

The Polar Distance Transform by Fast-Marching

Robin Strand and Kristin Norell*

Centre for Image Analysis

Uppsala University and Swedish University of Agricultural Sciences

Box 337, SE-75105 Uppsala, Sweden

E-mail: {robin,kristin}@cb.uu.se

Abstract

Image analysis tools that process the image using polar coordinates are needed to avoid the interpolation from polar to cartesian coordinates. We present a tool for analysing and processing circular objects – the polar distance transform computed by fast-marching. The fast marching method can be used for computing the grey-weighted distance transform by numerically approximating the Eikonal differential equation. We modify the Eikonal equation using weights that depend on the radius and angle relative to a pre-defined coordinate system.

1 Introduction

Given a binary image, the distance transform (DT) assigns each object grid point the distance to the nearest background grid point. Different DTs exist, giving different distance maps. When using the Euclidean distance measure, the distance between two pixels is defined by a straight line. However, this distance function is not always well-suited for digital grids, and may not be suitable for all applications.

By weighting the DT with a cost image, the *grey-weighted DT* can be computed by assigning to each object grid point the grey-weighted distance to the nearest background grid point.

Many algorithms for computing the grey-weighted DT have been proposed. For path-based distances, the distance between two points is defined as a minimal cost path. For this family of distance functions, a technique based on wave-front propagation can be used (see [6] and the papers cited there). If, instead, the Euclidean

distance is computed, the fast-marching method (FMM) described in [5] can be used.

The FMM gives an approximate solution to the Eikonal equation. In its original form, though the algorithm is consistent and stable, the DT obtained by FMM contains errors. For small distances, the errors are significant, see e.g. [1]. In [2], the errors are reduced by, in conjunction with the stencils used in the original algorithm, using stencils rotated by $\pi/4$.

In [4] the polar distance transform (PDT) and grey weighted polar distance transform (GWPDT) are computed using the path based technique. In this paper, we present a FMM to treat this problem. The PDT is a non-isotropic distance function, where different weights are used in the radial and angular directions. Ideally, a stencil aligned to these directions should be used. In the digital grid, however, this is in general not possible, so we use a best solution obtained by interpolating to the stencils that can be used in the digital grid. Here we also allow rotated stencils, as in [2].

We also include an example where the GWPDT is computed by FMM on a wood log end face image. This can be used as a tool for outlining the annual ring pattern, which is related to the quality of the wood ([3]).

2 The polar distance transform

The polar distance transform (PDT), presented in [4], is a DT, where a shortest path is not generally defined by a straight line. With the PDT, the distance between two pixels does not only depend on their spatial relation to each other, but also on the location compared to an image origin. For two pixels on the same radius, the shortest path is a segment of a (digital) circle. To achieve this, different weights are assigned to the radial and angular directions. A high weight in the radial direction compared to the angular will propagate the distance faster in the angular direction.

*Kristin Norell is financially supported by The Swedish Timber Measurement Council.

In the computations of the PDT, the polar coordinates, (r, θ) , are computed for each pixel. The origin of the image is determined prior to the computations, either by the user or by some automatic method, and should be the (approximate) centre of the circular pattern or object. To propagate the distance from a pixel, p , a local coordinate system is placed in p , with the unit vectors $\mathbf{v}_1 = (v_1(1), v_1(2))$ and \mathbf{v}_2 in the radial and angular direction, respectively (see Figure 1). A large cost for the direction \mathbf{v}_1 compared to \mathbf{v}_2 gives a DT that propagates faster in the angular direction than the radial. The costs for the two directions are set by two weights.

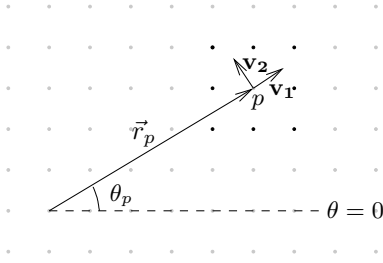


Figure 1. The geometry for the PDT on a digital grid. A local Cartesian coordinate system is placed in the pixel p . The black grid points represent the 8-neighbours of p .

3 The fast-marching method

In the FMM, a solution to the Eikonal equation is approximated using wave-front propagation. As the wave propagates, the already known values, called *frozen* grid points, are used to update the unknown points. In the update, only unknown grid points that are adjacent to at least one frozen point are considered. The unknown grid point with the lowest value is frozen, and the update is repeated, including also the unknown grid points that are adjacent to the new frozen point. The procedure is iterated until all grid points are frozen.

The Eikonal equation at a point $(a, b) \in \mathbb{R}^2$ is $|\nabla T|F = 1$, or

$$\left(\frac{\partial T(a, b)}{\partial x}\right)^2 + \left(\frac{\partial T(a, b)}{\partial y}\right)^2 = \frac{1}{F(a, b)^2}, \quad (1)$$

where $F(a, b)$ is a cost function and $T(a, b)$ is the arrival time for the wave-front at the point (a, b) . The partial derivatives of $T(a, b)$ represents the rate of speed, with which the wave moves in that point. A point with

a low cost $F(a, b)$ is thus a point where the wave moves fast.

Here we want to approximate the derivatives along the vectors \mathbf{v}_1 and \mathbf{v}_2 , mentioned in Section 2, which are generally not aligned to the coordinate axes of the grid. Given a vector \mathbf{v} , the directional derivative of a differentiable function T at (a, b) in direction of \mathbf{v} is

$$D_{\mathbf{v}}T(a, b) = \frac{\mathbf{v}}{|\mathbf{v}|} \cdot \nabla T(a, b). \quad (2)$$

Thus, given two vectors $\mathbf{v}_1, \mathbf{v}_2$ forming an orthogonal, right-handed basis (i.e. $\mathbf{v}_1 = (\cos \theta, \sin \theta)$ and $\mathbf{v}_2 = (-\sin \theta, \cos \theta)$ for some θ) and a function T that is differentiable at (a, b) , we get

$$\begin{aligned} D_{\mathbf{v}_1}T(a, b)^2 + D_{\mathbf{v}_2}T(a, b)^2 &= \\ (v_1(1) \cdot T_x + v_1(2) \cdot T_y)^2 + (v_2(1) \cdot T_x + v_2(2) \cdot T_y)^2 &= \\ = \frac{\partial T^2}{\partial x^2} + \frac{\partial T^2}{\partial y^2} = |\nabla T|^2. \end{aligned}$$

To achieve different speed for the different directions of \mathbf{v}_1 and \mathbf{v}_2 in the PDT we use a weighted version of Eq. 1:

$$(c_1 D_{\mathbf{v}_1}T(a, b))^2 + (c_2 D_{\mathbf{v}_2}T(a, b))^2 = \frac{1}{F(a, b)^2}, \quad (3)$$

where c_1 and c_2 are weights. Here, these weights are used to favour angular movement over radial movement of the wave-front.

3.1 Finite difference approximations and implementation

The partial derivatives of T at position (a, b) in the x - and y -directions are approximated by finite differences:

$$\frac{\partial T}{\partial x} \approx T(a+1, b) - T(a, b) \quad (4)$$

$$\frac{\partial T}{\partial y} \approx T(a, b+1) - T(a, b). \quad (5)$$

When \mathbf{v} is, e.g., the vector $(1, 1)$, it is possible to use Eq. 2 to approximate $D_{(1,1)}T(a, b)$. A better approximation is obtained by using finite differences

$$D_{(1,1)}T(a, b) \approx \frac{T(a+1, b+1) - T(a, b)}{\sqrt{2}}. \quad (6)$$

Let $\mathbf{v} = \mathbf{R} \cdot (x, y)$, where \mathbf{R} is the rotational matrix

$$\mathbf{R} = \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}.$$

For $0 \leq \theta \leq \pi/4$, the derivative in direction of \mathbf{v} is approximated through linear interpolation as

$$D_{\mathbf{v}}T(a, b) \approx \frac{T_x(a, b) \left(1 - \frac{\theta}{\pi/4}\right) + D_{(1,1)}T(a, b) \left(\frac{\theta}{\pi/4}\right)}{\sqrt{1 + \tan^2(\theta)}}. \quad (7)$$

Equation (7) is used to interpolate the values of the partial derivatives that appear in the digital grid to the angular and radial direction (where the weights are known). For values of $\theta \notin [0, \pi/4]$, straight-forward variations of formula 7 can be used by interpolating derivatives in directions of adjacent grid points.

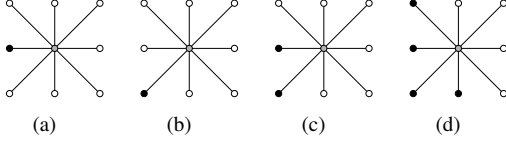


Figure 2. The different cases of frozen grid points in the neighbourhood of p (centre pixel). Distance information is propagated from one point (a,b), two points (c), or four points (d). Filled points represent frozen grid points, while non-filled represent unknown.

Let \mathbf{q} be a point for which a new distance value is to be computed, \mathbf{v}_1 be a unit vector in the radial direction, and \mathbf{v}_2 be a unit vector in the angular direction, such that $\mathbf{v}_1, \mathbf{v}_2$ form an orthogonal, right-handed basis. The rotation matrix \mathbf{R} is defined by the angle between \mathbf{v}_1 and the x -axis. Depending on the configurations of the adjacent grid points with frozen values, we get the following cases:

- When there are four grid points with frozen values in the 8-neighbourhood of \mathbf{q} , symmetric to Figure 2(d), Equation (7) is used for both vectors \mathbf{v}_1 and \mathbf{v}_2 .
- When there are two grid points with frozen values, symmetric to Figure 2(c), Equation (7) is used for vector \mathbf{v}_1 or \mathbf{v}_2 , depending on the angle θ .
- To get an algorithm that handles all possible configurations, there should also be a way to update using only one frozen grid point, symmetric to Figures 2(a) and 2(b). In this case, the finite difference approximation in (4), (5), or (6) must be used. However, the weights are only defined in direction of \mathbf{v}_1 and \mathbf{v}_2 , so the weights are interpolated using the angle θ . In other words, we approximate a weighted version of Equation (3):

$$\left(c'_1 \frac{\partial T}{\partial x}\right)^2 + \left(c'_2 \frac{\partial T}{\partial y}\right)^2 = \frac{1}{F(a, b)^2}, \quad (8)$$

and the corresponding equation obtained with a stencil rotated by $\pi/4$. For example, when $0 \leq \theta \leq \pi/2$, the following weights are considered for Equation (8):

$$c'_1 = c_1 \left(1 - \frac{\theta}{\pi/2}\right) + c_2 \left(\frac{\theta}{\pi/2}\right) \text{ and}$$

$$c'_2 = c_1 \left(\frac{\theta}{\pi/2}\right) + c_2 \left(1 - \frac{\theta}{\pi/2}\right).$$

We have now a framework in which we can use any weights c_1 in radial and c_2 in angular direction.

4 Experiments and results

Figure 3 shows PDT computed by FMM on images with the origins in the centers. In Figure 3(a) the weights $c_1 = 1$ and $c_2 = 1$ have been used, i.e., the directions of \mathbf{v}_1 and \mathbf{v}_2 are given the same weight, and thus the value of the distance map at a pixel p is not affected by the position of the pixel relative to the centre. Using the weights $c_1 = 1$ and $c_2 = 1/r$ (Figure 3(b)) we get a distance measure that propagates faster in the angular direction than the radial. The distance maps (with values modulo 10 and 1) are shown in Figure 3(a) and 3(b), respectively. In Figure 3(b) c_2 is a function of the radius, and thus the relation between the weights is different close to the origin, than for larger values of r . This causes a PDT that varies fast close to the origin. This, together with the modulo plot, explains the behaviour close to the origin in Figure 3(b).

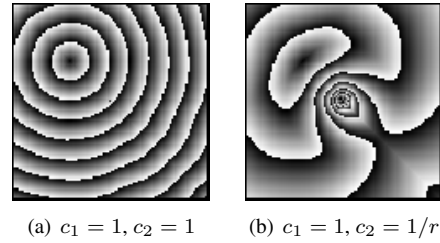


Figure 3. Distance maps (modulo 10 and 1, respectively) of 100×100 pixels. The origins are in the centers of the images.

An application for the GWPDT is outlining the annual rings on log end face images. In Figure 4 we show an end face image captured with a CCD camera, pixeLINK PL-A782, at a sawmill while the log passes on a conveyor with the speed of approximately 70 m/min. The origin is the centre of the annual rings, placed in the lower left of the image and the object, from which the GWPDTs are computed, is the white line above the origin. The GWPDTs have been computed using the FMM with $c_1 = 1$ and $c_2 = 1$, $c_1 = 1$ and $c_2 = 1/100$ and $c_1 = 1$ and $c_2 = 1/r$. The resulting distance maps modulo 300, 3 and 3 can be seen in Figures 4(b)–4(d), respectively.

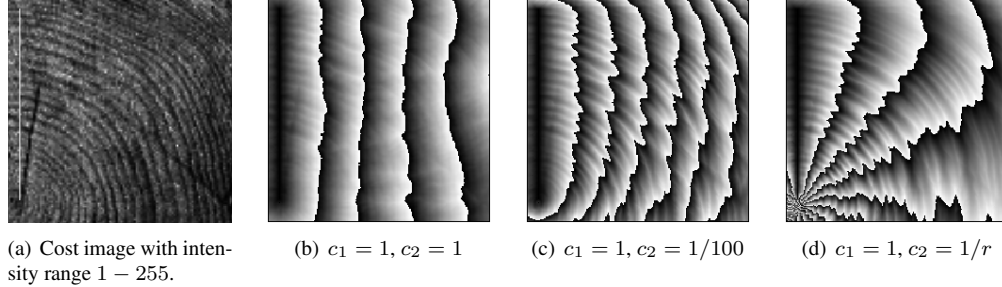


Figure 4. Distance maps computed on annual ring pattern. The image is 200×200 pixels with the origin in the lower left corner. The object is superimposed in white in Figure 4(a).

For the case when $c_1 = c_2 = 1$, the relative position (radius and angle) has no effect on the DT. We can see that the annual ring pattern is visible in the distance map mostly close to the object. As the distance propagates from the object, the influence of the underlying cost-image decreases.

In Figure 4(c) a smaller weight is used for the angular direction, giving a slower propagation in the radial direction than the angular. We can see that the ring pattern is visible further away from the object. This is because of the similar geometry of the PDT and the ring pattern, which make them enhance each other.

The length of a circular segment, s is given by $s = \Delta\theta \cdot r$, where $\Delta\theta$ is the angle that is covered by the circular segment on radius r . We want the distance function to be independent of r , so we define the cost to for a circular segment s to be inversely proportional to r . This is obtained when using $c_1 = 1$ and $c_2 = 1/r$. This can be useful when the object is elongated as in the case with the annual rings, see Figure 4(d).

5 Discussion

We have performed computations of the polar distance transform using the fast-marching method. By processing the objects in polar coordinates, there is no need to interpolate the polar coordinates to a cartesian grid. Since information is lost when interpolating, we gain precision by using the PDT. We also gain computation time, since the wave-front algorithm that is used has the same complexity as the original FMM, $O(n \log n)$, where n is the number of grid points. This follows from the fact that, as for the original FMM, the non-frozen grid point in the list with the lowest distance value is frozen and its non-frozen neighbours are updated. Each grid point is frozen once, which is linear with the number of pixels, and keeping the list sorted is logarithmic.

Future work is to evaluate the method by comparing it to the path-based PDT presented in [4] and corresponding results obtained by first interpolating the cost image to cartesian coordinates. Also, the PDT will be applied for finding the annual rings in log end face images, where the rings can be outlined in the propagation of the PDT. By using optimal weights c_1 and c_2 , the distance between two points $(r \cos(\theta_0), r \sin(\theta_0))$ and $(r \cos(\theta_1), r \sin(\theta_1))$ should ideally depend only on $|\theta_0 - \theta_1|$ and be independent on r . The result obtained in Figure 4(d) suggests that our method gives almost perfect result for $c_1 = 1$ and $c_2 = 1/r$ for the cost image in Figure 4(a). It is also clear that the method favours the annual rings in the cost image Figure 4(a).

References

- [1] J. A. Bærentzen. On the implementation of fast marching methods for 3D lattices. Technical report, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Kgs. Lyngby, 2001.
- [2] M. Hassouna and A. Farag. Multistencils fast marching methods: A highly accurate solution to the eikonal equation on cartesian domains. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(9):1563–1574, Sept. 2007.
- [3] I. Kliger, M. Perstorper, G. Johansson, and P. Pellicane. Quality of timber products from Norway spruce. part 3. Influence of spatial position and growth characteristics on bending stiffness and strength. *Wood Science and Technology*, 29(6):397–410, 1995.
- [4] K. Norell, J. Lindblad, and S. Svensson. Grey weighted polar distance transform for outlining circular and approximately circular objects. In *Proceedings of ICIAP 2007*, pages 647–652. IEEE Computer Society, 2007.
- [5] J. A. Sethian. *Level Set Methods and Fast Marching Methods*. Cambridge University Press, 1999.
- [6] R. Strand, F. Malmberg, and S. Svensson. Minimal cost-path for path-based distances. In *Proceedings of ISPA 2007, Istanbul, Turkey*, pages 379–384, 2007.