Fast Distance Transform Computation using Dual Scan Line Propagation

Fatih Porikli Tekin Kocak

Mitsubishi Electric Research Laboratories, Cambridge, USA

ABSTRACT

We present two fast algorithms that approximate the distance transformation of 2D binary images. Distance transformation finds the minimum distances of all data points from a set of given object points, however, such an exhaustive search for the minimum distances is infeasible in larger data spaces. Unlike the conventional approaches, we extract the minimum distances with no explicit distance computation by using either multi-directional dual scan line propagation or wave propagation methods. We iteratively move on a scan line in opposite directions and assign an incremental counter to underlying data points while checking for object points. To our advantage, the precision of dual scan propagation method can be set according to the available computational power. Alternatively, we start a wavefront from object points and propagate it outward at each step while assigning the number of steps taken as the minimum distance. Unlike the most existing approaches, the computational load of our algorithm does not depend on the number of object points either.

Keywords: Distance transform, wave propagation, shape comparison, morphology

1. INTRODUCTION

Distance transform[1] is a mapping that computes the minimum distances of all data points from a given set of object points. It has been extensively studied in computational geometry, image processing, computer graphics and pattern recognition. It is essential in a wide-spectrum of tasks ranging from geospatial map generation to deblurring to object retrieval. Finding distances plays a central role in comparison of binary images, particularly for images resulting from local features such as edge or corner detection. It is also used to obtain the medial axes of digital shapes.

There are various ways of applying the distance transform depending on which distance metric is being used and how the local distance information is propagated. The distance metrics include the L_1, L_2, L_∞, chamber,[5] etc., with the L_2 (Euclidean) being the most common. Chamfer metrics result in an approximation to the circular form, usually as an 8-sided or 16-sided closest fit polygon.[8] There are other even slower distance metrics that aim to make the distance computation robust to noise. In Gaussian distance transform, the value at a pixel is set proportional to the weighted sum of distances of that pixel to all object pixels, with the weights inversely proportional to the distances. Gaussians are used as the weights. The summation has an averaging effect and reduces the effect of noise. As

*fatih@merl.com, phone: 1.617.621.7586

the standard deviation of the Gaussian is increased, the effect of noise decreases more, but that at the same time reduces local shape details.

Since the exact Euclidean distance transform is computationally intensive, several algorithms have been proposed that use some mask which is swept over the image in two scans, e.g. as proposed by Rosenfeld,[1] to compute approximations like the city-block, chess-board, or chamfer distances. One intuitive but extremely inefficient way of applying distance transform is to perform multiple successive morphological erosion operators with a suitable structuring element until all foreground regions of the image have been eroded away. If each pixel is labeled with the number of erosions that had to be performed before it disappeared, then this is just the distance transform. The shape of the structuring element dictates the type of the distance metric. A square element results in the chess-board distance transform, a cross shaped element gives the city-block distance transform, and a disc shaped element approximates the Euclidean distance transform. The time complexity of these approaches is linear in the number of pixels of the image but it does not yield the exact Euclidean distance, which is required for some applications. Another drawback of these algorithms is that they are hard to parallelize for parallel computers since previously computed results are propagated during the computation. Zampriolli and Lofuto[2] present survey of efficient 2D morphological erosion algorithms that decompose the structuring element and take advantage of parallel rastering.

Paglieroni[3] proposed a unified algorithm that computes distance and related nearest feature transform concurrently for arbitrary bit maps. This algorithm has an efficient implementation on serial processors based on parallel row followed by parallel column scanning. Felzenszwalb *et. al.*[7] provided a linear-time algorithm for solving a class of minimization problems involving a cost function with both local and spatial terms. These problems can be viewed as a generalization of classical distance transforms of binary images, where the binary image is replaced by an arbitrary sampled function. Kolountzakis[10] developed an algorithm requires column-wise half plane search recursively followed by row-wise search to accelerate the Euclidean distance computation. Lee *et. al.*[9] implemented the chess-board distance on parallel processors, then extented this work for Euclidean distance computation[11] where they derive geometry relations and properties among parallel planes and adapt a parallel algorithm on the emulated EREW PRAM. Breu *et. al.*[4] presented asymptotically optimal algorithms for computing the Euclidean distance transform of a two-dimensional binary image. Their methods are based on the construction and regular sampling of the Voronoi diagram whose sites consist of the unit pixels in the image. They construct the Voronoi diagram where it intersects the horizontal lines passing through the image pixel centers. Their methods require only linear time. On the other hand, this explicit Voronoi diagram construction requires extra space of size proportional to the input image.

2. DUAL SCAN LINE PROPAGATION

We give an algorithm for 2D image data in Cartesian space, however, it is straightforward to extend the formulation for higher dimensional bounded Cartesian spaces.

Let $\mathcal{P}$ be a regular grid corresponding to the image I, p is a point in the set of background points $\mathcal{P}$, and q is a point in the set of object points $\mathcal{Q}$. Assume $\mathcal{P}$ has M points and $\mathcal{Q}$ has N points. For a binary image, the background and object points have different intensity values, i.e. $I(p) = 0$ and

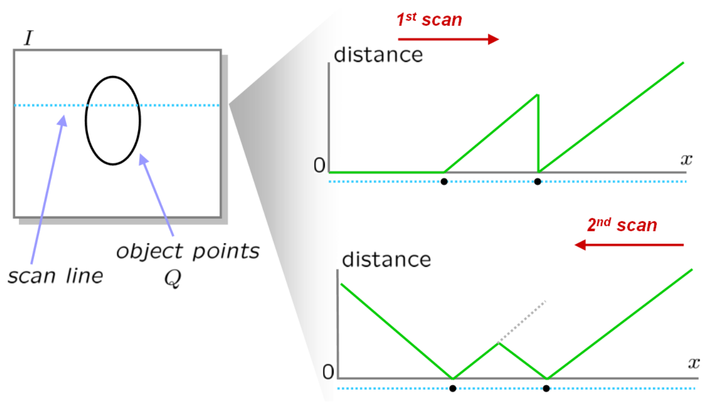

Figure 1. Back-and-forth scanning on one direction.

$I(q) = 1$, and their union constitutes the image. The distance transform f where $f : \mathcal{G} \rightarrow \mathbb{R}$ assigns each background point to the distance from the nearest object point as

$$f(p) = \min_{q \in \mathcal{Q}} d(p, q) \tag{1}$$

where $d(p, q)$ is a measure of the distance between the background point p and object point q. In case of using Euclidean distance, the metric is defined as

$$d(p, q) = ||p, q|| = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2}. \tag{2}$$

To integrate local weight constraints, some methods use the following alternative definition

$$f(p) = \min_{q \in \mathcal{Q}} (d(p, q) + m(q)) \tag{3}$$

where $m(q)$ is a membership function. In other words, this formulation finds a point q that is close to p and also has a small $m(q)$. A *nearest feature transform* of I that assigns each point to the nearest point such as

$$n(p) = \arg\min_{q \in \mathcal{Q}} d(p, q) \tag{4}$$

can also be easily obtained after the distance transform computation.

Instead of computing and finding minimum distances of all M data points from all N object points (a total of MN distance computations and MN comparisons), we repeat simple increment operations and do not explicitly compute any distance.

We take advantage of the fact that for two neighboring data points p_i, p_{i+1} the associated minimum distances should either $f(p_i) < f(p_{i+1})$ or $f(p_i) > f(p_{i+1})$ on a line passing through three points; an object point, and data points p_i, p_{i+1}. Note that the distances cannot be equal either. In other words, if we start from an object point and keep moving on a straight line, we get a larger minimum distance at each step. Thus, it is possible to move on a line originated from an object point, increment a counter, and assign the counter value to the current point on the line as the minimum distance. What

Algorithm 1 First Scan

```
origin flag = no;
for $i = i + 1$ do
    if $I(p_i) = 1$ then
        origin flag = yes;
        counter = 0;
    else
        if origin flag then
            counter = counter + 1;
            $f(p_i)$ = counter;
        end if
    end if
end for
```

would happen if we move on a line but we do not have an object point to start the counter? Since our conjecture needs an origin point, we cannot impose it any longer. Therefore, we move but we do not make any changes. The algorithm for the first scan is given in algorithm 1.

The above algorithm is illustrated fig. 1. As visible, the first scan can only find the minimum distances from the object points on the *left* of the data points. However, a data point may be more closer to an object point on its *right*. Consequently, we repeat the scan as above, in this case, in the opposite direction. One difference from the previous propagation is that some data points may already have minimum distances assigned after the first scan. Thus, we compare the new counter value with the assigned distance (if any) and choose the minimum of these two as the new minimum distance of the current point, as given in algorithm 2.

We process the input image line by line by applying the above dual scan propagation. Note that

Algorithm 2 Second Scan

```
origin flag = no;
for $i = i - 1$ do
    if $I(p_i) = 1$ then
        origin flag = yes;
        counter = 0;
    else
        if origin flag then
            counter = counter + 1;
            if $f(p_i)$ exists then
                $f(p_i)$ =$\min$(counter,$f(p_i)$);
            end if
        end if
    end if
end for
```

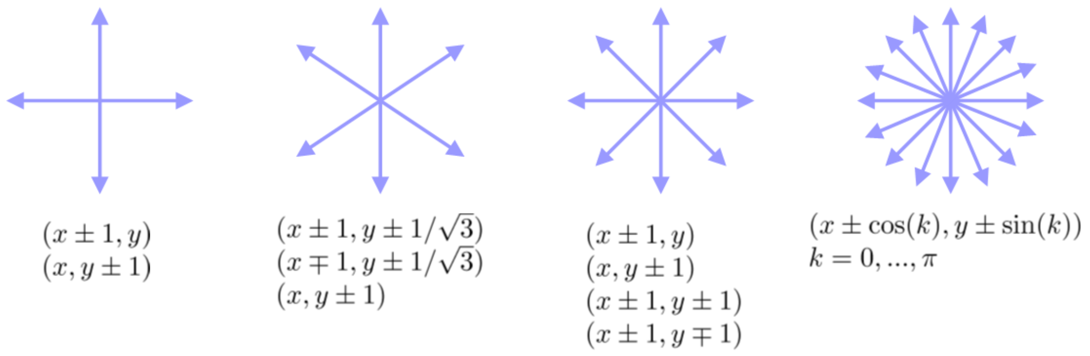

Figure 2. Multiple scanning directions. Either the scan direction is changed or the data space is rotated.

scanning each line independently guarantees only the extraction of the minimum distances normal to the object data point surface. Consider a convex or planar surface and a normal line at a certain object point. It is easy to visualize that the minimum distances of the data points on the normal line correspond to the distances from the object point. All the other object points will be further away due to the triangle inequality. Hence, each different normal direction exists on the object surface requires an additional dual-scan propagation in that direction. Since the object surface may be concave (or convex with multiple normal directions), we apply the dual scan propagation in multiple directions as illustrated in fig. 2. For the scans in multiple directions, we use the previous distance map as a prior.

In case the number of the dual scans is less than the number of the normal directions on the surface, e.g. due to the sharp edges on the surface or singleton data points, approximation error happens. Such errors become more apparent as the size of the data space becomes larger, as a result the statistical average of the minimum distance score increases. Still, in practice, there are limited number of normal directions due to the rasterization of the surface and the confined size of images. We emphasize that each additional direction refines the estimated distance transformation values. Being able to improve the quality of the estimations with respect to the available computational power is essential in computationally constrained applications.

Unlike the most conventional approaches, the dual scan propagation in multiple direction is not adversely effected by the increasing number of the object points. As the number of object points becomes greater, i.e. their density in the data space grows, the process becomes more accurate since the average minimum distance statistically expected to decrease. This is not the case for the algorithms that propagate local minima or compute distances from all data points.

Another important advantage of the proposed method is that, scanning can be executed for multiple parallel lines independent from each other for a single direction. In other words, the dual scan line propagation can be easily implemented using parallel programming.

Since the propagation results always update the same distance transform map, the dual scan method does not require any additional memory space as well.

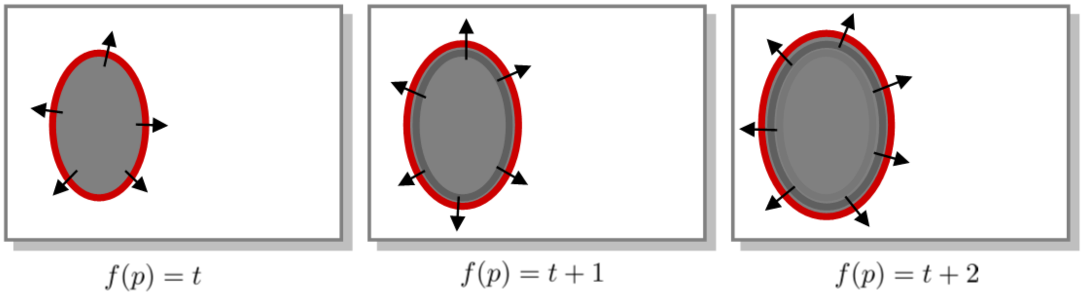

Figure 3. Fast wave-propagation method.

3. WAVE-PROPAGATION

Alternatively, we can start the lines from each object point and move in a direction normal to the surface. Usually, such an approach computes surface curvature and requires sub-pixel accuracy. A well-known surface propagation techniques is fast marching of Eikonal equations as described in Porikli.[12]

Here, we propose a much simpler implementation that does not depend on surface information. We use three separate labels, processed, active, and unprocessed, to group the data points at each step. We find the object boundary and assign them as active points and setting distance to zero, $f(p) = t = 0$. The points that we do not need to compute the minimum distances, e.g. the points inside the object surface, etc, are assigned as the processed points. Then, we start propagating the wavefront using the active set of points until no data points remains in the active set. At each step, we update a counter $t \leftarrow t+1$ and set it as the distances of all points in the active set $f(p) = t$. We search for the immediate unprocessed neighbors of the points that are in the active set, and construct a new set of active points from those points. After we update the old active set points as processed, we iterate the nest step. This is illustrated in fig. 3.

Similar to dual line scan, this algorithm is very fast and effectively approximates the city-block or chess-board distances depending on the connectivity rule in the update of the active set. Using 4-neighbor connectivity gives the city-block distance and using 8-neighborhood imposes the chess-board distance. In addition, the computations are in linear time and does not depend on the number of object points.

4. RESULTS & DISCUSSION

We compared the computational load of the our methods with the widely accepted distance computation proposed by Breu[4] and level-set based propagation.[12] Breu's algorithm has already been adapted

Breu[4]	Level-sets[12]	Dual-scan	Wave-front
160 msec	1200 msec	100 msec	30 msec

Table 1. Average computation times for 512×512 images.

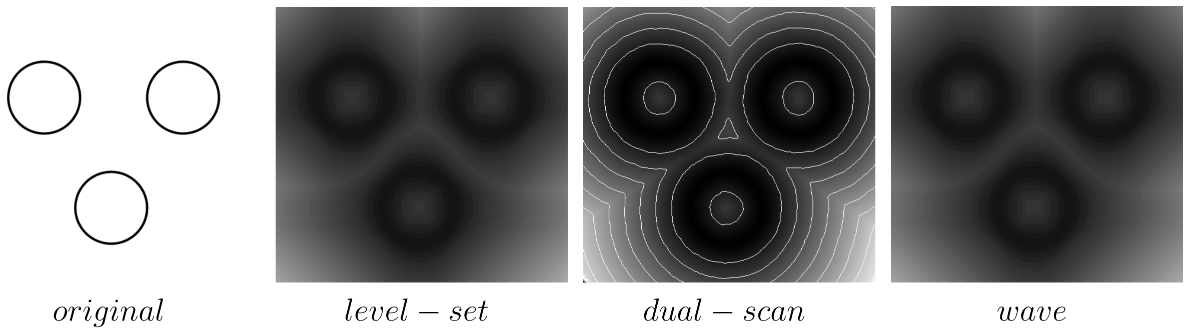

Figure 4. Estimated distance maps of for a sample image. Level set and wave propagation based methods use 4-neighborhood, thus they approximate the city-block distance. The dual-scan propagation method applied for 12 directions. To show the estimation errors, we plotted the isocontours.

in the latest version of MATLAB software. We tested several binary images with changing number of object points. The average computation times are shown in Table 3.

We observed that both of the proposed methods reduces the average computation times, almost to one third of the current state-of-art implementation for distance transform. As visible in Fig. 4, the dual-scan propagation approximates the Euclidean distance. The accuracy improves as the resolution of the scanning directions increases, which is also linearly reflected to the computational load. We observed insignificant difference for 512×512 images using 12 to 24 orientations.

The wave-front propagation is the fastest method we tested. Using the 4-connectivity in active set update imposes the city-block distance as visible in Fig. 4. The chess-board distance is obtained when the connectivity constraint is enlarged to the nearest 8 neighboring points.

As we explained before, the dual-scan propagation can be effectively implemented using parallel programming unlike the sequential wave-front propagation. Besides, the wave-front method requires a mask of labels, which doubles the memory imprint. Since the propagation results always update the same distance map, the dual scan method does not require any additional memory space.

REFERENCES

1. A. Rosenfeld, J. Pfaltz, “Distance Functions in Digital Pictures”, Pattern Recognition, Vol 1, 1968, pp 33-61.
2. F. Zampirolli, R. Lotufo, “Classification of the Distance Transformation Algorithms under the Mathematical Morphology Approach”, XIII Brizilian Symposium on Computer Graphics and Image Processing, 2000.
3. D. Paglieroni, “A unified distance transform algorithm and architecture”, Journal Machine Vision and Applications Volume 5, Number 1, December, 1992.
4. H. Breu, J. Gil, D. Kirkpatrick, M. Werman, “Linear Time Euclidean Distance Algorithms,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 17, no. 5, pp. 529-533, May, 1995.
5. R. Haralick, L. Shapiro, “Computer and Robot Vision”, Vol. 1, Addison-Wesley Publishing Company, Chapter 5, 1992.
6. L. Chen, H. Chuang, “A fast algorithm for Euclidean distance maps of a 2-D binary image”, Information Processing Letters, 5 1:25-29, 1994.

7. P. Felzenszwalb and D. Huttenlocher, “Distance Transforms of Sampled Functions”, Cornell Computing and Information Science, Techical Report, 2004.
8. G. Borgefors, “Hierarchical chamfer matching: A parametric edge matching algorithm”, IEEE Transactions on Pattern Analysis and Machine Intelligence, 10(6):849865, 1988.
9. Y. H. Lee , S. J. Horng, “Optimal computing the chessboard distance transform on parallel processing systems”, Computer Vision and Image Understanding, v.73 n.3, p.374-390, March 1999
10. M. Kolountzakis , K. Kutulakos, “Fast computation of the Euclidian distance maps for binary images”, Information Processing Letters, v.43 n.4, p.181-184, Sept. 28, 1992
11. Y. H. Lee, S. J. Horng, J. Seitzer, “Parallel Computation of the Euclidean Distance Transform on a Three-Dimensional Image Array”, IEEE Transactions on Parallel and Distributed Systems archive, 14:3, 203-212, March 2003
12. F. Porikli, “Automatic image segmentation by Wave Propagation”, Proceedings of IS&T/SPIE Symposium on Electronic Imaging, San Jose, 2004