

Distance transforms as a new tool in spatial analysis, urban planning, and GIS

Michael J de Smith

Centre for Advanced Spatial Analysis, University College London, 1–19 Torrington Place, London WC1E 7HB, England; e-mail: mike@desmith.com

Received 17 December 2002; in revised form 3 May 2003

Abstract. Many spatial datasets and spatial problems can be described with reference to regular lattice frameworks rather than continuous space. Examples include: raster scan and digital elevation model data, digital images, cost surfaces, cellular automata models, swarm models, and many others. This raises the question as to how distances should be measured in such cases and to what extent these relate to continuous space metrics. In this paper I show that a set of image processing algorithms known as distance transforms (DTs) may be applied to such datasets and can be extended to solve a wide range of 2D and 3D optimisation problems. These extended versions of the standard DT procedure have applications in many areas including location theory, path determination, planning, and decision support. As such I argue that they warrant consideration for inclusion as a standard set of tools within modern GIS and spatial analysis software packages. Sample pseudo-code for the transforms discussed is included in an appendix.

Introduction

In this paper I consider applications of *local* or *neighbourhood* metrics (also known as *chamfer* metrics) and *distance transforms* (DTs) that utilise such metrics over square lattices. In the first section of this paper I provide an introduction to DTs as developed within the field of image processing. I then provide a summary of results that have been obtained in recent years on the accuracy and optimum design of such transforms, notably in two-dimensional (2D) applications. This initial review leads on to an examination of the application of DTs to a range of problems in spatial analysis and urban modelling. Simple uses of existing DT algorithms are reviewed, with a first extension, application to problems involving obstructions.

Three further sections are then provided, each of which introduces a new extension to existing DT methods. The first of these involves combining and weighting DTs to generate solutions to problems in spatial decisionmaking—I call this method multiple weight distance transforms, or MWDTs. The second extension enables their application to the determination of shortest paths across physical landscapes, which I describe as variable topography DTs, or VTDTs. The third and final extension is perhaps the most powerful and widely applicable procedure. It addresses problems in which generalised costs (or traffic velocity) vary across a region of interest. The least cost distance transform (LCDT) that I develop has similarities to accumulated cost surface (ACS) methods and has a wide variety of applications. Examples of least cost and least time solutions to a number of classic and complex spatial problems are described. Each of these extensions involves minor alterations to a single core algorithm, which is very simple to implement and fast to run, even with large datasets. As such the use of extended DTs provides a unified methodology that may be applied to a very wide range of problems in spatial analysis.

Chamfer metrics and distance transforms

A sample of lattice metrics is shown in figure 1 (over). These are known as chamfer metrics because the locus of the metric about a single point generates a figure similar

Case	Description	Adjacency matrix		
1	Distances are determined by the L_1 or ‘city-block’ metric and paths correspond to the ‘rook’s move’ in chess parlance	2	1	2
		1	0	1
		2	1	2
2	As per 1 but with diagonal distances determined by the Euclidean metric applied locally—sometimes referred to as the local Euclidean metric.	$\sqrt{2}$	1	$\sqrt{2}$
		1	0	1
		$\sqrt{2}$	1	$\sqrt{2}$
3	Integer chamfer (3,4)/3 metric. These integer values provide an improved estimate of distance over cases 1 or 2; divide by 3 on completion.	4	3	4
		3	0	3
		4	3	4
4	Fractional chamfer (Borgefors, 1986)—optimal noninteger values for all directions (values shown are after Butt and Maragos, 1998).	1.36039	0.96194	1.36039
		0.96194	0	0.96194
		1.36039	0.96194	1.36039

Figure 1. 3×3 chamfer metrics.

to a cross-section of a piece of wood with chamfered or bevelled edges (for example, see figure 3, below). Chamfer metrics and their associated DT algorithms provide a very simple and extremely fast method for the approximation of Euclidean distances, or a multiple of Euclidean distances, from every cell of a square lattice to the *nearest cell* in a target set (Leymarie and Levine, 1992).

Distances over the lattice are calculated in an incremental manner based entirely on the distance to directly adjacent cells. The standard algorithm involves a two-pass scan of a square or rectangular lattice dataset: a forward scan from top left to bottom right, and then a backwards scan from bottom right to top left.

Each pass involves adding the values in a divided form of the adjacency matrix or *mask* to cell values in the underlying lattice—see the example masks in figure 2, where five values are used based on the (3, 4) integer-valued chamfer. The value in mask position 0 of the transformed lattice is then set to the minimum of the sums calculated. The central function in the algorithm is of the form

$$d0 = \min(d + \text{LDM}(k), d0)$$

where $d0$ is the current value at the central point (0) of the mask, $\text{LDM}(k)$ is the local distance to the k th element of the mask, and d is the current value at row r , column c of the lattice (this notation corresponds to the pseudo-code provided in the appendix). The algorithm involves in the order of Mn^2 computations where n is the maximum dimension of the lattice, and M is the number of cells used in the neighbourhood computation.

The underlying lattice is normally a binary image, but could be a single target point (or set of points) from which distances are automatically generated. In this case the target point(s) would be initialised to 0 and all other points as a large value (any value greater than the maximum possible distance to be computed, for example, 9999). On completion of the two-pass scan, each cell in the resulting lattice will contain the

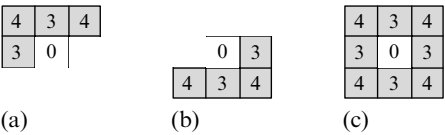


Figure 2. 3×3 integer chamfer masks for distance transformation: (a) forward scan, (b) backwards scan, (c) adjacency matrix.

distance to the nearest point in the set of target points. In the example above, division of the values by 3 can be made on completion of the scanning process, giving an approximation that will be within 6.1% of the true Euclidean distance.

There are a number of highly efficient sequential and parallel algorithms for performing this process and a great deal of research into these and the quality of approximation has been conducted for both binary and grey-scale images (for a fuller discussion see Leymarie and Levine, 1992). The subject is known as distance transforms (DTs)—DTs are used in a wide variety of image recognition and processing applications (for which they were designed). Example applications include image matching, skeletonisation, and 3D rendering.

Accuracy of distance transforms

By plotting the locus diagrams for a range of chamfer metrics against the optimal Euclidean locus (a circle) the relative merits of different local values or weights can be seen (figure 3, over). On examining the diagrams it is clear that the first two approximations underestimate most distances. The third approximation, based on integer computations, is very good, but the last of the four approximations achieves the closest possible match to the circle, with a mix of positive and negative errors at intervals of $\pi/4$ (positive) and $\pi/8$ (negative). The octagonal shape of all but the first example is the result of the 8-cell local neighbourhoods utilised. If a 5×5 local neighbourhood is used the locus is 16-sided (a 'hexadecagon'). The symmetry displayed and closeness of the best approximations to the Euclidean metric means that optimal chamfer metrics are nearly, but not completely, rotationally invariant. Integer values are frequently used in distance transforms for maximum processing speed, but Borgefors (1986) recognised that the approximation to Euclidean distance could be improved upon by permitting fractional values. She used Cartesian coordinate pairs to produce her noninteger results—this model generates a result that is nearly, but not fully, optimal in the propagation of distances around a point in a lattice. In a detailed analysis using polar coordinates, Butt and Maragos (1998) have since shown that the values derived by Borgefors can be improved upon marginally—their detailed results for the 3×3 neighbourhood are shown in table 1 (over): the (3,4)/3 metric is the best low-valued integer solution and yields correct values for horizontal and vertical paths; other choices involving larger integer pairs, such as (24,34)/25, can be used to approximate noninteger variants quite accurately.

There is a related result that may be obtained from this table. It relies on the fact that between any two points on a square lattice it is always possible to construct a path consisting of two components, one diagonal path and one horizontal or vertical path. If two points in a square lattice are selected at random, and these are M steps (horizontal/vertical) and N steps (diagonal) apart then the best estimated distance between them is: $d = 0.96194M + 1.36069N$. The maximum absolute error in this calculation is 3.96% of the larger of the horizontal or vertical distances.

As noted above, the lattice neighbourhood can be increased to a 5×5 matrix, in which case there are three distance weights to be assigned to the various cells rather than two, and the optimum fractional values in this case are (0.9866, $\sqrt{2}$, 2.2062). These values provide estimates that are within 1.36% of the direct Euclidean distance but at the cost of slightly increased computation. The integer value optimum values are (5, 7, 11)/5, and are remarkably accurate—within 2% of the Euclidean distance. The integer neighbourhood (mask) for the 5×5 model is shown in figure 4, (over)—values not entered are predetermined (for example, as $5 + 5 = 10$ or $7 + 7 = 14$). The mask is divided into two for forward and backwards scans, as for the 3×3 mask described above. The DT of

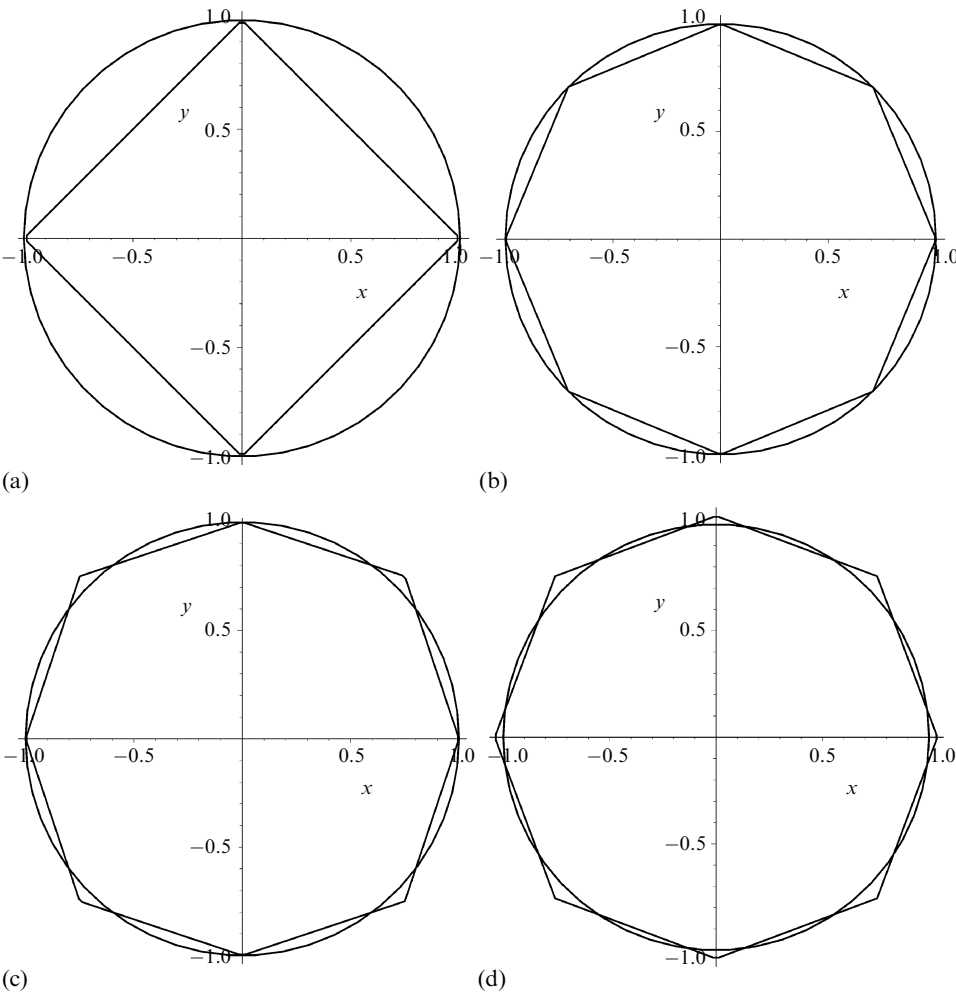


Figure 3. Chamfer metric locus diagrams: (a) chamfer (1, 2) or Manhattan metric; (b) chamfer (1, $\sqrt{2}$) or local Euclidean metric; (c) chamfer (3, 4)/3 metric; (d) chamfer (0.96194, 1.36069) optimal metric.

Table 1. Maximum absolute error for 3×3 chamfer metrics.

Local distances (a, b)	Maximum absolute error (%)	Comments
(1, 1)	41.41	Chess board ‘rook’s/bishop’s move’
(1, 2)	29.29	City block, L_1
(1, $\sqrt{2}$)	7.61	Euclidean local distance
(3, 4)/3	6.07	Integer chamfer
(1, 1.3507)	5.63	Borgefors, with $a = 1$
(1, 1.3420)	5.38	Butt – Maragos with $a = 1$
(0.95509, 1.36930)	4.69	Borgefors optimal
(0.96194, 1.36039)	3.96	Butt – Maragos optimal

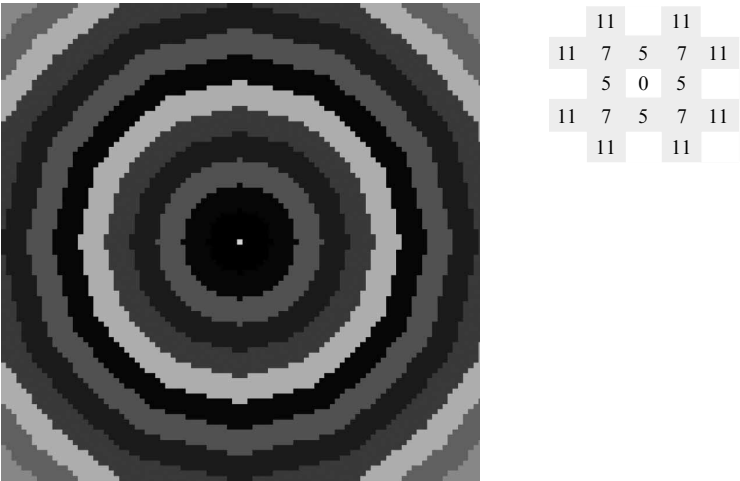


Figure 4. 5 × 5 distance transform and chamfer mask.

a point with the 5 × 5 mask can be seen to be a very close approximation to a circle over a square lattice.

With neighbourhoods of 7 × 7 or greater the maximum error falls below 1%. Results can be obtained for triangular and hexagonal lattices (Borgefors, 1989), with the latter providing improved results, but again at the cost of increased complexity in representation and processing.

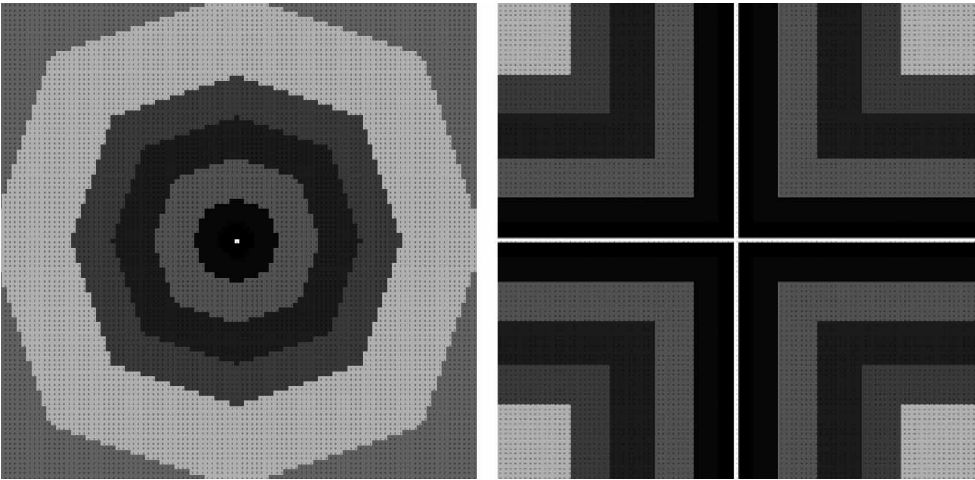
The lack of exact correspondence of Euclidean distances must be recognised and regarded as a form of systematic error or uncertainty, which may be exaggerated with scale changes and/or growth/shrinkage of objects by using DT methods (for example, topological inconsistencies). However, DT algorithms are available that provide exact Euclidean distances in near linear time, which may prove more suitable for some problems (Cuisenaire and Macq, 1999). Such algorithms may be compared with the ACS procedure of Douglas (1994) in which he utilises a spreading algorithm to generate circular (that is, exact or near exact) isolines about a single target point—these considerations are in addition to the representational issues associated with the original lattice or raster dataset.

Applications of simple distance transforms in spatial analysis

Simple DTs may be used for the fast computation of distances and multilevel buffer zones from single or multiple objects (points, lines, areas) rather than just single points, for the computation of watersheds and slope lines, and for the computation of Voronoi polygons from lattice or raster data (figures 5 and 6, over).

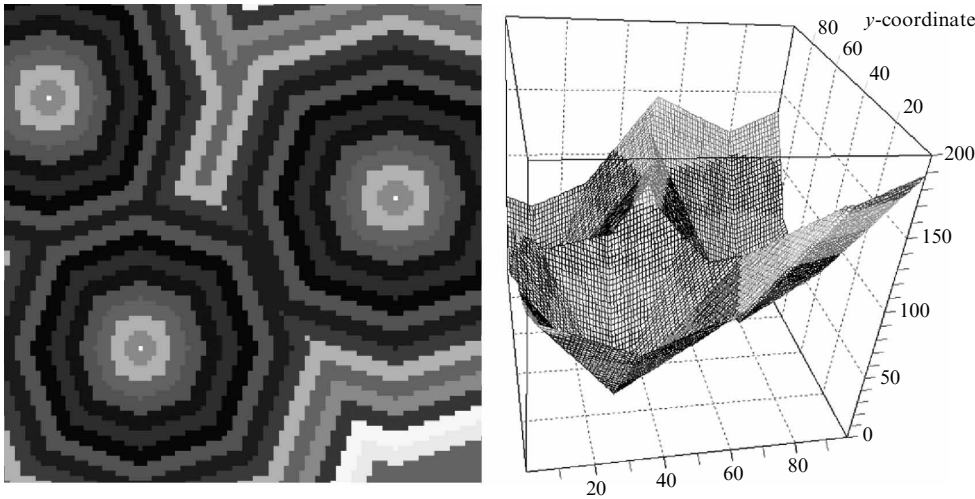
The examples in figure 5 show distance bands from object sets shown in white, comprised of (a) a single central point, and (b) a cross shape, over a 100 × 100 lattice, generated with a single forward and backwards pass of the masks. Bands indicate the distance from the nearest point of the object set. With multiple distinct objective points the distance transform generates the equivalent of Voronoi polygons [referred to as *planar digital Voronoi regions* by Okabe et al (2000, page 51, definition V6)]. These may be mapped in two dimensions as above, or in three dimensions, where distances are treated as elevations, shortest paths are lines orthogonal to the distance contours (figure 6), and region boundaries are watersheds.

Distance transforms can also be applied with almost no alteration to cases where obstacles are included. For example, if we introduced a rectangular region as an



(a) (b)

Figure 5. Raster buffer zones from distance transform using (3,4) chamfer.



(a) (b)

Figure 6. Multipoint distance transform (a) 2D and (b) 3D views.

obstacle in the single point example above, the resulting paths in the region are distorted [figure 7(a)]—in this example the transform has been generated by using fractional values (two or more complete scans are normally required for convergence). As before, shortest paths may be computed by following paths that are orthogonal to the boundaries of the distance bands shown. However, isolines and paths are distorted by the lattice representation and shortest paths may not follow local lines of steepest descent. If necessary, path determination may be performed using smoothed isolines and/or a retained record of the optimal path, for example, by storing additional data as part of the DT algorithm [figure 7(b)]. This latter information can be obtained during the iterative process by storing the relative or absolute address of the local neighbourhood cell with the least distance or cost from the current cell.

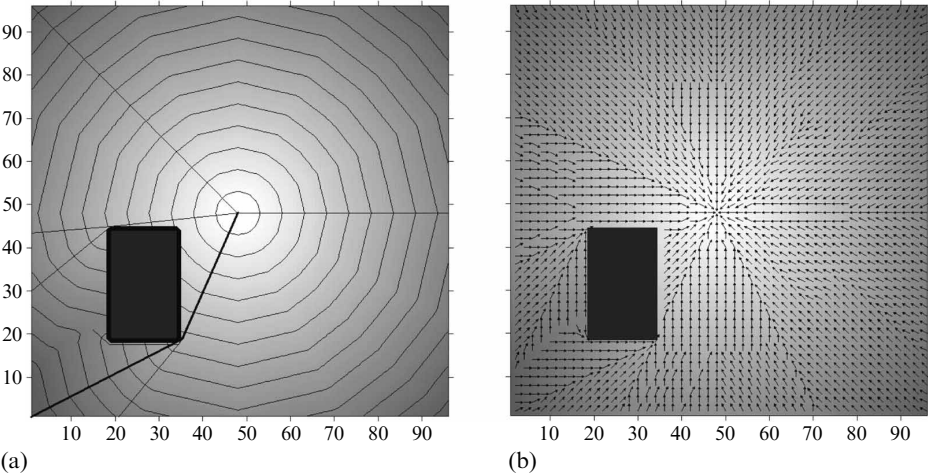


Figure 7. Shortest paths by distance transform with constraints: (a) isolines and sample shortest paths, (b) vector data for shortest path determination.

The constrained DT technique can be applied to analysis of buildings and urban form, for example, for pedestrian accessibility (Ratti, 2001). The examples illustrated in colour plates 1(a) and 1(b) (see page 95) show an area of central London that has been the subject of detailed study in connection with the management and planning of the annual Notting Hill Carnival (Batty et al, 2002).

- The procedure adopted in this example was as follows:
- (1) From a vector file of London road and transport routes the Notting Hill area was identified and the study region saved as a bitmap [plate 1(a)]; the sample region is roughly 320×250 pixels, each pixel being approximately 6 metres square; roads are all $3+$ pixels wide—a minimum of at least (DT mask size + 1) is preferable.
 - (2) The bitmap was then edited to remove Westway (an elevated road) and one or two rail or tube lines that were included in the original file, and the widths of one or two roads were increased for experimental purposes. The file was then saved as a RAW (binary) format image file.
 - (3) A small program was written to read the RAW (binary) file and recode the data so that routes were set to a temporary value of 5555, inaccessible areas (buildings etc) to 9999, and the target point in the centre of the study region ($x = 104, y = 166$) was set to 0. Values of 9999 in the file were treated as unmodifiable cells or barriers in step (4).
 - (4) A simple local Euclidean 3×3 DT with constraints (as described above) was run on the data matrix, iterated (six iterations in this example) to ensure that all points were reached and calculated distances were convergent; this example surface was saved as an ASCII GRD file in geographic (that is, not image) order, and then displayed graphically as shown in plate 1(b). The colour gradations emulate a ‘burning fire’ analogy to this type of transform (that is, as if a fire starts in the centre and grows outwards along the street network). The scale on the right-hand side of the diagram has a real maximum value of <300 units (around 1.8 km) with values above this used for the built-up (constrained access) regions of the sample space.

Processing time on this dataset is quite fast—the same problem but incorporating the carnival parade route and sound system points is computable in the same amount of time [plates 1(c) and 1(d), see over]. Larger areas or finer detail can be readily handled—a resolution of 1 m (around 2000×1500 cells) should be computable without too much difficulty, although processing time will depend on how it was coded and what processor type and operating system is used.

A number of initial observations should be made about this application of distance transforms:

- (a) The way in which the map matrix has been generated is not as satisfactory as working from an aerial photograph or high-resolution vector street map, so the layout is somewhat indicative—analyses of this type need to be computed on high-resolution datasets if they are to be used in planning and urban design programmes.
- (b) By comparing a selection of the road distances computed from a test point to the sample region borders with linear map measurement or odometer readings it is possible to confirm whether the estimation process works correctly and matches the true road map within the tolerances of the approximations made above.
- (c) Computed distance values can be used to identify optimal routing through the streets, based on unrestricted access—within the confines of relatively narrow street patterns, steepest paths in the distance transform will identify shortest paths—effectively the DT provides a form of accessibility surface with respect to the selected point(s).
- (d) Because the sampled region is finite some roads will be ‘unreachable’ from the target (that is, they are not connected to the remaining street network)—when applying DTs to problems of this type it is advisable to include a buffer zone around the study area to ensure that the best possible coverage and connectivity are achieved.
- (e) The unrestricted access assumption in this specific case, and in general, needs closer examination—there is an assumption that areas not built upon are uniformly accessible and usable (for example, by pedestrians) which is clearly not true in most cities. Depending on the requirements, models of the type described above will need to be modified if barriers across roads exist (easily incorporated); if pedestrians are restricted to pavements (manageable); and if physical traffic flows are to be considered where road-space limitations and one-way streets exist (more complicated).

A further observation is that DT solution matrices, such as those generated for the Notting Hill case study, may be analysed with simple statistical and mathematical measures enabling alternative layouts to be assessed directly. For example, in the case of the carnival parade route, the mean distance in the solution set is just over 300 m with a standard deviation of around 250 m, and a maximum value of 1 km. Alternative routes may be analysed in this way, enabling direct comparison of the solutions from an accessibility perspective, and potentially, with a view to reducing overall pedestrian movements and increasing safety. Such techniques may be applied in related problems of optimum location, such as bus route and bus stop planning, locating facilities for communities, and potentially in a wide range of other route planning and built-form design projects. Likewise, local maxima in the DT set indicate urban watersheds, enabling partitioning algorithms (for example, determination of postal delivery regions) to be applied in such environments, and facilitating the computation of distance statistics *within* partitions.

Multiple weighted distance transforms

If we denote by $DT_k\{A_i\}$ a distance transform of type k applied to an object set $\{A_i\}$, then we can define the weighted sum, z , of multiple transforms (or multiple weighted distance transform, MWDT) as

$$z = \sum_i w_i DT_k\{A_i\}.$$

This is a composite surface, or set of values, with one or more minima. In the classic three-point Steiner (or Fermat – Weber) problem, one seeks a single point, P , which minimises the sum of the distances to each of three vertices. For this problem a unique minimum value exists. However, the MWDT method may fail to locate the point P

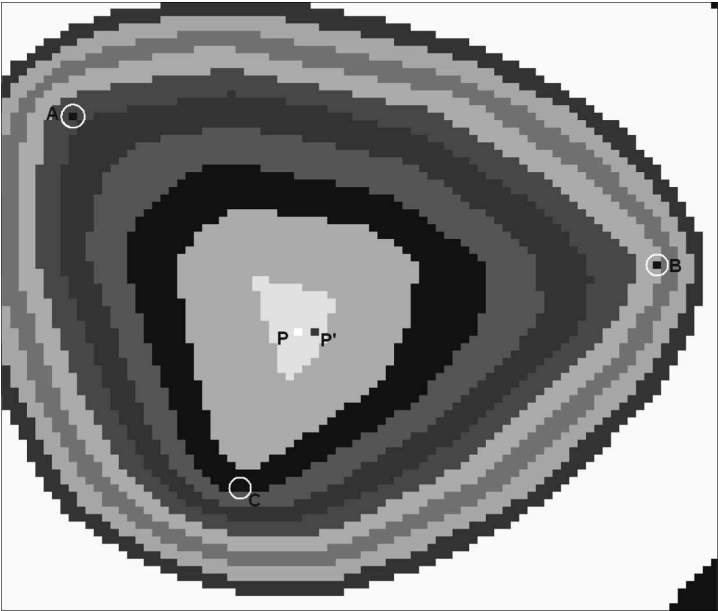


Figure 8. MWDT (multiple weighted distance transform) solution to a sample Steiner problem.

exactly because the procedure applied is not exact—the quality of the result will depend on the specific transform used, the type of lattice employed, the resolution of the lattice, and the behaviour of the cumulative surface (flatter surfaces will have similar values for the objective function over a larger area).

An example of this procedure applied to a simple problem with unit weights is shown in figure 8. The three locations A, B, C are the source points or vertices, the white square is the Steiner optimal point (P), and the square marked P' is the minimal value of the cumulative surface. Although the MWDT solution is not coincident with the optimal location, this can be explained by the observation that the region in the centre of the diagram is almost flat (under 1% variation in the objective function) and by the inherent approximations involved.

In this case a 5 × 5 fractional chamfer was used in order to ensure a close match to the location of the optimum—a 3 × 3 chamfer would have worked almost as well as far as the objective function is concerned but P' would have been further from the optimal solution. In the event the solution computed is within 0.5% of the true optimum. Of course, if all one is seeking is the location of the optimum point in an isotropic model, there are faster and simpler solutions to this problem. However, as we show below, the approach does provide the basis for solutions to a wider range of problems that are less amenable to alternative procedures.

In addition to the optimum result, figure 8 provides a valuable set of information to prospective users (for example, planners, spatial analysts, consumers)—a picture and evaluation of the possible near-ideal locations. An example to illustrate this observation is that of home finding. In choosing one's ideal home there is a wide range of factors to be considered: for example, how far is the nearest station? where are the nearest good schools? is the property close to open spaces? how near is the main shopping region? are there nearby eyesores or industrial plants? These questions can be represented as a number of distance transforms, which may be weighted if required, and accumulated, with 'decision zones' then generated from the aggregated surface. With two alternative

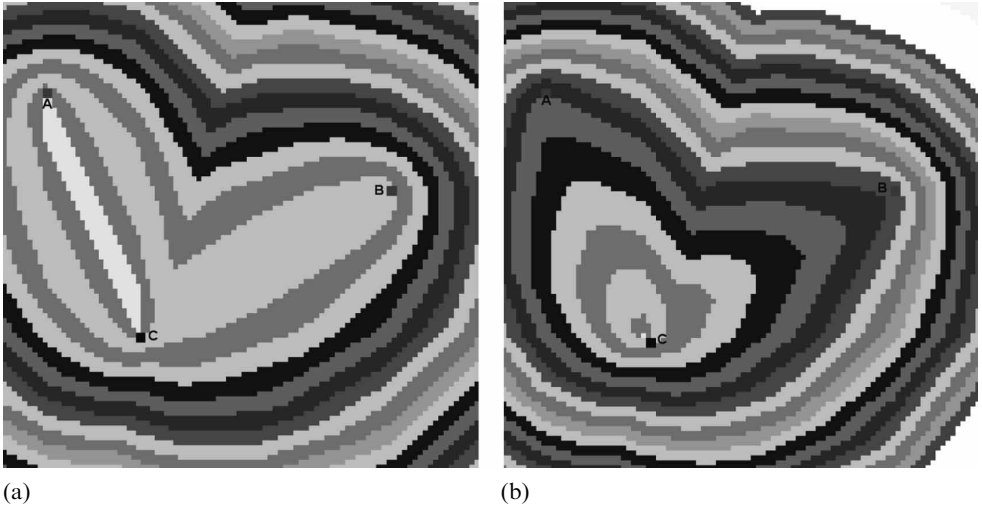


Figure 9. MWDT (multiple weighted distance transform) decision support diagram: (a) unweighted—upper two squares are the stations, lower square is the school; (b) weighted—school weight = 1.5.

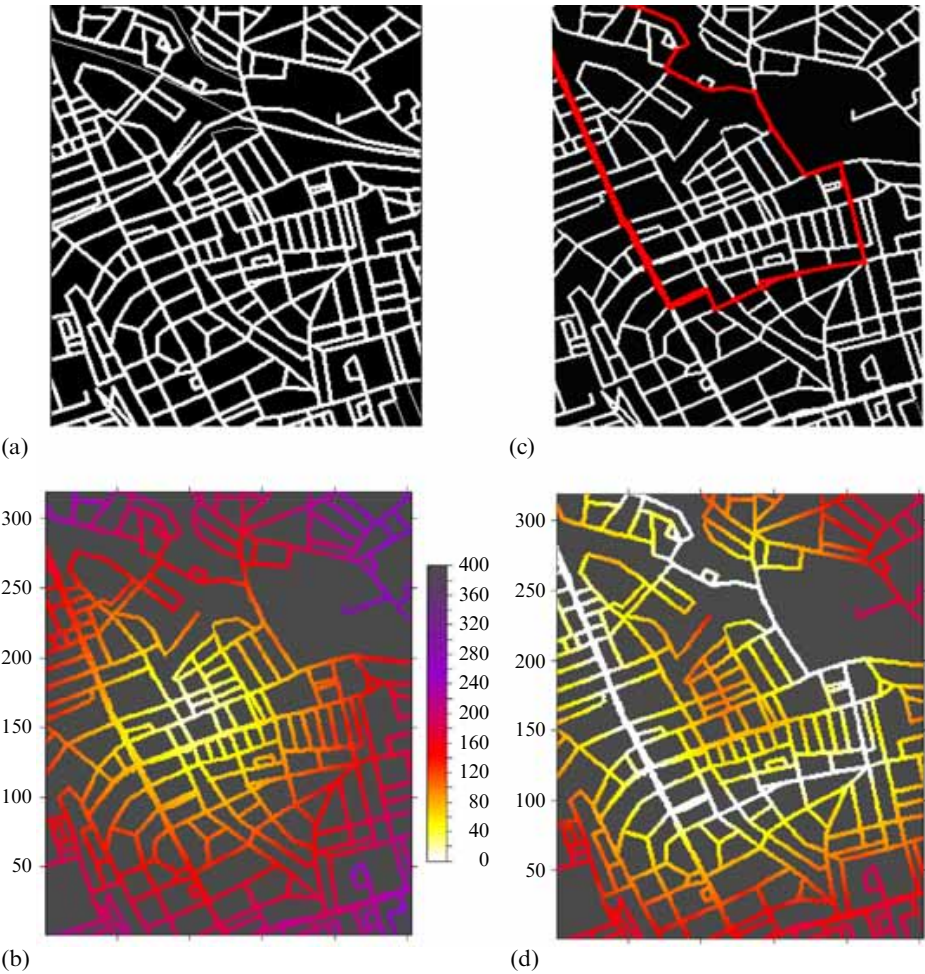
stations to choose from (A and B) and one other point, for example, a specific school (C), the combined cost surface can be generated and is shown in figure 9.

The central area and subsequent bands mark the ‘regions of indifference’, that is, the best zones from the point of view of minimal overall distance to the school and either one of the two stations. In the second diagram the additional weighting (importance/cost/time) associated with the location of the school has pulled the desirable region to search strongly towards the school. Onto these surfaces may be overlaid the actual locations of the target facilities and house-price data, in order to provide an interactive decision-support facility for the house-buyer, guiding their choice of areas to search and/or compromises to be made. One can envisage systems based on this general model being linked to GIS facilities, with web or desktop interfaces, enabling the development of a range of new interactive decision-support services, coupled with numerical interpretation of alternative selections. In this context I envisage the MWDT procedure as augmenting existing core GIS facilities, such as polygon overlay and buffer operations.

Variable topography distance transforms

The above analysis deals with the situation in which the sample space is uniform (flat) and isotropic. The use of chamfer metrics and distance transforms can be extended to a 3D model, in this case using $26 + 1$ cells (that is, 6 faces + 8 corners + 12 edges + 1 = 3^3 cells) in order to allow for all possible directions. A 3D or 2.5D digitised (lattice) surface can be viewed as a 3D graph, with arcs connecting each point to its immediate neighbours such that the arcs are weighted by a local distance estimate based on three weights (for faces, corners, and edges). In this case the optimum local weightings are no longer unique but depend upon the model assumptions one makes.

Using a range of test *surfaces*, with and without obstacles, Kiryati and Szekely (1993) found that solutions paths were readily obtained by using this procedure. However, as land surfaces are effectively open 2D manifolds, the need for a 3D model in this case is questionable. In principle a slope-adjusted 2D model should provide satisfactory solutions. Such a model, which I refer to as a variable topography distance transform (VTDT), requires computation of the estimated slope distance (height adjusted)



Colour plate 1. Accessibility and shortest paths in urban areas: (a) binary image of the street network in part of the Notting Hill area of central London; (b) distance transform of street network around point (104, 166); (c) Notting Hill Carnival—major part of the parade route highlighted; (d) distance transform of street network with respect to Carnival parade route. Scale: 1 pixel/cell $\approx$ 6 metres; gradations indicate distance by street from selected point(s).

for each cell position, and will require multiple iterations or passes rather than the normal two-pass algorithm. An example is shown in figure 10 (over).

In this example the physical surface has been modelled as a steep-sided elliptical hill, by using an elliptic paraboloid equation (see elliptical contours in the diagram) and then a cumulative least distance surface has been generated from a 5×5 height-adjusted integer DT about a point at the left centre of the diagram. The distance isolines about this point are shown and geodesics may be constructed by plotting paths orthogonal to these lines from any point in the sample region to the DT origin and/or following computed or stored optimal path vectors, as illustrated.

Real-world landscapes are rarely as simple and unless steep, dome-like structures exist, the shortest paths will be very close to straight line transects. Figure 11(a) (over) shows a relief map of a 400×400 sample 25 m digital elevation model (DEM) dataset (that is, a region of 100 km²) in the Pentland Hills area, just south of Edinburgh. The height difference between the valley to the right of the map and the highest point

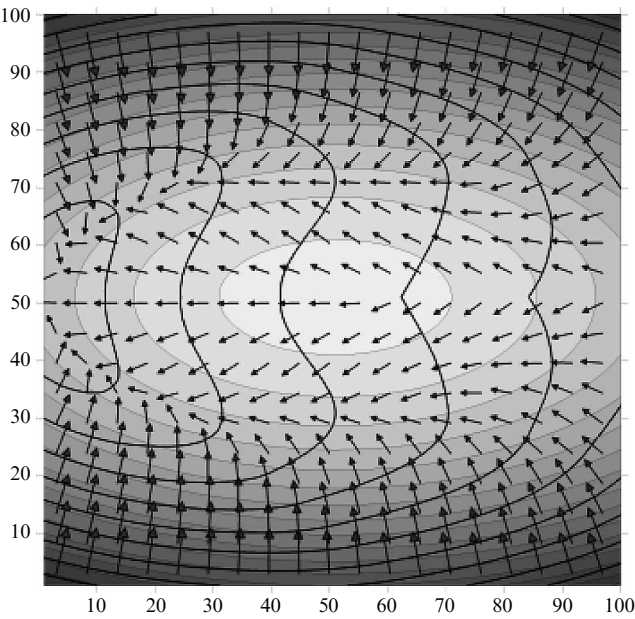


Figure 10. Geodesic path determination by distance transform.

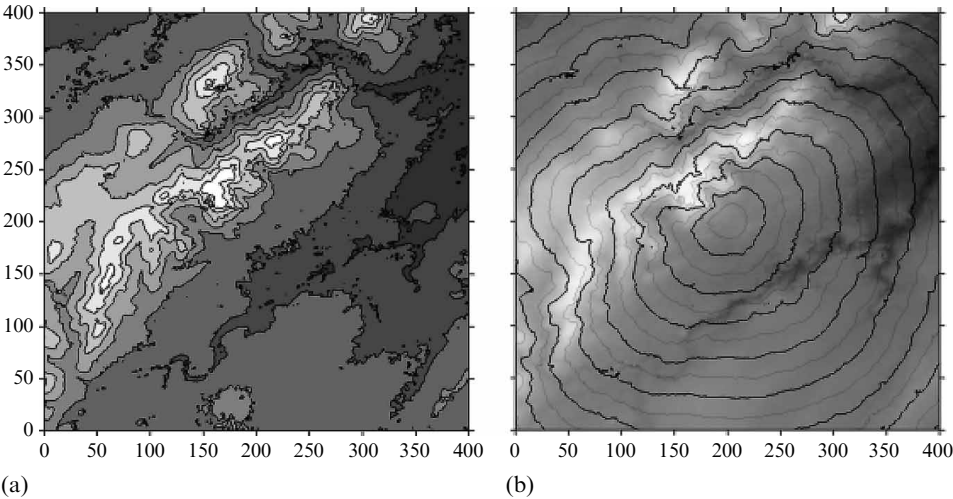


Figure 11. Relief map and distance isoline distortion in rugged terrain: (a) shaded contour map of part of the Pentland Hills, Scotland: 25 m DEM (digital elevation model) dataset; (b) image map and distance isolines: 2.5 m data model (10×rescaling).

on the hills to the left is around 300 m. However, DT isolines emanating from the map centre (200,200) are found to be almost circular, indicating that shortest paths on the surface (not least effort paths) are closely approximated by straight lines, despite the presence of the hills. This result is explained by the fact that much of this landscape actually has relatively gentle slopes (under 30°), the height difference is less than 10% of the plane distance from centre to boundary, and that the hills are broadly linear rather than deeply cut.

However, if the dataset is treated as a 10 m DEM (by introducing a lattice scale factor, p , into the standard algorithm) the rescaling results in some path modification, and when treated as a 2.5 m DEM the rescaling results in substantial distortion of

the isolines as shown in figure 11(b). This finding confirms that for most real-world landscapes (and GIS systems) a very good approximation to the shortest distance across a landscape can be obtained by computing the surface length along a transect, for example, by using (incremental plane distance)/(cosine of the slope) or DEM directly adjusted values.

Least cost distance transforms

The basic DT procedure can also be modified to take account of continuously variable costs. In this case we need to define cost values for all lattice points (a function or a set of values) and multiply the incremental (neighbourhood) distance calculations by the value of the cost function at the mask points. The central function in the algorithm is of the form:

$$d0 = \min(d + \text{LDM}(k) * (\text{COST}(r, c)), d0)$$

where $d0$ is the current value at the central point (0) of the mask, $\text{LDM}(k)$ is the local distance to the k th element of the mask, d is the current value at (r, c) and $\text{COST}(r, c)$ is the cost function at row r , column c [this cost function optionally may be averaged 50:50 with $\text{COST}(x, y)$].

I describe this procedure as a least cost distance transform (LCDT), and as such it is effectively an algorithm for generation of a form of accumulated cost surface (Douglas, 1994). The surface generated by this procedure includes lines of equal cost (*isolines* or *isodapanes*) providing the basis for subsequent determination of least cost paths (orthogonal to the isolines). To illustrate this process, albeit with a relatively coarse approximation (100×100 lattice), figure 12 shows the LCDT surface generated about the point (2, 5) with a cost function of $F = 10y + 1$ (increasing y is towards the top of the diagram, with the range in both directions being $[0, 10]$).

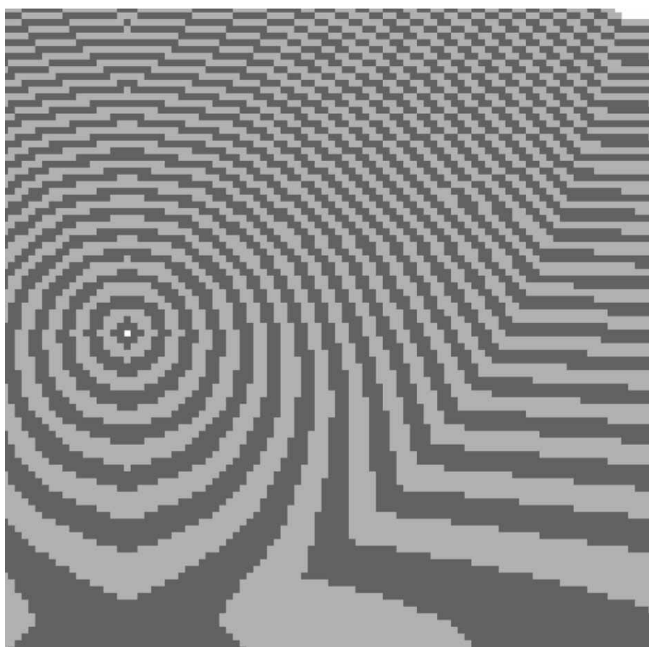


Figure 12. Distance transformed cost surface, $F = 10y + 1$.

In performing this operation the basic number of computations is the same as for a standard DT, but it is necessary to iterate the entire process T times to ensure the results converge—thus the algorithm's complexity is $O(TMn^2)$. Convergence issues and estimation of T are discussed further in the final section of this paper.

Although this procedure provides very good estimates of least cost or time distances, the lattice structure itself tends to distort optimal paths—as noted earlier, construction of smoothed isolines and orthogonal paths yield more representative results, as will computing shortest paths based upon a larger neighbourhood, for example, 5×5 or 7×7 . This is illustrated by taking a 2D contour model of the data and plotting sample paths—the diagrams in figures 12 and 13 may be compared with the graph of geodesic paths shown in figure 14, which has been generated by analytic solution methods (in this latter case, where shortest paths are shown as crossing, the geodesics are to be taken as the least overall distance or cost of the total path, which are via the x -axis for points to the right of the apparent singularity).

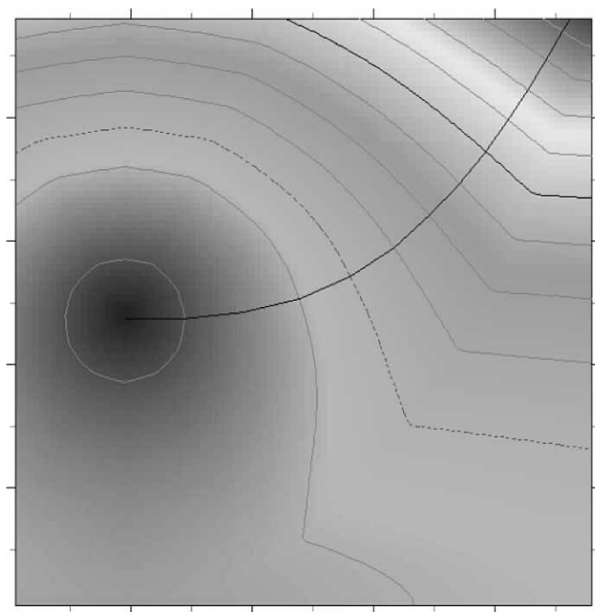


Figure 13. Isolines and sample geodesic for cost surface, $F = 10y + 1$.

Comparison of the cost distance values generated by LCDT with numerical integration of the analytic solution paths indicates that close approximation to true path costs can be achieved by this form of modified DT. Thus, on a more general level, the LCDT procedure can be viewed as a method for numerically solving certain types of nonlinear ordinary differential equation, and provides an approach for the numerical solution of a broad range of equations of this type.

A second example of the LCDT procedure is given by an analysis of the radially symmetric velocity field models of Angel and Hyman (1977). In these models the speed of road traffic is assumed to increase from the city centre in a radially symmetric manner according to a simple formula, such as $v = ar^p$, where r is the radius or distance from the centre, and a and p are parameters. The diagrams below show LCDT solutions for two cases: the symmetric model of Angel and Hyman (based on their study of traffic patterns in Manchester, England, taking $p = 1$ in this case); and

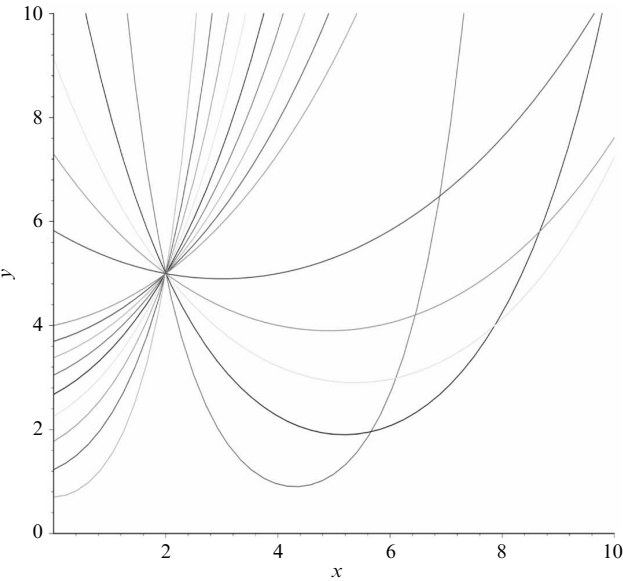


Figure 14. Analytically derived geodesics for cost surface, $F = 10y + 1$.

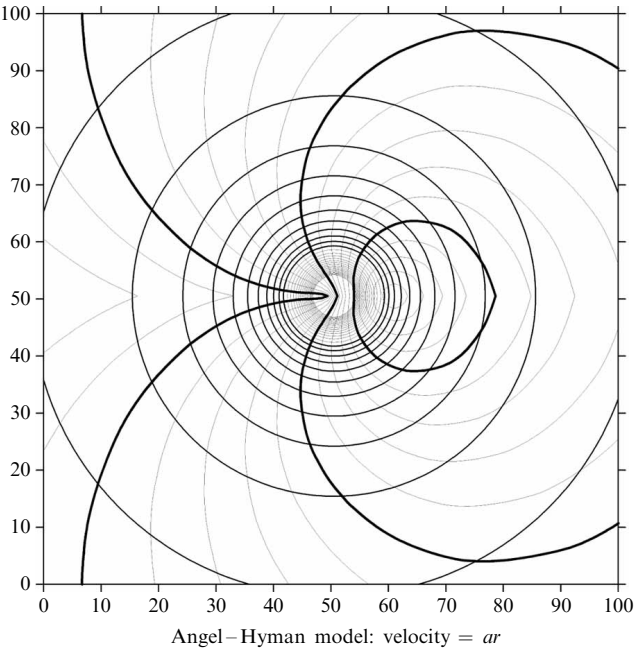


Figure 15. Distance isolines and velocity field—symmetric model.

the solution for a nonsymmetric case (an elliptical velocity field) to demonstrate the power of the technique to tackle problems that are analytically intractable.

The first case, shown in figure 15, may be compared with the analytically derived results and diagram produced by Angel and Hyman (1977, pages 15 and 159), which it matches very closely. In this example the velocity field is centred on (50 50) and is radially symmetric. The point (50, 60) is taken as the origin of journeys (slightly east of the city centre) and isolines extend around this point in all directions. Least travel time

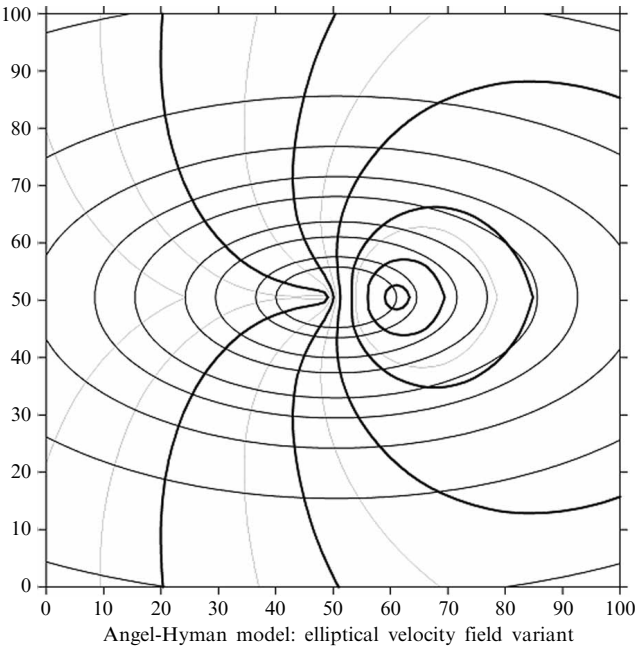


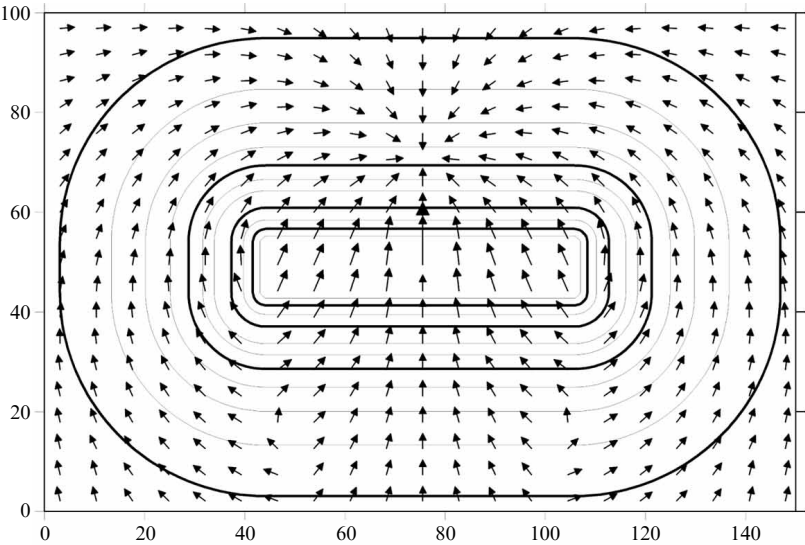
Figure 16. Distance isolines and velocity fields—symmetric model.

paths can be computed from the origin to all locations by drawing lines orthogonal to the isolines or using path vectors as described earlier.

The elliptical case (figure 16) has a velocity field shown as ellipses. The solution in this instance can be seen as a systematic distortion of the symmetric case, with isolines and associated shortest paths distorted by the asymmetric field. This example remains somewhat abstract, and as a first step towards development of a more realistic model we may construct a far more complex, composite velocity field model. In the example shown in figure 17(a) a composite congestion field has been constructed which I refer to as a *marquee*: the field is shown by the isolines and has a constant mean velocity region in a large part of the city centre which has high congestion and low speed (for example, 10 kph), surrounded by uniformly increasing speeds (decreasing congestion) from 10 to 55 kph at the city edges and beyond. The model is still simplistic, but is far closer to a real city such as London, than those previously discussed. The arrows show the expected least-time traffic flows from all points to a point north of the central region.

Figure 17(b) shows an outline of London, with three traffic cordon ‘rings’ shown in pale grey and marked with arrows—these are the lines across which government traffic surveys are conducted. The darker lines show the paths of least time routes from a point due north of the city centre in various directions based upon the model in the previous diagram. A more realistic model of London’s traffic could be constructed by using these overall concepts, with the velocity field modified to match recorded values, incorporating real-time traffic data for various times of the day, and allowing for key routes and special zoning (for example, the new central Congestion Charging Zone).

The LCDT procedure can also be applied to multiple weighted problems, such as that described in figure 8. In this case the optimal intermediate point, P' , will almost always be different to that found in figure 8 but will lie inside or at a vertex of the geodesic triangle constructed from the three initial points. The solution point, P' , can then be used as the origin and a new LCDT performed to determine the isolines and geodesics in relation to this point.



(a)



(b)

Figure 17. Isolines and fastest routes in a composite velocity cityscape.

Notes and conclusions

In applying standard and modified DTs of the kinds described in this paper the following issues must be considered:

- (1) If a standard form of DT is used (that is, there are no variations in cost or topography nor any obstacles) a predetermined number of passes of the mask will be required (one forward and one backwards pass in the case of the simplest DTs).
- (2) if the conditions in (1) do not apply, the solution procedure requires iteration to ensure convergence of the solution values. Experimental evidence suggests that for most problems two iterations suffice (that is, a total of four passes using the basic DT algorithm), but for some up to six or occasionally more iterations may be required. Computations based on the VTDT algorithm indicate that a larger number of iterations may be required for this procedure. No theoretical analysis of convergence has been carried out at this time and this remains as an area for further research.

(3) Solutions will not converge to regions that are unreachable, that is, to regions which are behind impermeable barriers, very high cost zones and/or sections of street networks not connected to the main area of study.

(4) If the size of the sample lattice (or subsections of this lattice that are surrounded by barriers) is not greater than the DT mask size, some or all distance values may not be updated and thus the procedure may be invalid.

We may conclude from the results and examples provided in the preceding sections that the various DT procedures described provide a simple, powerful, and extensible set of tools for spatial analysts, urban planners, and decisionmakers. These procedures include handling a wide variety of incremental Euclidean distance problems (such as buffering and determination of Voronoi regions), together with a range of least cost/time problems, incorporating spatial constraints such as obstacles and no-go regions, and weighted multicriteria problems. We have also shown that related DT methods can be used to determine geodesics on physically variable surfaces. Such procedures are in addition to the application of DTs to problems of 2D and 3D image processing and visualisation in spatial analysis and mapping. These various methods may be applied separately or in combination, and are ideally suited to implementation within current GIS software packages.

References

- Angel S, Hyman G, 1977 *Urban Fields* (Pion, London)
- Batty M, DeSyllas J, Duxbury E, 2002, "The discrete dynamics of small-scale events: agent-based models of mobility in carnivals and street parades", WP 56, Centre for Advanced Spatial Analysis, University College London, London
- Borgefors G, 1986, "Distance transformations in digital images" *Computer Vision, Graphics, Image Processing* **34** 344–371
- Borgefors G, 1989, "Distance transformations on hexagonal grids" *Pattern Recognition Letters* **9** 97–105
- Butt M A, Maragos P, 1998, "Optimal design of chamfer distance transforms" *IEEE Transactions on Image Processing* **7** 1477–1484
- Cuisenaire O, Macq B, 1999, "Fast and exact signed Euclidean distance transformation with linear complexity" *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing* **6** 3293–3296
- Douglas D H, 1994, "Least cost path in GIS using an accumulated cost surface and slope lines" *Cartographica* **31** 37–51
- Kiryati N, Szekely G, 1993, "Estimating shortest paths and minimal distances on digitised three-dimensional surfaces" *Pattern Recognition* **26** 1623–1637
- Leymarie F, Levine M D, 1992, "A note on fast raster scan distance propagation on the discrete rectangular lattice" *Computer Vision, Graphics, Image Processing* **55** 84–94
- Okabe A, Boots B, Sugihara K, Chiu S N, 2000 *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams* (John Wiley, Chichester, Sussex)
- Ratti C, 2001 *Urban Analysis for Environmental Prediction* unpublished PhD thesis, Department of Architecture, University of Cambridge, Cambridge

Appendix

Sample pseudo-code extracts for 5×5 distance transforms

$DT(,)$ is an array of lattice values, initialised to a large number, for example, 9999, greater than the maximum distance or cost distance that will be generated by the algorithm; $DT(n,m) = 0$ is a set of 1 or more source points (for example, a single point or set of points); $xdim$ = number of rows $- 2$; $ydim$ = number of columns $- 2$; $LDM(,)$ is local distance matrix (mask entries), with values $a1$, $a2$, and $a3$ (integer or fractional). For example, $a1 = 2.2062$, $a2 = 1.414$, $a3 = 0.9866$ or $a1 = 11$, $a2 = 7$, $a3 = 5$; $DX(,)$ is an array of row movements and $DY(,)$ an array of column movements, identifying the position in the mask to select.

In these examples, p is a scaling factor representing the lattice size (for example, for a 25 m DEM it would be 25), and $DEM(,)$ is a topographical adjustment array (for example, DEM values), that is, if $p = 1$ and $DEM(,) = \text{constant}$ then $d1 = d + LDM(k)$, and the transform is a standard DT. $COST(,)$ is an array of generalised costs or velocity field values.

Algorithm: Define mask, then scan DT array, adjusting the distances until least local distance is selected; note that forward scan starts from $row + 2$ $col + 2$ and similarly for backwards scan, to allow for mask size.

/ Forward scan:

Data: $LDM(0)$ to $LDM(8) = [a1; a1; a1; a2; a3; a2; a1; a3; 0]$

Data: $DX(0)$ to $DX(8) = [-2; -2; -1; -1; -1; -1; -1; 0; 0]$

Data: $DY(0)$ to $DY(8) = [-1; 1; -2; -1; 0; 1; 2; -1; 0]$

increment $i = 2$ to $xdim$ step $+1$

increment $j = 2$ to $ydim$ step $+1$

$d0 = DT(i, j)$

increment $k = 0$ to 8 step $+1$

$r = i + DX(k)$

$c = j + DY(k)$

$d = DT(r, c)$

/ if standard DT use

$d0 = \min(d + LDM(k), d0)$

/ else use one of the alternatives below:

/ if variable topography problem use following lines (or simplified variant):

$s = LDM(k) * p$

$t = DEM(r, c) - DEM(i, j)$

$s1 = s * s$

$t1 = t * t$

$s2 = \text{square-root}(s1 + t1)$

$d1 = d + s2$

$d0 = \min(d1, d0)$

/ end of inclusion

/ if variable cost/velocity field problem use following lines:

$d0 = \min(d + LDM(k) * (COST(r, c)), d0)$

/ end of inclusion

```

    next k
      / if problem includes obstacles, code obstacles as XXXX
      / (greater than max distance) and include the following lines:
        if  $DT(i, j) < XXXX$  then
           $DT(i, j) = d0$ 
        end if
      / else use
         $DT(i, j) = d0$ 
      end if
  next j
next i

/ Backwards scan:
Data: LDM(0) to LDM(8) = [0; a3; a1; a2; a3; a2; a1; a1; a1]
Data: DX(0) to DX(8) = [0; 0; 1; 1; 1; 1; 1; 2; 2]
Data: DY(0) to DY(8) = [0; 1; -2; -1; 0; 1; 2; -1; 1]

increment i = xdim to 2 step -1
increment j = ydim to 2 step -1

/ subsequent code is as per forward scan, above. For nonstandard DT cases, iterate
/ until change in all  $DT(i, j) = 0$  or  $<$  preset small value; to track solution paths record the
/ values of  $DX(k)$  and  $DY(k)$  which are selected as the min values, storing the results in
/ arrays  $x(i, j)$  and  $y(i, j)$  or in an additional dimension of  $DT(i, j)$ 

```