Research Article: Extensions to least-cost path algorithms for roadway planning

Jay Lee

International Journal of Geographical Information Science

INT. J. GEOGRAPHICAL INFORMATION SCIENCE, 2003
VOL. 17, NO. 4, 361–376

Research Article

Extensions to least-cost path algorithms for roadway planning

CHAOQING YU
Department of Geography, Pennsylvania State University, State College, PA 16802, USA

JAY LEE and MANDY J. MUNRO-STASIUK
Department of Geography, Kent State University, Kent, OH 44242-0001, USA

(*Received 15 August 2001; accepted 1 July 2002*)

Abstract. Finding a least-cost-path in a raster data format is a useful function in geographical information systems. However, existing algorithms are often inadequate for practical roadway planning. This paper improves conventional algorithms by including the considerations of spatial distances, anisotropic costs and the presence of bridges and tunnels in the paths. This new algorithm is implemented in JAVA to run with actual remote sensing and DEM data. The experimental results show that this approach produces realistic least-cost paths for practical roadway planning.

1. Introduction

Finding least-cost paths is a useful application in Geographical Information Systems (GIS). This topic has also received much attention in many other areas such as cartography, artificial intelligence, civil engineering and computer science. Potential uses include travel planning, military activities, road construction, irrigation systems, pipeline routing, and many other applications.

In order to improve existing algorithms for finding least-cost paths that are based on vector surfaces (Mitchell *et al.* 1987, Berg and Kreveld 1997, Varadarajan and Agarwal 2000), we present here an algorithm specifically designed for roadway planning that is suitable for use with raster data. Specifically, we consider in the algorithm spatial distances, anisotropic costs and the use of bridges and tunnels in the paths.

To find a least-cost path between a starting point and a destination point in a given gridded digital elevation model (DEM), there are two major steps. The first step is to create an accumulated cost surface with respect to all relevant cost factors. Dijkstra's (1959) shortest path algorithm is widely used to create a path on an accumulated cost surface. The second step is to construct the least-cost path with slope-tracing lines on the accumulated cost surface (Warntz 1957, Douglas 1994). Alternatively, the back-link mechanism (Xu and Lathrop 1995) can be used to

International Journal of Geographical Information Science
ISSN 1365-8816 print/ISSN 1362-3087 online © 2003 Taylor & Francis Ltd
http://www.tandf.co.uk/journals
DOI: 10.1080/1365881031000072645

connect the departure and destination locations when forming a least-cost path on the accumulated cost surface.

Recent progress in studies of least-cost paths has shown great promise in making realistic applications. These include constructing paths with constraints to avoid terrain obstacles (Mitchell 1988) by considering directional differences or anisotropy of the terrain surfaces (Zhan *et al.* 1993, Xu and Lathrop 1994, 1995, Collischonn and Pilar 2000), and improving the accuracy and efficiency of the resulting paths (Douglas 1994, Solka *et al.* 1995, Vörös 2001). In addition, the least-cost-path algorithms have been fine-tuned to solve real world problems. For example, Feldman *et al.* (1995) utilized least-cost paths for routing oil pipelines near the Caspian Sea. Stefanakis and Kavouras (1995) charted the shortest sea courses under various models of travel costs. Lee and Stucky (1998) computed hidden paths, scenic paths, strategic paths and withdrawn paths using visibility as the main cost factor.

Even with these studies, there are few successful applications of least-cost-path algorithms for roadway planning related to vehicle navigation on complex terrain surfaces. One important reason is that most existing algorithms are based on tracing only adjacent nodes in weighted networks or neighbouring cells in grids when constructing the paths. As a result, bridges or tunnels that are frequently used in reality cannot be automatically considered because they typically connect non-adjacent points to shorten the distance of paths. Together with using realistic measures of spatial distances and anisotropic costs, this paper introduces a new algorithm in a raster data format for constructing realistic least-cost paths.

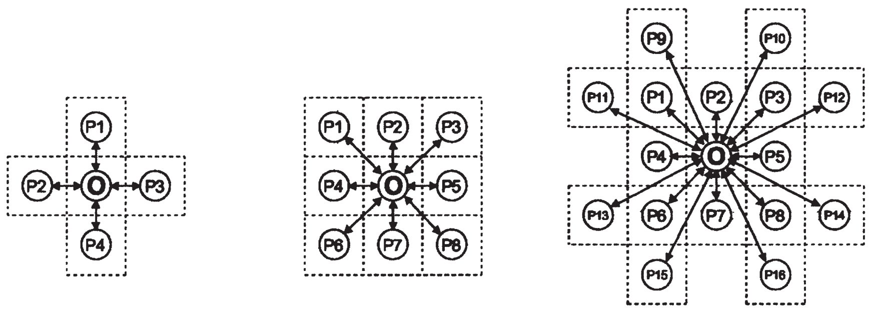
In the following sections we first discuss the conventional algorithms currently used in GIS for finding least-cost paths. We then discuss the concepts of spatial distance, anisotropic accumulated-cost surface, followed by a discussion of the new algorithm for determining bridges and tunnels. Results of testing the new algorithm on two sample regions are then presented followed by discussion and conclusions.

2. Conventional algorithms and improvements

2.1. *Virtual network in grids*

Dijkstra's algorithm is one of the most widely used algorithms for finding least-cost paths. It was designed for tracing the shortest path in a network with nodes connected by weighted links. To use this algorithm in a GIS grid data layer, a virtual network can be constructed under two conditions: (1) the centres of each grid cell serve as the nodes in the network, and (2) the connections between the neighbouring-cell centres act as the links of the network (Xu and Lathrop 1994, 1995). For example, Stefanakis and Kavouras (1995) built their network using direct, indirect and remote neighbouring cells. A more popular way is to connect a fixed number of neighbouring cells in a grid into a network. These have various definitions of neighbourhood types including Rook's pattern (4-cell), Queen's pattern (8-cell) and Knight's pattern (16-cell) (Goodchild 1977, Bemmelen *et al.* 1993, Xu and Lathrop 1994, 1995). Figure 1 shows the three neighbourhood types.

Once the virtual network is established, Dijkstra's algorithm can be applied to create the accumulated-cost surface, which is the sum of the weighted links between successive pairs of cells. The strategy of this algorithm is to examine all of the nodes adjacent to the most recently evaluated node on the path. Those closest to the starting node are evaluated first, and the most distance nodes are evaluated last (Dijkstra 1959, Weiss 1995, Xu and Lathrop 1995).



(*a*) Rook`s pattern (*b*) Queen`s pattern (*c*) Knight`s pattern

Figure 1. Three types of neighbouring patterns in raster data formats.

2.2. *Isotropic accumulated-cost surfaces*

An accumulated cost surface is also a grid, in which the value of every cell is the minimum accumulated cost from the cell where the least-cost path starts. It is usually based on a cost surface where each of its cell values represents the cost per unit distance of passing through the cell (Douglas 1994). The costs may be expressed with time as dollar cost, distance, or risk (Collischonn and Pilar 2000). If these costs are uniform for all directions and over the entire surface, the surface is called an isotopic surface (Xu and Lathrop 1995). In an isotropic cost surface a unit distance corresponds to the horizontal width of a cell in the grid. If an accumulated cost surface is derived from the isotropic cost surface, the surface can be called an isotropic accumulated cost surface.

To calculate the isotropic costs, there are two equations for Queen's pattern. Equation (1) is used to compute the accumulated costs of four direct connections (figure 1(*b*)).

$$CC_{(O,P_i)} = \frac{(C_O + C_{P_i})}{2}\mu + CC_O \qquad \text{where } i = 2, 4, 5 \text{ and } 7 \tag{1}$$

Equation (2) is used to compute the four diagonal accumulated costs:

$$CC_{(O,P_i)} = \frac{(C_O + C_{P_i})}{2}\sqrt{2}\,\mu + CC_O \qquad \text{where } i = 1, 3, 6 \text{ and } 8 \tag{2}$$

where $CC_{(O,P_i)}$ is the accumulated cost from cell O to cell P_{P_i}. C_O and $CC_{(P_i)}$ are the costs for crossing the individual C_O and CC_{P_i} cells respectively; μ is the width (or resolution) of each cell; CC_O is the accumulated cost at cell O. The above two equations provide the conceptual foundation for the new algorithm to be discussed later.

A back-link cell is the previous cell used for calculating the accumulated cost of the current grid cell. For example, cell O could be the back-link cell of P_1 in figure 2(*b*).

2.3. *Anisotropic accumulated-cost surfaces*

In reality, isotropic accumulated cost surfaces rarely exist. For example, when terrain surfaces are complex, slopes in different directions are not constant. For roadway planning, these directional differences are important, and it is therefore

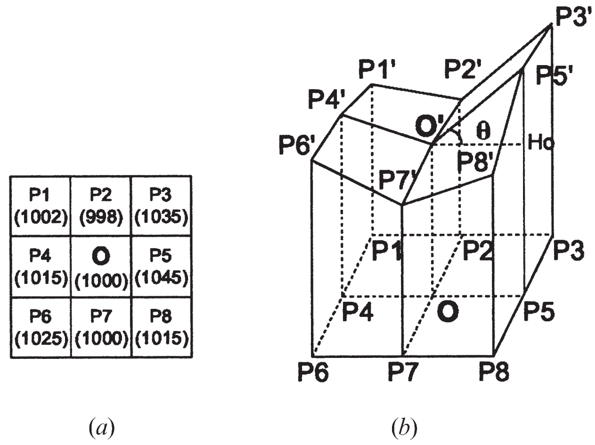


Figure 2. A digital elevation model (*a*) and its geometric representation (*b*).

necessary to use anisotropic accumulated cost surfaces. Zhan *et al.* (1993) introduced a directional path distance model to perform distance mapping. Xu and Lathrop (1994 and 1995) discussed the problems of anisotropic spread using elliptical patterns. Collischonn and Pilar (2000) showed a more realistic least-cost path algorithm for roads and canals. But they did not explain the details of calculating anisotropic accumulated costs, and they did not use real cost functions.

2.3.1. *Spatial distance*

To calculate the anisotropic accumulated costs from one cell to another, the distance between two given cells is called the spatial distance, which is defined as the distance between the centres of these two cells on the terrain surface. The spatial distance will not always equal the cell width because it varies with changing direction and slope. For example, on the sample DEM (figure 2) the spatial distance $OP5$ does not equal $OP3$ because the distance between two cells that are diagonally adjacent is longer than the distance between two cells that are directly adjacent. Importantly, slope will also change the distance between cells. For example, the cell distance $O'P5'$ does not equal to $O'P7'$ because the slope angle between O' and $P5'$ is larger than that between O' and $P7'$ (figure 2(*b*)). Thus, when actual distances are projected onto the horizontal plane they will be different from the straight line distance between two cells. Because the straight line (2D) distances are used to calculate the accumulated cost from start to end, small slopes between single cells could accumulate in a substantial distance error. Therefore, using spatial distance (in 2.5D) instead of 2D distance will improve the accuracy of the least-cost paths.

In this paper, three equations have been created for calculating the spatial distances in the Rook's, Queen's and Knight's patterns (equations (3), (4) and (5)). For the Rook's pattern and direct connections in the Queen's pattern (figure 2(*b*)), the spatial distances are:

$$D'_{(O,P_i)} = \sqrt{\mu^2 + (H_{P_i} - H_O)^2} \qquad \text{where } i = 2, 4, 5 \text{ and } 7 \tag{3}$$

For the diagonal connections in Queen's and Knight's pattern (figures 2(*b*) and (*c*)) the distances are given by:

$$D'_{(O,P_i)} = \sqrt{2\mu^2 + (H_{P_i} - H_O)^2} \qquad \text{where } i = 1, 3, 6 \text{ and } 8 \tag{4}$$

The eight outer diagonal connections of Knight's patterns (figure 2(*c*)) are:

$$D'_{(O,P_i)} = \sqrt{5\mu^2 + (H_{P_i} - H_O)^2} \qquad \text{where } i = 9\text{–}16 \tag{5}$$

where H_{P_i} is the height of cell P_i ($i = O, 1, \ldots, 16$); $D_{(O,P_i)}$ is the 2D distance from cell O to P_i and $D'_{(O,P_i)}$ is the spatial distance from O' to P'_i; μ is the horizontal width of each cell, which is always 30 m in 7.5 minute DEMs.

2.3.2. *Slope calculation*

For roadway planning, slope and the slope directions are important factors for weighing construction and travelling costs. In this paper, we use two sets of input data: land cover classes derived from Landsat ETM+ satellite imagery and terrain surfaces derived from DEMs. To compute anisotropic accumulated costs, slopes can be calculated from DEMs. In figure 2(*b*), assuming the distance is μ, the slope of each of the eight directions can be calculated from the elevations. The slope of the direct connections can be stated as:

$$\theta_i = \arctan\left(\frac{H_{P_i} - H_O}{\mu}\right) \qquad \text{where } i = 2, 4, 5 \text{ and } 7 \tag{6}$$

For the diagonal connection the slope angle is:

$$\theta_i = \arctan\left(\frac{H_{P_i} - H_O}{\sqrt{2}\,\mu}\right) \qquad \text{where } i = 1, 3, 6 \text{ and } 8 \tag{7}$$

Similarly, for one of the eight outer diagonal connections of Knight's pattern (figure 1(*c*)) the slope is:

$$\theta_i = \arctan\left(\frac{H_{P9} - H_O}{\sqrt{5}\,\mu}\right) \qquad \text{where } i = 9\text{–}16 \tag{8}$$

In figure 1, if $\mu = 30$, the slope angle from cell O to each of the eight directions can be calculated using equations (6) and (7). For example, from O' to $P7'$, the slope angle is 0, because there is no elevation change. But from O' to $P5'$, the slope is high at an angle of 56°. For a normal vehicle, it is easy to move from cell O' to $P7'$, but it is almost impossible to move from cell O to $P5'$. Therefore, for a grid cell, costs should be assigned different values based on the degree of slope. If these types of directional differences are considered in calculating the accumulated-cost surface, the resulting accumulated-cost surface can be called an anisotropic accumulated-cost surface. The significance of using an anisotropic accumulated-cost surface is that it restricts the direction of a vehicle's movement according to the true characteristics of the terrain surface. For example, the prospective optimal path should be a curve instead of a straight line if the overall slope is too steep for a regular vehicle. This is because the only way to overcome a steep slope is with zigzags (or switchbacks) at the expense of increasing path length.

2.3.3. *Anisotropic accumulated cost calculation*

Using the spatial distance in equations (3), (4) and (5) and assigning weights as costs according to their slope angles (equations (6), (7) and (8)), the methods for computing anisotropic accumulated costs have been developed in this paper using the following equations:

(*a*) For the four direct connections:

$$CC_{(O,P_i)} = D'_{(O,P_i)}\left(\frac{C_O + C_{P_i}}{2} + \theta_i \text{ weight}\right) + CC_{(O)}$$

$$CC_{(O,P5)} = \sqrt{\mu^2 + (H_{P5} - H_O)^2}\left(\frac{C_O + C_{P5}}{2} + \arctan\left(\frac{H_{P5} - H_O}{\mu}\right)\text{weight}\right) + CC_{(O)}$$

$$\text{where } i = 2, 4, 5 \text{ and } 7 \qquad (9)$$

(*b*) For the four direct connections:

$$CC_{(O,P3)} = \sqrt{2\mu^2 + (H_{P5} - H_O)^2}\left(\frac{C_O + C_{P3}}{2} + \arctan\left(\frac{H_{P3} - H_O}{\sqrt{2}\,\mu}\right)\text{weight}\right) + CC_{(O)}$$

$$\text{where } i = 1, 3, 6 \text{ and } 8 \qquad (10)$$

(*c*) The outer diagonal connections of Knight's pattern (figure 2(*c*)) is exemplified by the case for *P*9:

$$CC_{(O,P9)} = \sqrt{5\mu^2 + (H_{P9} - H_O)^2}\left(\frac{C_O + C_{P1} + C_{P2} + C_{P9}}{4} + \arctan\left(\frac{H_{P9} - H_O}{\sqrt{5}\,\mu}\right)\text{weight}\right) + CC_{(O)} \qquad (11)$$

For *P*10–16 comparable equations can be developed.

In equations 9–11, H_{P_i} is the height of Point P_i ($i = O, 1, \ldots, 16$); θ, arctan-$(H_{P_i} - H_O/\sqrt{2}\,\mu)$ is the degree of the slope in the direction of diagonal connection or direct connection; 'weight' is the contribution of this slope on the cost; C_{P_i} is the isotropic cost in the cell p_i; $CC_{(O,P_i)}$ is the accumulated cost from cell O to P_i; $D'_{(O,P_i)}$ is the spatial distance; and μ is the horizontal width of each cell; and $CC_{(O)}$ is the accumulated cost of cell O.

The equations (9), (10) and (11) are the core formulae to calculate the accumulated anisotropic cost surfaces for neighbouring patterns.

2.4. *The Smart Terrain (ST) Algorithm*

2.4.1. *Methods for finding bridges/tunnels*

Although the outlined anisotropic accumulated cost surfaces can produce more realistic least-cost paths (Collischonn and Pilar 2000) for roadways, there are still shortcomings because the accumulated costs are calculated strictly from adjacent network neighbours. Usually, bridges and tunnels are necessary when roadways need to cross over rivers or obstacles. Since bridges or tunnels normally link non-adjacent network neighbours, the algorithm for finding least-cost paths needs to be modified to solve this problem.

In improving existing algorithms for this purpose, the difficulty is in finding out how to effectively determine if a bridge or tunnel should be added to a least-cost path between a given cell and any of the other cells in the grid. For our purpose, two characteristics of bridges and tunnels should be noted: (1) most of them are horizontal; and (2) they usually connect pairs of two non-adjacent points with straight lines. These two characteristics also help to reduce the total cost of building a bridge or tunnel.

To determine where bridges or tunnels should be located in calculating realistic least-cost paths, we developed a new algorithm that is based on the concept of contouring. This takes advantage of a contour line, guaranteeing that all points on it have the same elevation. It is apparent that connecting a pair of non-adjacent points on a contour line can form a horizontal straight line (Zhu *et al.* 2001).

To assist the illustration of the new algorithm, figure 3 shows a contour line with an elevation of 1000 m. The grey cells all lie on this contour line. Note that non-adjacent cells A, C and D are on the same contour line. In figure 3, assume that A is a new starting cell. The major steps of the new algorithm are to connect cell A with all of its non-adjacent cells along the contour line, such as AC or AD, to determine whether these connections are bridges or tunnels and to calculate the costs of all possible bridges or tunnels connecting with A.

The new algorithm will need to be capable of performing four tasks. The first is the determination of whether the candidate cells are adjacent to cell A or not. If they are adjacent, the accumulated costs will be calculated from equations (9), (10) or (11). Otherwise, the algorithm will continue tracing the rest of the cells on the contour line.

Second, the algorithm will determine if a bridge or tunnel should be added between cell A and any of the non-adjacent cells on the contour line. Because a bridge or tunnel connects only two non-adjacent cells, one method is to count the number of intersections between potential connections and the contour line. If the straight connection between cell A and one non-adjacent candidate cell intersects the contour line only twice (this includes the start and end points), this connection may be a bridge or tunnel. In figure 3, AC can be connected with a bridge or tunnel. But AD cannot, because there are three intersections between AD and the contour line. Here, for the purpose of adding bridges and tunnels, AC can be called a *true connection*, and AD a *false connection.*

Thirdly, the algorithm has to distinguish whether a true connection should be either a bridge or a tunnel. One way to do this is to pick up an additional cell, such as B in figure 3, which is in the middle of the true connection AC, and check to see if its elevation is greater or less than the elevation of the contour line. If cell B's

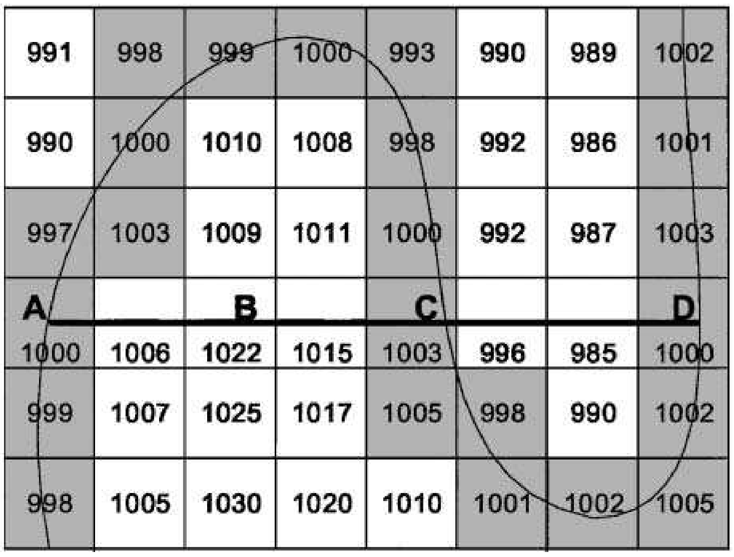


Figure 3. A contour line for determining bridges and tunnels.

elevation is less than the contour line's elevation, a bridge is needed. If it is higher, a tunnel is needed. In figure 3, B's elevation is 1022 m, which is greater than 1000 m, thus AC could be a tunnel. Once a bridge/tunnel is determined, its total cost can be calculated according to its unit costs and length. When all of the non-adjacent cells on the contour line have been checked, the algorithm will mark the current starting cell A to prevent it from further calculations, and pick up an unmarked cell with the smallest accumulated cost as a new starting cell.

Finally, when there are no more unmarked cells left, an accumulated-cost surface is created, and the least-cost path with bridges and tunnels can be found by tracing the back-link cells from end to start. This algorithm for dealing with zigzag paths and bridges and tunnels on a 3D to 2.5D terrain surface is named the 'Smart Terrain' (ST) Algorithm.

2.4.2. *Extracting the cells that intersect with contour lines*

Extracting cells intersecting contour lines (the grey ones in figure 3) from the DEM is an important procedure in the ST algorithm. To determine if a cell is on a specific contour line, three possible cases should be considered: is the elevation of the cell equal to, greater than or less than the elevation of this contour line. For the first case, when the two elevations equal each other, the cell intersects with the contour line. For the second and third cases, a Queen's pattern is used to determine whether the cell intersects the contour line or not. Figure 4 illustrates how to determine if cell *O* intersects a contour line, which is 1000 m.

For the second case, when cell *O*'s height is greater (1005 m in figure 4(*a*)) than the contour line, the contour line will intersect the cell only when the following condition is satisfied: the elevation of contour line (1000 m) is greater than the mean of cell *O*'s elevation (1005 m) and one of its neighbouring elevations. For example, the mean elevation of *O* and *P*2 in figure 4(*a*) is 999.5 m, which is the middle (*M* in figure 4(*a*)) of *OP*2. Because 1000 m is greater than 999.5 m, cell *O* intersects the contour line.

Similarly, for the third case, when the cell's height is less (995 m in figure 4(*b*)) than the elevation of the contour line, the contour line intersects the cell only when elevation of the contour line is less than the mean of cell *O*'s elevation (995 m) and one of its neighbouring elevations. For example, the mean elevation of *O* and *P*6 in figure 4(*b*) is 1002 m, which is also the middle (*M* in figure 4(*b*)) of *OP*6. Because the contour's elevation is less than 1002 m, cell *O* intersects the contour line.

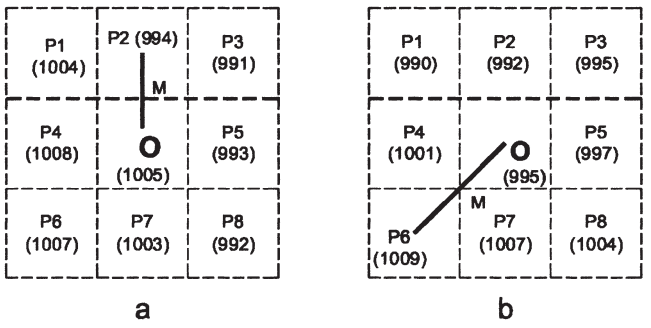


Figure 4. Two possible conditions to determine whether cell *O* intersects a contour line (1000 m) when *O*'s elevation is not equal to 1000 m.

Once it is determined that cell O intersects the contour line using one of the eight neighbours, the calculations for the rest of cell O's neighbours are ignored. However, if cell O does not intersect the contour line, all eight neighbours will be visited. In this paper, contour lines are prepared as a part of the inputs before calculating the accumulated-cost surfaces.

2.4.3. *Determining true or false connections*

As explained in figure 3, the method for determining a true or false connection is to count the number of intersections between the connection and the contour line. The strategy of counting the intersections is to count how many cells are on the line connecting two cells (e.g. AC in figure 3) and the contour line. Therefore, it is necessary to extract all the cells intersecting the connection AC. Because all grid cells are equal in size, once the coordinates of cell A and C are known, the gradient of AC can be determined. Thus the intersecting cells (the grey ones in figure 5) can be found by increasing the values of coordinates according to the gradient from A to C or from C to A. After all of the intersecting cells of AC are found, the next step is to count how many cells are on the connection (AC) and the contour line. If the number is two, it is a true connection. Otherwise, it is a false one. To save running time, it is not necessary to compare all cells on the contour line. Instead, the range can be restricted within Y_A and Y_C and/or X_A and X_C.

3. Testing the algorithm

3.1. *Data and tools*

The ST algorithm was tested on two small mountainous regions in Venango County of Pennsylvania, USA. The input data included land cover and 7.5 minute DEMs (1:24 000) for this region. The land cover values were derived from a Landsat ETM+ image (path 18, row 31, 5 September 2000) that was downloaded from http://www.ohioview.org. The DEM data were downloaded from http://www.usgs.gov. The resolution for both of the input data is 30 m. For programming, JAVA2 SDK (standard edition v.1.3) was used. PCI Geomatica was used for satellite image classification. ArcInfo and ArcView were used for data visualization and mapping.

3.2. *Results and analysis*

In our implementation, the major parameters affecting the cost of the path include weighted land cover, slope, cost of bridge/tunnels, and spatial distance. From highway

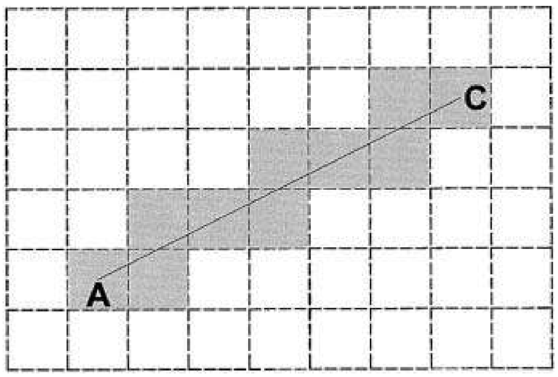


Figure 5. The intersecting cells (grey) on AC.

engineering literature, O'Flaherty (1967) estimated that the average cost for road construction in urban areas is at least 10 times higher than that of rural farmland. Moreover, the costs of constructing bridges are about 12 to 15 times higher than that of levelled highways. With regards to slope on highways, when the grade is greater than 7% (slope of 7 in 100, or 4.0°), the speed of passenger vehicles is greatly affected. Sloped highways are, of course, even less suitable for vehicles with heavy loads. Only a few highway agencies have permitted grades as steep as 12% (or 6.8°). For special situations, the maximum grades in highway can be up to 16% (or 9.1°), but this needs to be carefully considered in the light of safety (Oglesby and Hewes 1963, O'Flaherty 1967). Because the least-cost paths in this paper are not just designed for highways, the weights of the parameters have been adjusted from the highway standards.

In our experiments, we have tested how these parameters affect the results of the least-cost path between two particular cells. Furthermore, to examine the reliability, the algorithm has been tested through changing the departure and destination of the roadway and experimental regions.

3.2.1. *Isotropic paths without considering slopes and bridges/tunnels in Queen's pattern*

The first test experimental region is located in the south of the Venango County, Pennsylvania. The actual size of this region is $1.8 \times 2.7\,\text{km}^2$ (figure 6(*a*)). Assuming the slopes and bridges/tunnels are not included in the cost surface, finding the least-cost path is based on two parameters: the cost derived from land cover and the spatial distances of the path. The weights for the parameters we assigned for our experiments are shown in table 1. The result in the Queen's pattern under this case is illustrated in figure 6(*a*) from location A to B, the path cannot cross rivers in this case. It tends to meander along the land with the smallest land costs and follow the shortest spatial distances.

3.2.2. *Isotropic paths with average slopes but without bridges/tunnels*

Conventional algorithms for finding least-cost paths are based on isotropic accumulated cost surfaces and bridges or tunnels are generally not considered. To approximate the actual practices in highway engineering, in table 2, the slope weights have been set up with critical values so that they would not allow a path to traverse any area with average slopes greater than 16°. Under these conditions, no path is found between A and B, because the average slopes close to B are always greater than 25°.

When the slope tolerance is increased as shown in table 3, a least-cost path is found between A and B. As shown in figure 6(*b*) the ST algorithm finds a path that avoids higher cost land covers such as urban areas and water. It always attempts to find the flatter regions such as on the top of hills, or at the bottom of valleys. When steep slopes are inevitable, the algorithm tries to find the directions with the shortest distance in the steep regions to reduce the accumulated costs. These directions have the steepest slopes (see the path close to B in figure 6(*b*)), which are always perpendicular to the contour lines.

3.2.3. *The Queen's anisotropic path without bridges/tunnels*

To compare the Queen's anisotropic paths with the conventional isotropic paths, all parameters are set exactly the same as in table 3. Figure 6(*c*) shows a least-cost

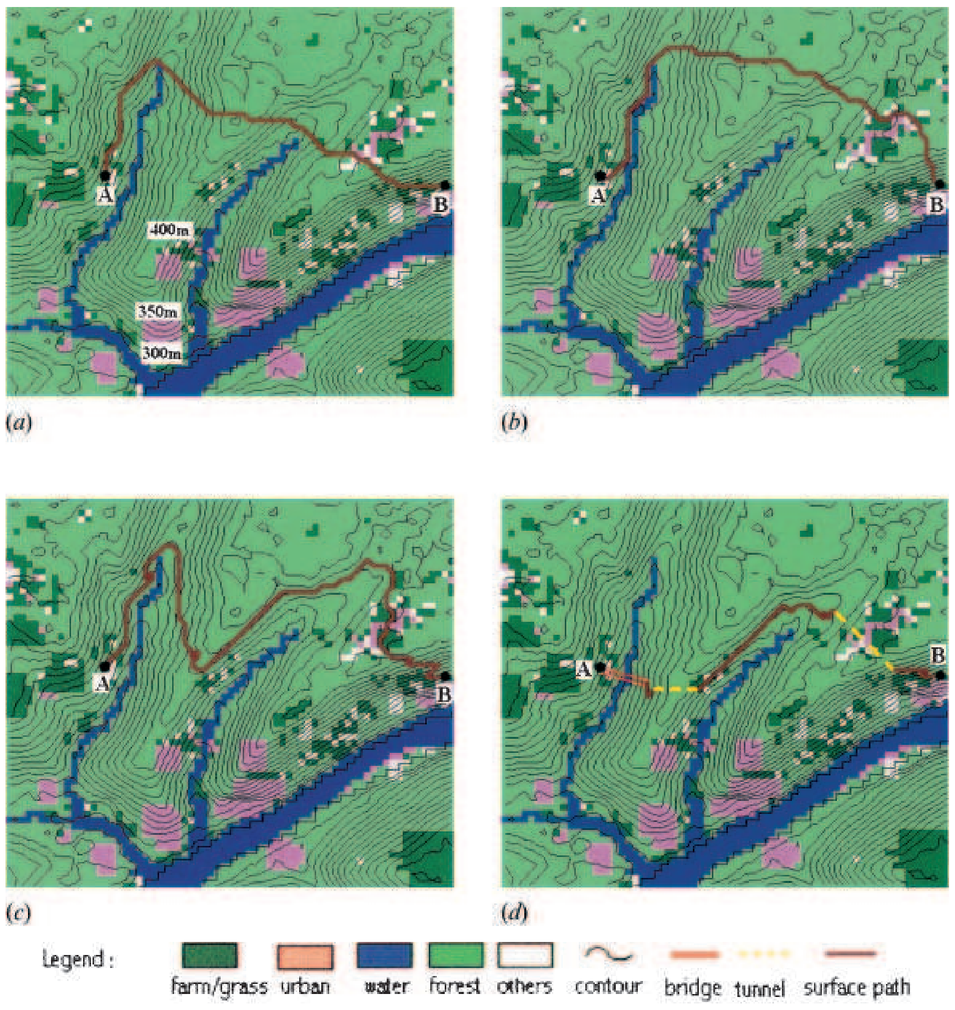
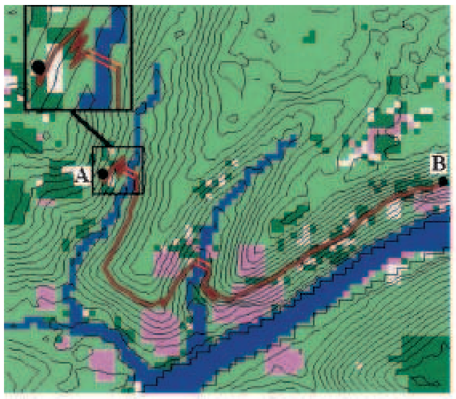


Figure 6.

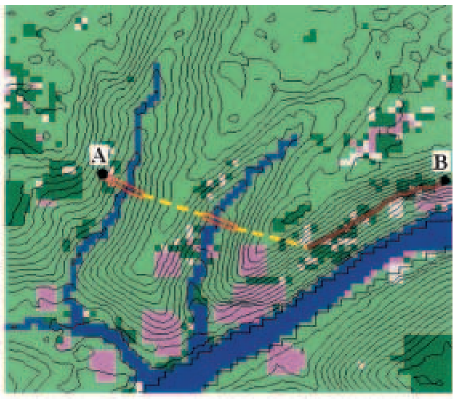
path on the anisotropic accumulated cost surface that is found between A and B using the Queen's pattern. Instead of keeping the path perpendicular to contour lines, it is now allowed to zigzag upslope to reduce the slope steepness and the overall length of the path (Collischonn and Pilar 2000). Alternatively, a path may follow (parallel to) the directions of a contour line to avoid zigzagging. However, it will inevitably lengthen the path.

3.2.4. *The Queen's and Knight's anisotropic path with bridges/tunnels*

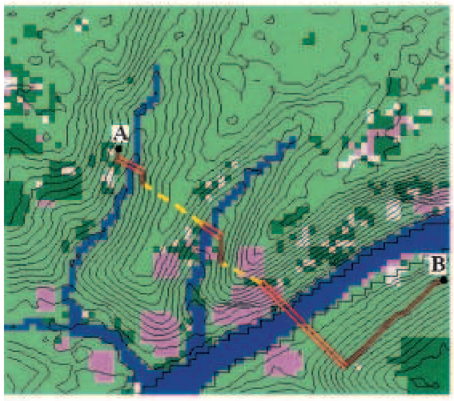
To compare the ST algorithm with conventional algorithms under the conditions of table 2, a critical standard of slope has been set up. The weights for bridges and tunnels have also been assigned as in table 4. Figure 6(*d*) shows the result of the least-cost path between A and B using Queen's anisotropic pattern. In this path, one bridge and two tunnels are created. Notice that the characteristics of reducing slope changes can also be seen in the figure. From the computed results, the accumulated cost from A to B in Queen's pattern is 22259.6 by using the parameters in table 4.



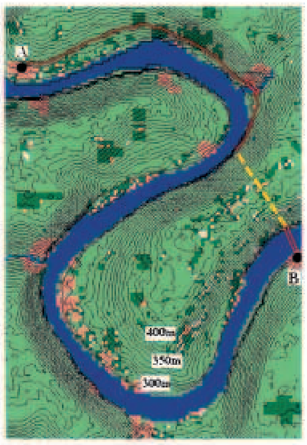
(*e*)



(*f*)



(*g*)



(*h*)

Table 1. Parameters without slopes or bridges and tunnels.

Land covers	Weights	Slopes	Weights	Bridge/tunnel	Weights
Farm/grass	1	0 ~ 90°	no	Bridge	no
Forest	2			Tunnel	no
Urban	10				
Water	∞				
Others	3				

Using the same parameters in table 4, figure 6(*e*) shows the result of the least-cost path between A and B using Knight's pattern. Although no tunnels are created, two bridges are created over the path. The enormous differences between the results in figure 6(*e*) and the results in figure 6(*d*) are because this algorithm has more choices in the Knight's neighbour pattern than in the Queen's pattern. A Knight's pattern can improve the accuracy of the least-cost paths, especially in regions with steep slopes. For example, because the slope near B is steep, it is difficult in the

Table 2. Parameters for conventional isotropic paths with average slopes (1).

Land covers	Weights	Average slopes	Weights	Bridge/tunnel	Weights
Farm/grass	1	0 ~ 3°	0	Bridge	no
Forest	2	3 ~ 6°	4	Tunnel	no
Urban	10	6 ~ 9°	8		
Water	∞	9 ~ 12°	20		
Others	3	12 ~ 16°	80		
		> 16°	∞		

Table 3. Parameters for conventional isotropic paths with average slopes (2).

Land covers	Weights	Average slopes	Weights	Bridge/tunnel	Weights
Farm/grass	1	0 ~ 3°	0	Bridge	no
Forest	2	3 ~ 6°	1	Tunnel	no
Urban	10	6 ~ 9°	2		
Water	∞	9 ~ 12°	4		
Others	3	12 ~ 16°	8		
		16 ~ 30°	16		
		30 ~ 45°	32		
		> 45°	∞		

Table 4. Parameters with bridges, tunnels and anisotropic slopes.

Land covers	Weights	Anisotropic slopes	Weights	Bridge/tunnel	Weights
Farm/grass	1	0 ~ 3°	0	Bridges	12
Forest	2	3 ~ 6°	4	Tunnels	12
Urban	10	6 ~ 9°	8		
Water	∞	9 ~ 12°	20		
Others	3	12 ~ 16°	80		
		> 16°	∞		

Queen's pattern to keep the neighbouring slopes small. Thus, tunnels are the best choices to reduce the high slope costs on the surface path. It is easier for the Knight's pattern to find smaller neighbouring slopes than for Queen's case. With this, the path can parallel the directions of the contour lines, minimizing slope changes. From the calculated results for figure 6(*e*), the accumulated cost of the Knight's pattern from A to B is 12059.6. This is approximately half the cost of the path in figure 6(*d*).

3.2.5. *The Knight's anisotropic path with smaller difference in land cover*

In table 5, the slope is kept the same as that in table 4. However, the differences among weights of the bridges and tunnels and individual land covers are reduced. In this case, the distance becomes critical. This is suitable for considering long-term costs, such as (1) when surface road construction cost is expensive; (2) when gasoline and other travelling costs are expensive; and (3) when maintenance costs of the roadways are high. Figure 6(*f*) shows the result of the least-cost path between A and B using a Knight's anisotropic pattern as an example. This is significantly different from the result in figure 6(*e*): two bridges and two tunnels are created, and

Table 5. Parameters with smaller difference in land-cover weights.

Land covers	Weights	Anisotropic slopes	Weights	Bridge/tunnel	Weights
Farm/grass	2	0–3°	0	Bridges	6
Forest	3	3–6°	4	Tunnels	6
Urban	6	6–9°	8		
Water	∞	9–12°	20		
Others	4	12–16°	80		
		$>16°$	∞		

their spans are large because of the low cost assigned to bridges and tunnels. Furthermore, the path has been kept as straight as possible to reduce the distance.

3.2.6. *The Queen's anisotropic path with different departure and destination cells*

To test the reliability of the ST algorithm, the departure and destination cells are changed, but the parameters are kept the same as those in table 4. Figure 6(*g*) shows that three bridges and two tunnels are found for the least-cost path using Queen's pattern.

3.2.7. *The Knight's anisotropic path in a different region*

To test the performance of the ST algorithm in different data sets, another small region ($2.79 \times 4.5\,\mathrm{km}^2$) in Venango County was chosen for an additional experiment. To allow direct comparison, the parameters are kept the same as those in table 4. Figure 6(*h*) shows that the resulting least-cost path has two bridges and one tunnel using the Knight's pattern.

4. Discussion and conclusions

Compared to the traditional algorithms, the ST algorithm produces more realistic least-cost roadways by using spatial distances, anisotropic accumulated-cost surfaces, and bridges and tunnels. On the basis of our test results, this algorithm is robust and practical. In this paper, both Queen's and Knight's patterns have been used to calculate the accumulated costs of neighbouring cells. According to the results, the Knight's pattern produces paths with lower total costs than those produced in Queen's pattern, especially on rugged terrain surfaces.

In most of the current commercial GIS software products, the spatial distances, anisotropic costs and bridges/tunnels have not been fully considered for functions of finding least-cost finding paths. The ST algorithm can be applied in these software products to improve roadway planning on more realistic terrains. Even though the concept of using spatial distances in the ST algorithm is new, it is simple to implement within current commercial GIS software. Moreover, the ST algorithm has been developed using existing GIS software packages so it is directly compatible with prevailing data structures. When adapted into commercial GIS software packages, the ST algorithm may need to be optimized to reduce running time for large distances.

The lesson we have learned from this paper is that the absolutely 'optimal' or 'least-cost' path does not exist. Only relatively 'optimal' paths can be found under particular conditions. For instance, this paper has shown that the settings of parameters can significantly change the results of least-cost paths. Because this paper focuses on discussion of the ST algorithm, the various methods for weighting different parameters have not been fully explored. For example, geological conditions are always important for road construction; the cost of a bridge/tunnel does not increase linearly with its length; and the cost of a bridge is also related to the height above the valley floor based on engineering and flood risks. This represents some possible extensions to the ST algorithm. If these differences of uphill and downhill slopes are considered (Berg and Kreveld 1997, Collischonn and Pilar 2000), the ST algorithm can also be used for other types of applications, such as irrigation systems.

The ST algorithm is not without limitations. Because the data in raster-based models are discrete in nature, they cannot completely represent the continuous surface of the real world. To improve the precision, the potential value beyond cell centres is needed. One way to achieve this is to add more cells to the existing neighbouring patterns (e.g. 32-cell pattern, Bemmelen *et al.* 1993), but it could miss the directional differences and add unnecessary difficulty when considering bridges/tunnels on anisotropic surfaces. Douglas (1994) introduced a method for isotropic surfaces based on calculating additional cross points on the edges of grid cells. But it is difficult to decide where the optimal cross points are in the case of an anisotropic surface. For example, when considering slopes, using the smallest slope gradient does not always produce the least-cost path. This is because the final result is dependent on the accumulated costs that might be affected by many other parameters as well. Using quadtrees is also a possible way to increase precision (Vörös 2001). However, there may be deviation errors if the quadtree cells are too large on isotropic surfaces (Bemmelen *et al.* 1993).

Because we only used Queen's and Knight's patterns without considering the potential values beyond the cell centres, the ST algorithm may produce a 'zigzag' form in the resulting least-cost path. If very sharp curves are not desired in railway or highway planning, a simple directional constraint can be added to the algorithm to limit it. Alternatively, the sharpness can be reduced through setting the maximum angle of turns. It should be noted, however, that potential errors may exist because of the limited choices in the Queen's or Knight's neighbourhood. For this reason, the ST algorithm may produce some bridges or tunnels in less than desirable locations because it is designed to avoid slope changes.

Sometimes, the bridges and tunnels on the least-cost paths do not need to be completely level. The ST algorithm currently assumes that all bridges and tunnels are completely horizontal. However, this limitation can be relaxed by expanding the checking mechanism (e.g. setting tolerances) so that some differences in elevation between the starting and ending cells of a bridge or tunnel are allowed.

Finally, it should be noted that there are other possible limitations of the ST algorithm because the algorithm does not take into account the curvature of terrain surfaces. To overcome this limitation, more refined definition is needed for measuring spatial distances with curvature. While mathematically possible, including such consideration will inevitably lengthen computation time of finding least-cost paths on terrain surfaces.

Acknowledgments

We wish to acknowledge the funding and support received from OhioView via NASA and USGS for assistance in data acquisition and analysis in preparation of this paper. We would also like to thank Scott Sheridan, Donna Peuquet and Alan MacEachren for their comments and suggestions about this paper. Thanks also go to the three anonymous reviewers for their insightful and constructive comments.

References

Bemmelen, J. van, Quak, W., Hekken, M. van, and Oosterom, P. van, 1993, Vector vs. Raster-based algorithms for cross country movement planning. In *Proceedings of Auto Carto 11* (Bethesda: American Congress on Surveying and Mapping), pp. 304–317.

Berg, M. D., and Kreveld, M. V., 1997, Trekking in the Alps without freezing or getting tired. *Algorithmica*, **18**, 306–323.

Collischonn, W., and Pilar, J. V., 2000, A direction dependent least-costs path algorithm for roads and canals. In *International Journal of Geographical Information Science*, **14**, 397–406.

Dijkstra, E. W., 1959, A note on two problems in connection with graphs. *Numerische Mathmatik*, **1**, 269–271.

Douglas, D. H., 1994, Least-cost path in GIS using an accumulated-cost surface and slope lines. *Cartographica*, **31**, 37–51.

Feldman, S. C., Pelletier, R. E., Walser, E., Smoot, J. R., and Ahl, D., 1995, A prototype for pipeline routing using remotely sensed data and geographic information system analysis. *Remote Sensing of Environment*, **53**, 123–131.

Goodchild, M. F., 1977, An evaluation of lattice solutions to the corridor location problem. *Environment and Planning*, **A9**, 727–738.

Lee, J., and Stucky, D., 1998, On applying viewshed analysis for determining least-cost paths on Digital Elevation Models. *International Journal of Geographical Information Systems*, **12**, 891–905.

Mitchell, J. S. B., 1988, An algorithmic approach to some problems in terrain navigation. *Artificial Intelligence*, **37**, 171–201.

Mitchell, J. S. B., Mount, D. M., and Papadimitriou, C. H., 1987, The discrete geodesic problem. *SIAM Journal on Computing*, **16**, 647–668.

O'Flaherty, C. A., 1967, *Highways* (London: Edward Arnold), 321 pp.

Oglesby, C. H., and Hewes, L. I., 1963, *Highway Engineering, 2nd edition* (New York: John Wiley and Sons, Inc), pp. 256–259.

Solka, J. L., Perry, J. C., Poellinger, B. R., and Rogers, G. W., 1995, Faster computation of optimal paths using a parallel Dijkstra algorithm with embedded constraints. *Neurocomputing*, **8**, 195–212.

Stefanakis, E., and Kavouras, M., 1995, On the determination of the optimum path in space. *COSIT '95*, 241–257.

Varadarajan, K., and Agarwal, P. K., 2000, Approximating shortest paths on a non-convex polyhedron. *SIAM Journal of Computing*, **30**, 1321–1340.

Vörös, J., 2001, Low-cost implementation of distance maps for path planning using matrix quadtrees and octrees. *Robotics and Computer Integrated Manufacturing*, **17**, 447–459.

Warntz, W., 1957, Transportation, social physics and the Law of Refraction. *The Professional Geographer*, **9**, 2–7.

Weiss, M. A., 1995, *Algorithm, data structures, and problem solving with C++* (MA: Addison Wesley).

Xu, J., and Lathrop, R. G., 1994, Improving cost-path in a raster data format. *Computers & Geosciences*, **20**, 1455–1465.

Xu, J., and Lathrop, R. G., 1995, Improving simulation accuracy of spread phenomena in a raster-based geographic information system. *International Journal of Geographical Information Science*, **9**, 153–168.

Zhan, C., Menon, S., and Gao, P., 1993, A directional path distance model for raster distance mapping. *COSIT '93*, 434–443.

Zhu, H., Eastman, J. R., and Toledano, J., 2001, Triangulated irregular network optimization from contour data using bridge and tunnel removal. *International Journal of Geographical Information Science*, **15**, 271–286.