A General-purpose Parameterized Shortest Diagonal Algorithm for Surface Reconstruction from Planar Contours

Chunming Zhang¹, Shaofei Jiang², Qing Wang¹, Tianhong Yang¹, Hanrui Zhang³
1. School of Resources and Civil Engineering, Northeastern University, Shenyang 110004, China
2. College of Civil Engineering, Fuzhou University, Fuzhou 350108, China
3. School of Information Science and Engineering, Northeastern University, Shenyang 110004, China
E-mail: zhangchunming@mail.neu.edu.cn, 408074568@qq.com

Abstract: In the development of a mining CAD program, the authors of this paper first attempted to use the widely accepted shortest diagonal algorithm to reconstruct 3D iron ore body from a set of planar contours, so as to aid in the comprehension of the ore body structure and facilitate its manipulation and analysis. However, the algorithm failed under certain circumstances, to be specific, the routine got “suspended” and “cones” were formed, due to its limitations. To address this problem, we first made an analysis of possible causes for these failures and then experimented on various approaches to intervene the advance speed of either contour in a contour pair so as to break the “dead loop”. Finally, we arrived at a cost effective, general-purpose parameterized shortest diagonal algorithm for triangulation and 3D surface reconstruction from planar contours. For easier apprehension, we present in sequence in this paper an explanatory description of the problem, a detailed analysis of the problem, an informative elucidation of the synchronization mechanism introduced into the new algorithm, and the new algorithm itself as well as the function of the differential parameter.

Keywords: surface reconstruction; triangulation, algorithm; computer graphics

1. Introduction

A geological section is viewed as the horizontal plane at a specified elevation, whereas a geological contour line is regarded as the intersection of an arbitrary ore body surface and a geological plane. A set of these serial geological sections on vertically and usually evenly spaced parallel planes comprises a contour line definition of an ore body surface, a conventional way to describe topographical surfaces. To aid in the comprehension of the structure of an iron ore body and to facilitate its manipulation and analysis, it would be more convenient if a three-dimensional solid could be constructed from a series of sections or if a three-dimensional surface could be fabricated over a set of cross-sectional contours, the procedures of which are virtually the same. The methods or algorithms used for conversion from contours to solids or surfaces can be grouped into two categories, the first being locally optimized approaches renown for their effective performance, like the one proposed by Christiansen [1] and used in this study, and the second being globally optimized ones as presented by Keppel[2] and Fuchs etc. [3]

In an endeavor to add three-dimensional surface reconstruction capabilities to a newly developed mining
CAD program, we first adopted the shortest diagonal algorithm [1] for triangulation of the three-dimensional surface of a solid iron ore body spanning over a set of cross-sectional contours, also known as a face definition of the surface, which employs triangular faces to describe a continuous tone irregular surface. The algorithm is based on the following assumptions: if any two nodes of the same contour are to be defined as nodes of the same triangle, they must neighbor each other on the contour line. Also, no more than two vertices of any triangle may be recruited from the same contour line. Things went quite smoothly until we stumbled upon some “disagreeable” pairs of contours, the triangulation of which had resulted in a mismatched surface or “a cone” to some extent, that is, some nodes or edges of the adjacent contours had neither been connected nor become part of any triangular face while other nodes or edges had been repeatedly connected. However, in a strict sense, these contour pairs did measure up to contour pairing protocols required by the algorithm, i.e., mutually centered, of similar size, and of similar shape after having been mapped to a unit square. In addition, the contours identified to have caused the problem were only mildly complex in shape. In search for the causes of this problem, numerous experiments and thorough investigations were carried out [4-12] before it is ascertained that, when one contour’s center of gravity, not necessarily the center of the rectangular window or “envelope” which encloses the contour, offsets another contour’s center of gravity after mapping or coordinate normalization, a mismatched or misconnected surface is very likely to be constructed under certain circumstances which will be described later in detail.

Now that the reason is known, the algorithm must be revised to cope with the problem or completely replaced by a new algorithm capable of dealing with situations like this. The authors of this paper decided to modify the original algorithm so as to make it applicable to extreme circumstances as well. By implementing a mechanism that is intended to keep the algorithm from being “suspended” at a node, i.e., repeatedly connecting adjacent contour edges to this same node thus forming a cone or a partial cone, we arrived at the revised shortest diagonal algorithm that can well handle this problem, and with which pairs of adjacent cross-sectional contour lines whose centers of gravity fall far apart can be reconstructed into a three-dimensional surface without leaving any nodes or edges untended.

2. Statement of problem

As shown in Fig. 1, a pair of contour lines on adjacent cross sections is used for the experiment. In the interest of simplicity of presentation, the sample pair of adjacent contour lines for triangulation has been deliberately simplified to the maximum so as to not to obstruct readers from comprehending the triangulation process.
Suppose the sample contour pair selected for illustrative purpose has already been mapped onto a unit square, in other words, it has already become mutually centered, of similar size and of similar shape after mapping (see Fig. 1). Before triangulation can commence, we must make sure the node sequence of both contours goes in the same direction and the first nodes of each contour are proximate. Since the shortest diagonal algorithm requires that the connectivity of both contours proceed in the same rotation direction, we might just as well adopt the anti-clockwise direction as the default rotation direction of both contour lines for convenience’s sake. Next we go through an iterative process in order to find the proximate pair of nodes between the planar contours and make them the starting nodes for respective contour lines. It can be seen from Fig. 1 that the top contour P consists of three distinct nodes (P0-P2) starting with node P0, whereas the bottom contour Q comprises five distinct nodes (Q0-Q4) beginning with node Q0. The proximate diagonal P0Q0 is assumed to be the start line for triangulation. Since contour connectivity requires PDP1 and Q0Q1 to be the bases of triangles, there are only two candidates for the first triangle: ΔPDP0Q0, and ΔP1Q0Q1. The diagonal PDP1 is shorter than the diagonal P1Q0, therefore triangle ΔP0Q0Q1 is selected on the basis of the shortest diagonal. The routine moves on to consider the next triangle: ΔPDQ1Q2, and ΔPDQ1Q2. This time ΔP0Q0Q1 is selected as a consequence of its shorter diagonal. Repeat the above procedure until the current “node pointer” for each contour line loops back to its start position again respectively. The completed triangulation based upon this shortest diagonal merit function is shown in Fig. 2 and Fig. 3. Fig. 2 presents a three-dimensional wire-frame view of the contour reconstruction, while Fig. 3 displays a three-dimensional surface view in which the contour planes themselves are not shaded in order to give readers a clearer perspective. Fig. 4 shows the contour connectivity scheme. Contour lines are straightened as such that the connections made or “moves” taken can be facilely observed. Sequentially numbered in the figure are the triangles constructed by consecutively connecting diagonals PDP1, PDP2, PDP3, PDP4, PDP5 and PDP6 starting from diagonal P0Q0 (dash lines), namely, ΔPDP0Q1, ΔPDP0Q1, ΔPDP0Q2, ΔPDP0Q3, ΔPDP0Q4, and ΔPDP0Q5. However, since some diagonals (dotted lines) such as PDP2, PDP4, etc., are mistakenly selected, the triangles that are created hereafter, for instance, ΔPDP0Q1 and ΔPDP0Q4, etc. (not shown in the figure for tidiness), either appear invalid or intersect those triangles previously created. A partial cone is thus formed, resulting in a dangling contour edge that is not base of any triangles (Fig. 3).
3. Problem analysis

Through experimentation and analysis, the authors of this paper discover that the likely causes for the above-mentioned problem are either or both of the following: 1) mismatching centers-of-gravity of the contours concerned; 2) synchronization failure due to the limitations of the shortest diagonal algorithm.

The “cone” problem is supposed to have been solved by mapping contour lines onto a unit square prior to triangulation, as stated by Christiansen in his algorithm\(^1\). However, the problem still exists with our example contours, even though they are mutually centered after mapping. In fact, they are only centered round the geometry centers of the “envelopes” which enclose the contours (marked by tiny red dots in Fig.1), rather than the centers-of-gravity of the contours themselves (indicated by small blue crosses in Fig.1). Most of time geometry centers of the envelopes do closely match the corresponding centers of gravity, but not necessarily. What makes our case go extreme is that the contours do not even overlap after mapping (see Fig.1), which is regarded as a precondition for “cone” formation and precisely the reason why “a cone” can be formed. That is,
under certain circumstances the formation of a “cone” is still evitable even if the mapping has been done.  
When the contours have a great disparity in number of nodes, or when they take on similar outer shape but with totally different inner shape like in our example, measures should be taken whenever necessary to intervene in contour synchronization once triangulation is in process, to prevent the routine from getting suspended at a node, i.e., keeping connecting to the same node, as a side effect of which a “cone” is formed as such. The shortest diagonal algorithm constantly searches for the next proximate diagonal without taking into consideration the contour synchronization, which should be properly addressed when dealing with complex contours with a great difference in number of nodes. In other words, one contour proceeds too fast while its counterpart goes too slowly. It gets suspended somehow at certain node. As long as the diagonal from the node on one contour line to its corresponding node on another contour line is contemporarily the shortest among alternative or candidate diagonals, the routine will simply not move on to the next node on the same contour line. Obviously, the situation would not improve without external assistance or intervention because of the algorithm’s nature.

4. Solution and new parameterized algorithm

4.1. Solutions

On the basis of problem analysis, the authors have worked out several feasible approaches to resolve the problem, for example, 1) to add a new center-of-gravity oriented mapping procedure to ensure that contours somehow overlap, and with which contours are mutually centered on the centers-of-gravity of contours instead of the geometry centers of the envelop boxes that enclose the contours; 2) a new algorithm that allows, on some occasions, inconsistent rotational direction in triangulation in which one contour goes clockwise while the other moves anti-clockwise; 3) a branching of original contour lines to make them agreeable to the original algorithm; 4) a revised shortest diagonal algorithm that mainly involves in introduction of a parameter to synchronize the triangulation process, etc. In brief, options are many, and cited above are just a few of them.

For simplicity and applicability, it is presented in this paper a general-purpose parameterized shortest diagonal algorithm for surface reconstruction from planar contours. A synchronization mechanism is introduced to the algorithm so as to intervene at a crucial moment, for instance, to slow down the pace when it senses a fast movement or to speed up when the routine gets suspended at a node. Specifically, the synchronization is controlled by a differential parameter introduced into this algorithm, the value of which can be either calculated automatically given a user defined equation or simply assigned to an experiential value, like 5% by default. Note that the value of the differential parameter is within the range of 0 to 1.

We presume that the advance speed for a particular contour line can be denoted by the ratio of the distance that the current node pointer (representing the current node) has moved so far along its path to the perimeter of the contour line, hereinafter referred to as the advance ratio. As triangulation proceeds, when the absolute value of the difference between the advance ratio for one contour and that for the other is greater than the value of the differential parameter that is either preset or dynamically calculated as it is described earlier, the triangle to be selected will no longer be determined by the proximate diagonal rule but rather by the difference of advance ratios for the two contours, i.e., the contour whose advance ratio is less that that of the other contour is forced to make a move forward, that is, move to the next node, no matter what result is for the proximate diagonal rule. On the other hand, if the absolute value of the difference of the advance ratios for contours falls within the range designated by the differential parameter, the proximate diagonal rule prevails.

A very interesting fact about this new synchronization mechanism is that when the differential parameter is
equal to or set to 0, the algorithm turns out to be the so-called preferential contour synchronization algorithm; when the differential parameter is equal to or set to 1, the algorithm becomes the original shortest diagonal algorithm.

Take the contours in Fig. 1 for example. With this revised algorithm the triangulation process goes like this: the proximate diagonal \(P_0Q_0\) is assumed to be the start line for triangulation. Since contour connectivity requires \(P_0P_1\) and \(Q_0Q_1\) to be the bases of triangles, there are only two candidates for the first triangle: \(\Delta P_0P_1Q_0\) and \(\Delta P_0Q_0Q_1\). The diagonal \(P_0Q_1\) is shorter than the diagonal \(P_0Q_0\), therefore triangle \(\Delta P_0Q_0Q_1\) is selected on the basis of the shortest diagonal. The routine moves on to consider the next triangle: \(\Delta P_0P_1Q_1\) and \(\Delta P_0Q_0Q_1\). This time \(\Delta P_0P_1Q_1\) is selected as a consequence of its shorter diagonal. Repeat the above procedure until the current “node pointer” for each contour line loops back to its start node again respectively. The completed triangulation based upon this shortest diagonal merit function is shown in Fig. 5 and Fig. 6. Fig. 5 presents a three-dimensional wire-frame view of the contour reconstruction while Fig. 6 displays a three-dimensional surface view in which the contours themselves are not shaded in order to have a clearer perspective. Fig. 7 shows the contour connectivity scheme. Contour lines are straightened as such that the connections made or “moves” taken can be faciely observed. Sequentially numbered in the figure are the triangles constructed by consecutively connecting diagonals \(P_0Q_0\), \(P_1Q_1\), \(P_2Q_2\), \(P_3Q_3\), \(P_4Q_4\), \(P_5Q_5\) and \(P_6Q_6\) starting from diagonal \(P_0Q_0\) (dash lines), namely, \(\Delta P_0Q_0Q_1\), \(\Delta P_0P_1Q_1\), \(\Delta P_1Q_1Q_2\), \(\Delta P_2Q_2Q_3\), \(\Delta P_3Q_3Q_4\), \(\Delta P_4Q_4Q_5\) and \(\Delta P_5Q_5Q_6\). As you can see from the figure, intersections or dangling nodes or edges are no longer present with this new algorithm. A comparison with the triangulation process shown in the “STATEMENT OF PROBLEM” chapter shows that the new algorithm selects \(P_1Q_1\) instead of \(P_2Q_2\) when the top contour attempts to advance from \(P_1\) to node \(P_2\) and the bottom contour tries to move from \(Q_2\) to node \(Q_3\). Actually, diagonal \(P_2Q_2\) is supposed to be selected based on proximate diagonal rule alone, but not when synchronization is taken into consideration. For current nodes in both contours, the advance ratio for top contour \(P\) is the length of path \((P_0-P_1-P_2)\) divided by the perimeter of the top contour or the length of path \((P_0-P_1-P_2-P_0)\), while the advance ratio for bottom contour \(Q\) is the length of path \((Q_0-Q_1-Q_2-Q_3)\) divided by the perimeter of the bottom contour or the length of path \((Q_0-Q_1-Q_2-Q_3-Q_4-Q_5)\). The calculation result shows that the advance ratio for contour \(P\) is greater than that for contour \(Q\). Moreover, the difference of advance ratios is greater than the differential parameter, which means one contour moves too fast and should not be encouraged. Therefore, the contour with a less advance ratio, i.e., contour \(Q\) in this case, is expected to step forward to the next node, forming triangle \(\Delta P_1Q_2Q_3\).

**Fig. 5 A wire-frame view of correct triangulation (planar contours are shaded)**
Fig. 6 A three-dimensional surface view of correct triangulation (planar contours are not shaded)

Fig. 7 The connectivity scheme of correct triangulation

Fig. 8 Three-dimensional surface reconstruction from contour lines
4.2. New parameterized shortest diagonal algorithm

Let the top contour $P$ be defined by the sequence of $m$ distinct nodes $P_0, P_1, \ldots, P_{m-1}$ and let the bottom contour $Q$ be defined by the sequence of $n$ distinct nodes $Q_0, Q_1, \ldots, Q_{n-1}$ (See Fig. 8). Note that $P_0$ follows $P_{m-1}$ and $Q_0$ follows $Q_{n-1}$ to complete the loops respectively. Suppose $\Psi$ is the perimeter of contour $P$, and $\Omega$ is that of contour $Q$. In contrast, $\Psi_i$ is the accumulated length of the path from node $0$ to node $i$ on contour $P$, and $\Omega_j$ is the accumulated length of the path from node $0$ to node $j$ on contour $Q$. $S$ is a result set of triangles selected by the algorithm. $\delta$ is the differential parameter.

**Step 1:** Unify the rotational direction for both contours. The anti-clockwise direction is preferred.

**Step 2:** Normalize all node coordinates by mapping both contour lines onto a unit square.

**Step 3:** Search for the proximate pair of nodes between contour $P$ and $Q$, and use them as the starting nodes for triangulation. Sort the nodes of contour $P$ and $Q$ when necessary.

**Step 4:** Calculate the perimeters of contour $P$ and $Q$ respectively, i.e., $\Psi$ and $\Omega$.

**Step 5:** Commences the triangulation process. For each node in the contours, repeat the following procedures (1)-(4). Suppose the current node pointer for contour $P$ has moved to node $i$ and that for contour $Q$ has reached node $j$.

1. Compute $\Psi_i+1$ and $\Omega_j+1$ respectively.
2. If $\Omega_j+1/\Omega - \Psi_i+1/\Psi > \delta$, contour $P$ moves forward to node $P_{i+1}$. Select the triangle $\Delta P_iP_{i+1}Q_j$ and add it to the result set $S$.
3. If $\Psi_i+1/\Psi - \Omega_j+1/\Omega > \delta$, contour $Q$ advances to node $Q_{j+1}$. Select the triangle $\Delta P_iQ_jQ_{j+1}$ and save it to the result set $S$.
4. If neither condition above is satisfied, then select the shorter diagonal from two candidates $P_iQ_{j+1}$ and $P_{i+1}Q_j$ and add the corresponding triangle to the result set $S$. Specifically, if $|P_iQ_{j+1}| < |P_{i+1}Q_j|$, then contour $Q$ advances to node $Q_{j+1}$. Select the triangle $\Delta P_iQ_jQ_{j+1}$ and save it to $S$. Otherwise, contour $P$ moves forward to node $P_{i+1}$. Select the triangle $\Delta P_iP_{i+1}Q_j$ and add it to $S$.

**Step 6:** De-normalize node coordinates of both contours and reconstruct the three-dimensional surface by rendering the triangles in the result set $S$.

5. Conclusions

In an endeavor to add three-dimensional surface reconstruction capabilities to a newly developed mining CAD program, the authors happen to find that the shortest diagonal algorithm widely used today does not work properly under some circumstances, even for contours that are mutually centered, of similar size and of similar shape after mapping. The likely causes for this are ascertained by the authors to be mismatching centers-of-gravity of the contours, no or little overlapping, as well as synchronization failure because of the algorithm’s limitations.

With this understanding, the authors of this paper put forward a new parameterized algorithm as an improvement to the original shortest diagonal algorithm. The new approach implements a synchronization mechanism that is controlled by a differential parameter to intervene the advance speed so as not to get suspended under extreme circumstances. A very interesting fact about this synchronization mechanism is that the greater the differential parameter value is, the more the new algorithm looks like the original shortest diagonal algorithm. When the value is set 1, it becomes the original shortest diagonal algorithm.

Furthermore, the algorithm proposed is also applicable to other circumstances where 3D surface
reconstruction from planar contours is required.

**Acknowledgments:** The authors of this paper acknowledge the financial support from the Natural Science Foundation of China (Grant No. 50878057, 50974041, 50674025), the Ph.D. Programs Foundation of Ministry of Education (Grant No. 20093514110005), and the Program for New Century Excellent Talents in University, Fujian Province (No. XSJRC2007-24).

**References**


