mathematical morphology approach is adopted and applied to accomplish the decomposition of the global operation into local operations. However, it is found that the approach based on mathematical morphology is still very complicated and inefficient in deriving the accurate Euclidean distance measure. In this paper, we present a new approach to obtain the true Euclidean distance map efficiently and effectively based on the proposed rippling model which models the ripple phenomenon in the real environment.

The organization of this paper is as follows. In section II, the rippling model, which accounts for why accurate Euclidean distance codes can be obtained and how our idea works in continuous case, is introduced. The digital approximation to the rippling model and the proof of why it remains correct in the digital images are presented in section III. Experimental results and discussion are given in section IV.

## II. Rippling Model

Rippling is the physical phenomenon occurring when the wave propagates from a single point source. The wave propagates outward in the circular shape with center as the source. The developed method is based on modeling ripple effect. In our model, each surface point of the segmented object is considered as a ripple source. Ripple propagation is represented by traversing all the pixels, which construct the wave front surfaces of the ripples, from the source points. From all surface points, the propagation speed is the same in all directions.

Our model can be depicted by Fig. 1. Suppose $s_1$ and $s_2$ are two surface points in an image and are taken as the sources of rippling. At the first time instant, the waves visit all points that are $r_1$ distant away from $s_1$ and $s_2$ due to the same propagating speed. These points constitute two circles centered at $s_1$ and $s_2$ respectively. If the points currently visited are labeled $r_1$, the distance codes of the points are in fact determined (i.e., $r_1$). In continuous case, $\Delta r$ can be chosen arbitrary small so that the distance codes represent the exact Euclidean distances from the surface to these points. Suppose at another time instant, two circles stemmed from $s_1$ and $s_2$ collide at some point $b_1$, which is equally distant away from $s_1$ and $s_2$. In our model, the two ripples will stop propagating right at the meeting point $b_1$ although ripples continue to propagate in the real world. All meeting points generate a line L through which ripples from both $s_1$ and $s_2$ will not pass. Therefore, L becomes the perpendicular bisector of $s_1$ and $s_2$. This is slightly different from the real ripple effect, but is just suitable for our application (i.e., Euclidean distance transformation). The reason is that any point, say $p_1$, lying on the same side of line L as $s_1$ is sure to be closer to $s_1$. This can be easily shown by the geometric relationship shown in Fig. 1. Since propagation speed is assumed the same for all ripples, the precise Euclidean distance code of $p_1$ can be simply determined by the real distance between $s_1$ and $p_1$ (i.e., $|p_1 - s_1|$) without referring to other ripple centers. By this way, we can reduce significant amounts of computation.

In the proposed method, all points in the image will be sequentially visited. The Euclidean distance code (EDC) of any point $p_i$, originated from surface point $s_i$, is determined by

$$EDC(p_i) = | p_i - s_i | \qquad (1)$$

Therefore, the distance map is obtained. The surface image can be segmented into regions that are visited by corresponding surface points. The result is depicted in Fig. 2. Fig. 2 in fact is the well-known Voronoi diagram. We will present the digital approximation to this continuous rippling model in the following section. An algorithm for distance transformation will be given also.

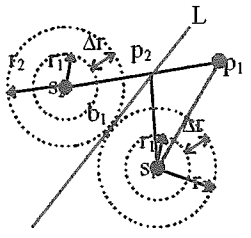


Fig. 1 Two ripple sources with s1 and s2 being the ripple centers. Line L is the perpendicular bisector, so $|p_2 - s_2| = |p_2 - s_1|$. By triangle inequality, $|p_2 - s_1| + |p_2 - p_1| > |p_1 - s_1|$. Therefore, $|p_1 - s_2| = |p_1 - p_2| + |p_2 - s_1| > |p_1 - s_1|$.

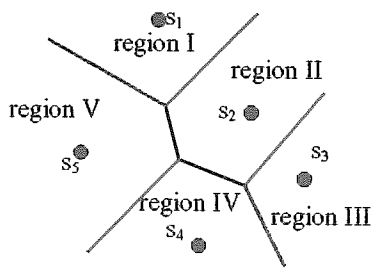


Fig. 2 Voronoi diagram resulted from our rippling model. All points in region I are visited by wave originated from $s_1$, all points in region II are visited by wave originated from $s_2$, and so on.

## III. Digital Approximation to Ripple Effect

In this section, we will introduce the digital approximation of the proposed ripple model. At first, the required data structures are defined in subsection A. In subsection B, some problems for digital ripples are presented. Then, we describe how the proposed algorithm works, and how digital ripples are generated to approximate the continuous ones in 2-D space in subsection C. Explicit algorithm using the proposed data structures is also given here. In subsection D, it is shown that the digital approximation to ripple effect meets our requirement. Extension to 3-D case is described in Subsection E.

### A. Data Structure

In order to derive the exact Euclidean distance map of a given image, we need three extra data structures to record some information when modeling ripple effect in digital images.

The first data structure includes the visiting pixel list (VPL), and the next visiting pixel list (NextVPL). VPL is used to record the coordinates of all pixels that are currently visited and NextVPL is for the pixels that will be visited at the next iteration. This data structure can be implemented as a linked list since the number of visited pixels varies at different time instants and for different images. The second one, called the source map (SM), is used for every pixel in the given image to record the corresponding surface point from which the point is originated. In other words, no other ripples generated from other surface points visit the specific pixel earlier than the ripple originated from the one recorded in its SM. As mentioned in the previous section, the exact Euclidean distance code can be obtained by using Eq. (1) based on the source points recorded in SM. The third data structure, called the minimum flag (MF), is used to mark whether the accurate Euclidean distance code of the corresponding pixel is determined. With this flag, quite some unnecessary computation can be avoided.

### B. Characteristics of Digital Ripples

Although the digital ripple is designed to simulate the continuous ripple, the following characteristics are still different in these two cases.

First, continuous ripples can expand by arbitrary step size in all directions. Thus, real circles can always be generated in any time instant. However, in digital case, circles can only be approximated but not exactly formed. To overcome this problem, digital ripples are allowed to expand by the range of a given step size, within which the pixels away from the surface points are visited at the same time instant, instead of the sequential visiting of all pixels in the continuous case. Note that the same time instant here, is actually a time interval since the proposed algorithm is sequential. For consistency with the real ripple effect, the term "time instant" is used in lieu of "time interval" hereafter.

The second major difference occurs when two digital ripple collide. Tow continuous ripples always meet at a pixel, from which the two ripple centers are with the same distance. However, digital ripples may expand and meet at a point with different distance codes. Since the accurate Euclidean distance code is desired, the minimum radius among the possible distance codes is chosen for the visited pixel. This will be illustrated in more details in the next subsection.

### C. Algorithm

At the first stage in this algorithm, the given image is scanned. Meanwhile, the source map (SM), distance code, and minimum flag (MF) at every surface point are initialized to be the current position, 0, and 1 respectively. The linked list VPL is then formed and used to record the coordinates of the surface points subsequently.

The next stage is the digital approximation to the expansion of continuous ripple. The coordinates of visited pointed are recorded in the VPL. The subsequent points which will be visited at the next time instant can be obtained by searching the 8-connected neighborhood of the points recorded in the VPL.

During the searching process, only the neighborhoods within the allowed distance are recorded in a new linked list, called the NextVPL. To be more specific, assuming that at the current time instant, the pixels being visited are shorter than $r_d$ away from the surface points, then at the next time instant, all pixels which are longer than or equal to $r_d$ but shorter than $r_d+1$ away from surface points should be visited. For instance, the step size is 1 pixel and the points with Euclidean distance shorter than 2 pixels away from surface points are visited at the first time instant, the points with distance shorter than 3 pixels but longer than or equal to 2 pixels away from the surface points are visited at the second time instant, and so on. Hence digital ripples are formed at each time instant. SM and distance codes at the visited points are set according to the SM at those neighboring points visited at the last searching process. There may be more than one neighboring points (candidates) which can be used to set the SM and distance codes. The minimum radius among the

candidates is chosen to be the correct distance code for the visited pixel. The SM of this visited pixel are set accordingly to record the surface point which expands to the visited point with the minimum digital radius. This is achieved by delaying the setting of MFs at the visited pixels until the end of each iteration but not at the instant when pixels are visited. This can be clearly seen in our algorithm.

When the searching process is finished, the VPL (current linked list) can be dropped and the NextVPL, which is formed during the searching process is adopted to replace it. This completes a visiting iteration, and the next iteration can then be resumed. The iterations continue until all pixels in the given image are visited. An example when expanding process at the third time instant is shown in Fig. 3.

```
/* Pseudo-code for Ripple Algorithm */
Allocate memory for SM, MF, and EDM (Euclidean Distance Map);
/* First stage */
Scan given image and set DM, MF and SM at the surface points to be 0, 1, and coordinates of surface points
respectively;
Form a VPL consisted of surface points;
/* Second stage */
r = 1;
do
  r++;    /* The allowed distance range is 2 pixels for the first iteration and the step size is 1 */
  do
    For each of the 8-connected neighborhood points of the visited pixel (8-CNP)
      if (MF at 8-CNP is false) then
        square of ED = square of distance from SM(at visited pixel) to 8-CNP;
        /* ED stands for Euclidean distance. Square root is not necessary here */
        if (square of ED < r*r) then
          if (first visit to 8-CNP is true) then
            SM(at 8-CNP) = SM(at visited pixel);
            EDM(at 8-CNP) = ED; /* Square root operation is needed here */
            Add one more node which records the coordinate of 8-CNP to NextVPL;
          else        /* two or more ripples collide */
            EDM(at 8-CNP) = min{ED, EDM(at 8-CNP);
    Get the next node from VPL and delete the used node;
  until (VPL == NULL)
  VPL = NextVPL;
  set MF(at coordinates recorded in VPL) to be true;
until all pixels in the image are visited.
```

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $^{(1)}1$ | × | $^{(1)}1$ | $^{(2)}2$ | $^{(3)}3$ | -1 | -1 | -1 |
| $^{(1)}\sqrt{2}$ | $^{(1)}1$ | $^{(1)}\sqrt{2}$ | $^{(2)}\sqrt{5}$ | $^{(3)}\sqrt{10}$ | -1 | -1 | -1 |
| $^{(2)}\sqrt{5}$ | $^{(2)}2$ | $^{(2)}\sqrt{5}$ | $^{(2)}\sqrt{8}$ | $^{(3)}\sqrt{13}^{*}$ | -1 | -1 | -1 |
| $^{(3)}\sqrt{10}$ | $^{(3)}3$ | $^{(3)}\sqrt{10}^{*}$ | $^{(3)}\sqrt{10}^{*}$ | $^{(3)}3$ | $^{(3)}\sqrt{10}$ | $^{(3)}\sqrt{13}$ | -1 |
| -1 | $^{(3)}\sqrt{13}^{*}$ | $^{(2)}\sqrt{8}$ | $^{(2)}\sqrt{5}$ | $^{(2)}2$ | $^{(2)}\sqrt{5}$ | $^{(2)}\sqrt{8}$ | $^{(3)}\sqrt{13}$ |
| -1 | $^{(3)}\sqrt{10}$ | $^{(2)}\sqrt{5}$ | $^{(1)}\sqrt{2}$ | $^{(1)}1$ | $^{(1)}\sqrt{2}$ | $^{(2)}\sqrt{5}$ | $^{(3)}\sqrt{10}$ |
| -1 | $^{(3)}3$ | $^{(2)}2$ | $^{(1)}1$ | × | $^{(1)}1$ | $^{(2)}2$ | $^{(3)}3$ |
| -1 | $^{(3)}\sqrt{10}$ | $^{(2)}\sqrt{5}$ | $^{(1)}\sqrt{2}$ | $^{(1)}1$ | $^{(1)}\sqrt{2}$ | $^{(2)}\sqrt{8}$ | $^{(3)}\sqrt{10}$ |

Fig. 3 Example of the distance map at the 3rd time instant. × denotes the surface points and the pixels with values -1 are not visited yet. $^{(i)}$ denotes at which time instant the pixel is visited and $^{*}$ denotes the colliding pixels.

## D. Proof of Accuracy

We now show why Euclidean distance codes can be obtained accurately by using the following mathematical induction: consider at the first time instant, all pixels with distance codes smaller than 2 are visited, it can be easily shown that all these points are assigned with accurate Euclidean distance codes by tracing our algorithm. We then assume at time instant $t_k$, all pixels with distance codes smaller than k+1 are assigned with accurate distance codes. Now let us consider at time instant $t_{k+1}$. We want to make sure that all pixels visited at this time instant are also assigned with accurate Euclidean distance codes. The accurate Euclidean distance codes of the visited pixels must be within [k+1, k+2) at the (k+1)th time instant. Since problems may arise only at the colliding pixels and the distance codes of these pixels can be uniquely determined as the minimum of all competing distance codes in our algorithm, the accurate Euclidean distance codes, which represents the minimum distance from the pixel to all surface points, can also be derived at the time instant $t_{k+1}$. By mathematical induction, all points in the surface image can be assigned with accurate Euclidean distance codes and hence the correctness of the proposed algorithm is proved.

## E. Extension to 3-D

The extension to 3-dimensional surface distance transformation is straightforward from the 2-D algorithm. The only difference in 3-D case is that ripples expand in the shape of a sphere instead of a circle, so that total of 26 neighborhood voxels must be searched to check whether the Euclidean distance from the ripple centers lie in the range [r, r+1) or not, while not the 8-connected neighborhood pixels employed in the 2-D images. It can be easily shown that the sets of colliding points constitute planes of perpendicular bisector.

## IV. Result and Discussion

To demonstrate the correctness of our algorithm, an image of 2-D transverse brain surface and the obtained distance map are shown in Fig. 4 as an example. In the following, we show why the proposed method is superior to the others.

Mathematical morphology for digital image processing, in essence, is not appropriate to approximate Euclidean distance metric in the continuous domain. For instance, the sequential distance encoding process, starting from the given seed points, based on the morphological operations can not easily achieve accurate distance codes[10] even with various of complex structuring elements.

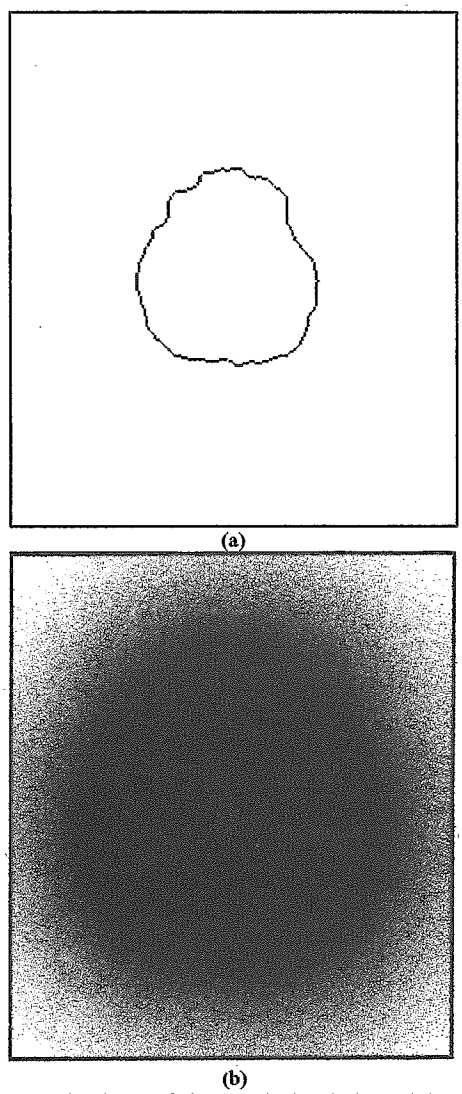


Fig. 4 (a) Surface image of transverse brain. The image is inverted. (b) The corresponding Euclidean distance map (the distance values are multiplied by 2).

To obtain the accurate Euclidean distance codes for all the pixels in a digital image, gray scale morphological erosions/dilations are used with a mask which must be at least as large as the largest object in the input image. The elements of the mask must also be well-predefined. This is essentially a global operation. Since all global operations are prohibitively costly, algorithms that consider only a small neighborhood at a time are introduced to simplify the computation while still giving a reasonable approximation to the Euclidean distance. The four-point and eight-point sequential Euclidean distance mapping algorithms (4SED and 8SED) were proposed by Danielsson[12]. These algorithms are based on a two-component descriptor and a two picture scans. The errors as compared to Euclidean distance are within 0.29 pixel units for 4SED and 0.09 pixels units for 8SED at some points. Borgefors[11] used a four-pass algorithm to obtain the exact Euclidean distances at most pixels but there

are still some exceptions at special feature pixels. Via combining the city block and chessboard distance transformations to approximate the EDT, Frank obtained the accurate Euclidean distance codes except when the closest point on the boundary for the Euclidean distance is not located at the same position as that for the city block or for the chessboard distances[10]. If the accurate Euclidean distance codes are to be obtained for all the pixels in the input image, further decomposition of the structuring element into smaller one is required in order to reduce the computation. Then, total of n (n + 1) / 2 gray scale erosions are needed for an image size of $(2n + 1)\times(2n + 1)$ if the method in[10] is adopted. For large image size, it is still too costly in computation. Raster scan sequential algorithms for computing EDT were implemented by Leymarie and Levine [14]. They showed that those algorithms have computational complexities comparable to the city block, chessboard, and other simple chamfer DTs. These algorithms are optimal in numerical computation but exact Euclidean distance codes for all the pixels in a digital image still cannot be obtained.

The proposed method provides a more efficient approach and has the following major advantages:

1) Accurate Euclidean distance codes can be obtained for all the pixels in the input image: even at the non-integer coordinates, true accurate Euclidean distance codes can still be derived since the ripple center of the non-integer coordinates can be known from their integer-coordinate neighborhoods.
2) Highly efficient: suppose the image size is n $\times$ n. It can be easily proved that the computation complexity for our algorithm is only $O(n^2)$, which is just the same degree with the size of the given image. The only redundant computation is on the perpendicular bisectors.
3) The idea is very simple and can be easily implemented.

The only costs that we have to pay are the additional memory spaces allocated for the mask of MF and SM, and the VPL. This payment is almost trivial as the easy access of memory with the computer today.

## V. Conclusion

Euclidean distance transformation is in essence the problem of finding the minimum distances among various points. The idea of Voronoi diagram is a good solution to this problem. The proposed algorithm based on modeling the ripple effect is intuitive. More importantly, the proposed method is one of the few works that can obtain accurate Euclidean distance codes. In addition, our approach is superior to the others in that it is computationally efficient and can be easily implemented. Almost only one visit to each pixel is needed (except at the colliding pixels) in the encoding process. Instead, intensive computation is required by using mathematical morphology as every pixel has to be visited repeatedly. The experimental results reveal that the proposed method is indeed precise and efficient. It has also been well applied to the multi-modality image fusion in brain studies. Furthermore, parallelization of the proposed algorithm is the theme of future research.

## References

1. G. Borgefors, "Distance Transformations in Digital Images," *Comput. Vision, Graphics, Image Proc.*, Vol. 34, pp. 344-371, 1986.
2. Valery Starovoitov, "Distance-morphological Transformation of Digital Images," *Proc. of SPIE Vol. 2488*, pp. 425 - 430.
3. Ron Kimmel, Doron Shaked, Nahum Kiryati, "Skeletonization via Distance Maps and Level Sets," *Computer Vision and Image Understanding*, Vol. 62, No. 3, pp. 382-391, Nov. 1995.
4. G. T. Herman, J. Zheng, and C. A. Bucholtz, "Shape-based Interpolation," *IEEE Computer Graphics & Applications*, pp. 69-79, May 1992.
5. Hongjian Jiang, Richard A. Robb, Kerrie S. Holton, "A New Approach to 3-D Registration of Multimodality Medical Images by Surface Matching," *Proc. of SPIE Vol. 1808, Visualization in Biomedical Computing*, pp. 196-213, 1992.
6. Stephane Lavallee, Richard Szeliski, Lionel Brunie, "Matching 3-D Smooth Surfaces with their 2-D Projections using 3-D Distance Maps," *Proc. of SPIE Vol. 1570, Geometric Methods in Computer Vision*, pp. 322-336, 1991.
7. A. Rosenfeld and J. L. Pfaltz, "Digital Picture Processing," New York: Academic, 1982.
8. P. A. Maragos and R. W. Schafer, "Morphological Skeleton Representation and Coding of Binary Images," *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. 34, pp. 1228-1244, Oct. 1986.
9. A. M. Vossepoel, "A Note on Distance

Transformations in Digital Images," *Comput. Vision, Graphics, Image Proc.*, Vol. 43, pp. 88-97, 1988.

10. Frank Yeong-Chyang Shih, Owen Robert Mitchell, "A Mathematical Morphology Approach to Euclidean Distance Transformation," *IEEE Trans. Image Processing*, Vol. 1, No. 2, pp. 197-204, Apr. 1992.
11. G. Borgefors, "Distance Transformations in Arbitrary Dimensions," *Comput. Vision, Graphics, Image Proc.*, Vol. 27, pp. 321-345, 1984.
12. Per-Erik Danielsson, "Euclidean Distance Mapping," *Computer Graphics and Image Processing*, 14, pp. 227-248, 1980.
13. H. Yamada, "Complete Euclidean Distance Transformation by Parallel operation," in *Proc. 7th Int. Conf. on Pattern Recognit. Montreal*, Canada, pp. 69-71, 1984.
14. F. Leymarie and M. D. Levine, "Fast Raster Scan Distance Propagation on the Discrete Rectangular Lattice," *CVGIP: Image Understanding*, Vol. 55, No. 1, pp. 84-94, Jan. 1992.