An Algorithm of Euclidean Distance Transform in the Presence of Obstacles

Qing-Nian Zhang[1]

[1]Sun Yat-sen University, 135 Xingangxi Road, Guangzhou 510275, China
Telephone: 86 20 84115833
Fax: 86 20 84115833
Email: zqnzsu@163.com

1. Introduction

A distance transform (DT) is an operation of converting binary images, widely applied in such areas as image processing and pattern recognition. It groups pixels into feature and non-feature elements, and produces an image where each pixel is assigned a distance to the nearest feature element (Gunilla, 1984). Rosenfeld and Pfaltz (1966) conducted an early work on the DT, where the distance was calculated based on the city-block and chessboard distance functions by a two-scan sequential algorithm. A large number of algorithms were proposed afterwards. Recent researches focus on improving the accuracy of distance and the simplicity and time efficiency of DT algorithms. Most of them compute distances in the absence of obstacles, or neglecting the existence of obstacles.

However, there are various obstacles, such as rivers and mountains, in the real world, which block the delivery of the substances from feature elements to nearby elements. In such situations, the delivery distance from a feature element on one side of an obstacle to another element on the opposite side of the obstacle is larger than the distance in a straight line between them. Therefore, it is necessary to take obstacles into consideration in the DT, so as to approximate accurate delivery distances in real world. This kind of DT can be applied in such areas as viewsheld analysis, object segmentation, thickness measurement, route planning and impact area.

At present, only a few DT algorithms take obstacles into account. All these algorithms compute the distances around the feature elements circle by circle outwards, and judge the visibility of a pixel by ingenious techniques. Lantuejoul and Maisonneuve (1984), and Piper and Granum (1987) are the earliest researchers to consider obstacles in the DT, while pixels are located on different circles outside feature elements, and the distances are accordingly calculated circle by circle. Coeurjolly et al. (2004) proposed a DT algorithm, which checks the visibility of a pixel using obstacle lists. Each obstacle list contains the set of obstacles sorted by polar angles met during the visibility propagation associated to each source. Cárdenes et al. (2010) put forward a method based on detecting occlusion points and comparing the angle related to occlusion points. All of them are complicated and difficult to implement.

This paper proposed a DT algorithm in the presence of obstacles. The algorithm propagates distance by scanning the plane for two runs, and checks the visibility of pixels by an algorithm of line rasterization. It is simple in principal and easy to implement, with a linear time complexity. Results showed that the new method created pictures with accurate distances.

2. Methodology

Our method was developed on the basis of existing two-scan DT algorithm. In the case of 8-connected neighbors, the main processes of two-scan Euclidean DT are described as follows. Firstly, scan the plane from top to bottom, updating the value of current pixel according to the distances of its neighbors at the left, upper left, upper, and upper right side. Then scan the plane from bottom to top, updating the value of current pixel according to the distances of its neighbors at the right, lower right, lower, and lower left side in a similar way.

In distance transform process, the distance of the target pixel can be computed using a vector related to a feature element (Figure 1). The vector of the target pixel related to a feature element is updated according to the distance vector of its neighbor to a feature element and that of its neighbor to the target pixel. The formula is defined as follows.

$$a_{ij} = a_{uv} + \overrightarrow{P_{ij}P_{uv}}$$

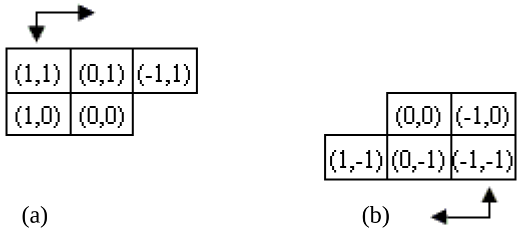


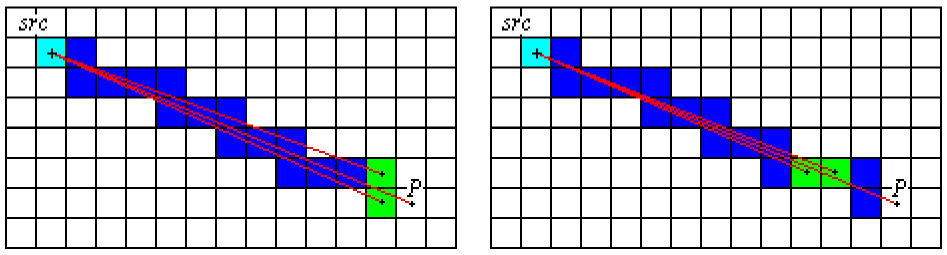
Fig. 1 Distance vector in raster scan algorithm

Where a_{ij} is the distance vector of target pixel P_{ij}, a_{uv} is the distance vector of neighbor pixel P_{uv}, $\overrightarrow{P_{ij}P_{uv}}$ is the distance vector from P_{ij} to P_{uv}.

Unfortunately, distance vector approach cannot be applied to DT in the presence of obstacles. If an obstacle lies between the target pixel and a feature element, the shortest path between them is a polyline turning in front of the obstacle, and thus does not confirm to the straight path determined by the distance vector of the target pixel.

In this paper, we change the two-scan algorithm based on distance vector approach to check the visibility of the target pixel and bend the path to the target pixel in front of obstacles. In order to extend the path behind the obstacles, we add auxiliary feature element near the obstacles to construct successive segments of the path behind the obstacles. By adding virtual sources to distance transform, our algorithm proceeds distance propagating behind an obstacle by switching the path from one line segment to another, which makes the calculated distances verging on reference value. Furthermore, our algorithm checks the visibility of last level source of a neighbor pixel, introduces a mechanic to track back to nearer feature element on last level, and thus avoids the unnecessary detour in the computed path.

A key issue is to check the visibility related to the source in the algorithm described above. This paper proposes an efficient method based on full path rasterization to check the visibility of pixels, which need not to check whether each pixel on the path is an obstacle. It works by checking the visibility of the pixels on the beginning part of the fully rasterized line. The full rasterization process is executed in the direction from the target pixel, *P*, to related source, *src*, and is expected to be terminated after checking two more rows or columns to blacken certain pixels (Figure 2).



(a) (b)

Fig. 2 Check the visibility of pixel P

3. Case Studies

In order to verify our algorithm, we constructed an image with 128 rows and 128 columns. There are two feature elements O_1 and O_2, two obstacles B_1 and B_2 in the image. We calculated manually the reference distance values, as shown in Figure 3a. The distances calculated by our method are almost the same as those in the reference image, as shown in Figure 3b. The errors related to reference image were shown in Figure 3c. There is no error in the area where the pixels are not obscured by obstacles, and the error in the area behind the obstacles is no more than 1 pixel. Furthermore, the size of deviation is related to the position of obstacles, while in no relation to the distance from feature elements.

We also computed the distance by using the ArcGIS software, as shown in Figure 3d. Obviously, the error by ArcGIS is far larger than that by our algorithm.

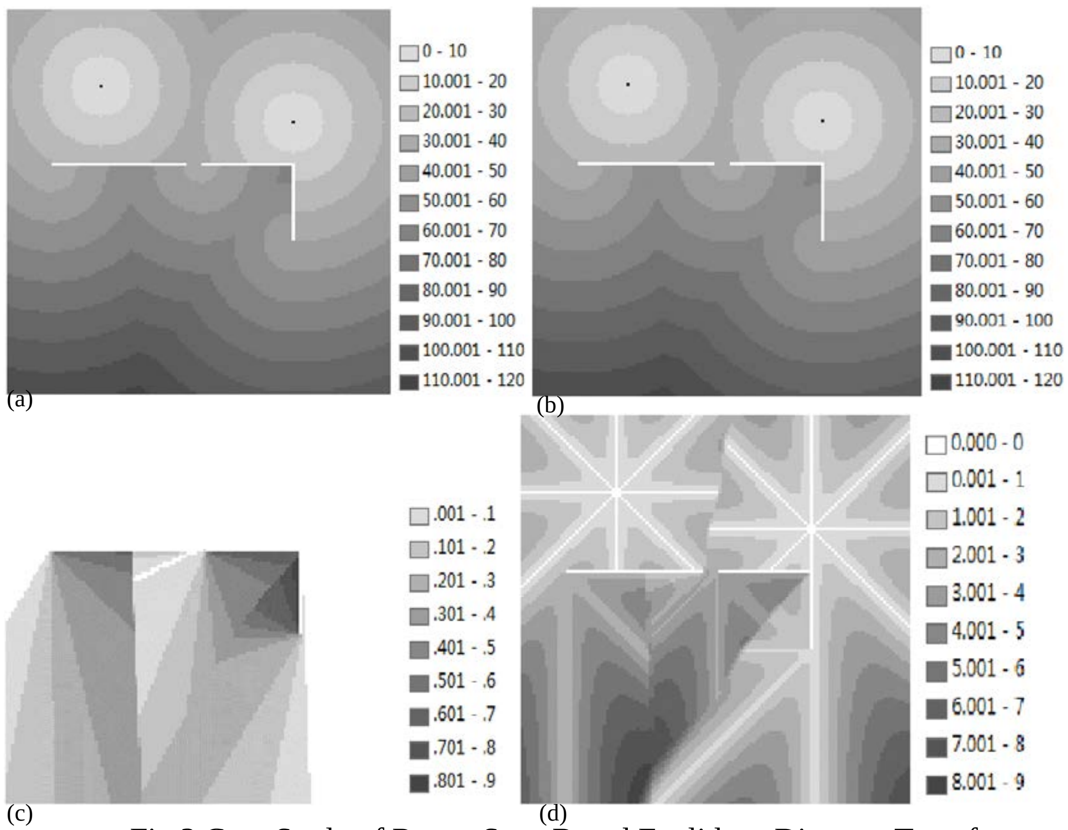


Fig.3 Case Study of Raster Scan Based Euclidean Distance Transform

4. Conclusions

This paper presented a new method to compute the Euclidean distance in the presence of obstacles. The algorithm is easy to implement, having a computational complexity in $O(n)$ order. It produces accurate distance pictures.

5. Acknowledgements

This work was supported by National Natural Science Foundation of China under grant 40971210.

6. References

Borgefors G. 1986, Distance transformations in digital images. *Computer Vision, Graphics and Image Processing*, 34: 344 – 371.

Cárdenes R, Alberola-López C and Ruiz-Alzola J. 2010, Fast and accurate geodesic distance transform by ordered propagation. *Image and Vision Computing*. 28: 307 – 316

Chen L. 1995, Optimal algorithm for complete euclidean distance transform. *Chinese J. Computers*, 18(8): 611-616.

Coeurjolly D, Miguet S and Tougne L. 2004, 2D and 3D visibility in discrete geometry: an application to discrete geodesic paths. *Pattern Recognition Letters*. 25: 561 – 570

Fabbri R, Costa L D F, Torelli J C et al. 2008, 2D Euclidean distance transform algorithms: A comparative survey. *ACM Computing Surveys*, 40(1): 1-44

Gustavson S and Strand R. 2011, Anti-aliased Euclidean distance transform. *Pattern Recognition Letters*, 32: 252 – 257

Lantuejoul C and Maisonneuve F. 1984, Geodesic methods in quantitative image analysis, *Pattern Recognition*. 17: 177 – 187.

Lucet Y. 2009, New sequential exact Euclidean distance transform algorithms based on convex analysis. *Image and Vision Computing*, 27(2): 37~44.

Paglieroni D W. 1992, A unified distance transform algorithm and architecture. *Machine Vision and Applications*, 5(1): 47-55

Piper J, 1987, Granum E. Computing distance transformations in convex and nonconvex domains. *Pattern Recognition*. 20 (6): 599 – 615.

Rosenfeld A and Pfaltz J. 1966, Sequential operations in digital picutures processing. *Journal of the ACM*, 13(4): 471-494.

Saito T and Toriwaki J. 1994, New algorithms for euclidean distance transformation of an n-dimensional digitized picture with applications. *Pattern Recognition*, 27: 1551 – 1565.

Wang Z, Li W and Pang Y. 1998, An algorithm for complete euclidean distance tranformation based on contour tracing. *Chinese J. Computers*, 21(3): 217-222.

Xu D, Ren H, Xu H and Zhao P. 2009, Improved distance transform algorithm based on chain code.*Computer Engineering and Applications*, 45(25): 176-178.