ADAPTIVE HIERARCHICAL TRIANGULATION†

Lori Scarlatos
Grumman Data Systems
1000 Woodbury Rd.
Woodbury, NY 11797

Theo Pavlidis
State University of New York
Dept. of Computer Science
Stony Brook, NY 11794

ABSTRACT

Numerous cartographic applications rely on triangulated surface models for accurate three-dimensional representations of real-world data. Some applications require a series of triangulations to represent a single surface at progressively finer levels of detail. Past work has emphasized techniques relying on plane geometry using little or no surface data. We propose a technique that adapts the triangulation to surface characteristics. Because our adaptive hierarchical triangulation focuses on the topology of a surface, it reduces the number of triangles required for a good approximation. It also produces fewer long and slivery triangles within each level of detail. Our structure guarantees the accuracy of each level of detail. Our structure only retains important triangles, thereby reducing the total number of triangles that must be stored and searched. Furthermore, the tree-like structure of our hierarchy is well-adapted to multiple resolution views, allowing smooth transitions between levels of detail in flight simulators. These advantages add up to a triangulation that provides great accuracy in a model that can be rapidly searched, rendered, and otherwise manipulated. Tests on data with digital elevation input have confirmed the above theoretical expectations. On eight such sets the average ''sliveriness'' with the new method was between 1/5 and 1/10 of old triangulations and number of levels was about one third. Although the number of descendants at each level increases slightly, the total number of triangles is lower, implying faster spatial search.

INTRODUCTION

Geographic information systems, flight simulators, and numerous other cartographic applications rely on digital terrain models for simulation, visualization, and analysis. Increasingly, these applications require both greater accuracy and data compression from these models. Triangulated models are popular because triangles are simple to manipulate and render. Triangulated Irregular Networks (TINs) offer the additional

† Portions of this paper were included in a paper presented at Visualization '90. For further details and examples of this work, interested readers should request a technical report entitled "Hierarchical Triangulation Using Terrain Features" from Lori Scarlatos.

advantage of not being bound by regularity constraints. TINs can therefore approximate any surface at any desired level of accuracy using a very small number of polygons. Organizing TINs in a level of detail hierarchy provides accurate generalizations meeting different application requirements. Hierarchical organization allows easy implementation of such operations as zooming when viewing the surface. It also facilitates searching and other geometrical operations such as finding the intersection of two surfaces. Furthermore, it makes real-time simulation and visualization possible for applications that can represent less important areas with less detail in mixed-resolution models.

This paper describes a hierarchical triangulation built from a digital elevation model in grid form. Each level in the hierarchy corresponds to a different level of detail that approximates the surface within a given tolerance (i.e. maximum error), The top level is the coarsest, containing the fewest triangles and approximating the surface within the greatest tolerance value t_0. The $i+1^{th}$ level in the hierarchy is related to the i^{th} level as follows. Tolerance t_{i+1} is smaller than t_i. Each triangle T_j^i of the i^{th} level is split into n descendent triangles $T_{j_1}^{i+1}, \cdots, T_{j_n}^{i+1}$ at the $i+1^{th}$ level, where n can be any positive integer.

In the following section, we provide a background of past work on triangulation and hierarchical triangulation. We then describe our adaptive hierarchical triangulation methodology, and discuss its advantages over other methods. Next, we outline the implementation of this algorithm and the resulting data structure. We conclude with a discussion of test results obtained from running this implementation.

BACKGROUND

Triangulation

Triangulation algorithms generally fall into two groups: those that efficiently triangulate a given polygon, and those that use triangulation to approximate surfaces. In the former category, the primary issues are computational complexity (Aho et al 1974, Garey et al 1978, Clarkson et al 1989, Fournier and Montuno 1984) or size and shape of the resulting triangles (Baker et al 1988). We are more interested in the latter category where the primary goal is to produce the best possible surface approximation. This surface approximation should contain as few triangles as possible while still meeting given accuracy requirements. At the same time, it must minimize the number of very thin, slivery triangles which can produce artifacts in renderings of surface models.

Surface triangulation algorithms may be further categorized by the input data they triangulate. Surface triangulation produces a planar graph by adding edges, and sometimes even points, to an initial graph. This initial graph, comprised of points (nodes) on the surface, may or may not include connecting edges (critical lines) that further define that surface.

In the first sub-category of surface triangulation algorithms, the initial graph contains no initial edges. Although some of these triangulation algorithms rely on alternate techniques (Mirante and Weingarten 1982, Manacher and Zobrist 1979) most are a variation on the Delaunay triangulation scheme (Watson 1981, Dwyer 1987, Preparata and Shamos 1985, Watson and Philip 1984 are only a few). Algorithms based on Delaunay triangulation have the advantage of producing few slivers. However, Delaunay's method was developed to find nearest neighbors on a plane, not approximate surfaces. These algorithms tend to ignore the third dimension, and may therefore produce triangle edges that contradict the topology of the actual surface (Christensen 1987).

The second sub-category of algorithms assumes that all points in the initial graph have at least one connecting edge. These edges correspond to the linear patterns that characterize many surfaces, particularly natural ones such as terrain. Because these edges describe surface topology, they are retained in the final triangulation to maximize model accuracy. Some papers such as (Christiansen and Sederberg 1978, Dennehy and Ganapathy 1982) deal with triangulating cross-sections from tomographic scans, although the methods of both of these papers require human intervention when the contours get complex. Other algorithms for triangulating cartographic critical lines have been recently published (Christensen 1987, Scarlatos 1989, Chew 1989).

Hierarchical Triangulation

Hierarchical triangulations provide both multiple levels of detail and a structural ordering for fast spatial search. Recent papers (Goodchild 1989, Fekete 1990) propose to represent the entire planetary surface with a quadtree-like hierarchy of regular triangular tessellations. This is an excellent scheme for dividing huge data bases into manageable areas of interest which may be georeferenced in constant time. However, as shown in (Scarlatos 1990b), the placement of points in a regular tessellation is independent of the surface topology. Hence coarser levels of detail can distort or entirely miss important terrain features, and finer levels of detail can cause unnecessary bottlenecks by producing large numbers of triangles where a few would do as well.

Previous work by one of the authors has researched techniques to find critical points and lines (Scarlatos 1990a), triangulate them (Scarlatos 1989) and then refine those triangulations to produce a hierarchy of detail levels for fast spatial search with maximum accuracy (Scarlatos 1990b). These algorithms represent significant improvements over other algorithms, producing good triangulations. However, the above algorithms do not allow for refinement down to a specified level of accuracy.

Although several refinement techniques have been suggested in the literature (Fowler and Little 1979, DeFloriani et al 1984, DeFloriani 1989), these algorithms can introduce artifacts to a terrain model because they consider only the locality of points in a 2D plane instead of actual terrain topology. Consider, for example, DeFloriani's first algorithm for triangle refinement (DeFloriani et al 1984) which splits a triangle by connecting its corners to a selected interior point (usually, the point of maximum distance between the surface and the plane defined by the vertices of the triangle). The algorithm ignores the *coherence* of cartographic features such as valleys or ridges which have a linear structure.

Figure 1 shows the results of ignoring such coherence. We assume that a ridge (its points marked by small circles in (a)) crosses the triangle. (b) shows that the maximum point triangulation will produce an unreasonably large number of triangles. Even worse, the triangles will have very sharp angles, which is an undesirable property (Baker et al 1988). Such triangulations may cause numerical stability problems in finite element methods and also produce undesirable display artifacts. In contrast, if we realize that we deal with a ridge and introduce a dividing line along it as shown in (c) we will end up with fewer triangles, none of them slivery. We should point out that triangles with very sharp angles may be inevitable for some types of data. For example, if we have a steep cliff we will see large differences in the value between adjacent elevation points. Then triangles with very sharp angles cannot be avoided.

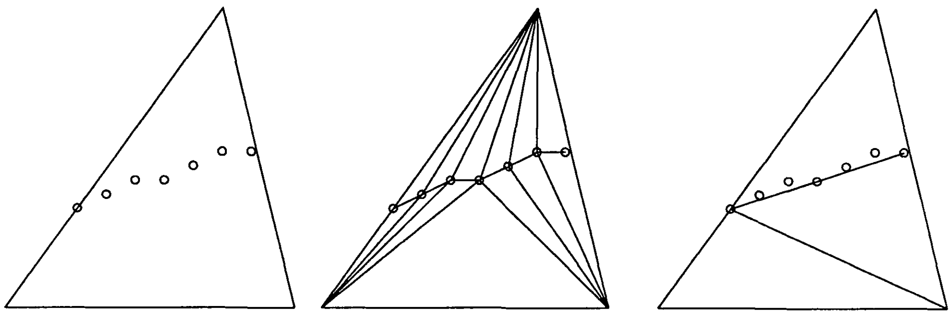


Figure 1. A ridge passing through a triangle has (a) points along that ridge that are farthest from the triangle which may be triangulated using (b) the maximum error point to split each triangle, or (c) cartographic coherence to approximate the ridge line.

METHODOLOGY

Our goal is to reduce the number of splits or refinements required to achieve a desired level of detail and limit the number of slivery triangles in the results. A generalization of the critical line method could produce better accuracy with fewer triangles. We have implemented such a strategy as follows. We start with a coarse triangulation. This may be carefully produced by techniques outlined in Scarlatos' three papers, or it may be as simple as a rectangular area split in two. We then refine this triangulation by adding points from the original digital elevation grid and connecting edges. Our refinement technique pays particular attention to terrain characteristics, approximating critical lines at each step.

To accomplish this, we determine the best places to split each triangle by calculating four error values: one inside the triangle, and one on each of the three edges. All error values measure the difference between original grid point elevations and their projections to the surface of the triangulated model. To avoid quantization artifacts, grid points near a triangle edge are considered to be on that edge.

Figure 2 shows the five ways that a triangle may be refined. If an isolated peak or pit resides within the triangle, it is split at that central peak or pit point as shown. If a single ridge or channel line travels up to that peak or pit, the triangle is split where that line crosses the edge of the triangle and at the central peak or pit point. If, however, a single ridge or channel line enters the triangle and ends at a saddle point or flat, then the center point is insignificant and the triangle is split by one edge as shown. If a ridge or channel line passes through the triangle, significant errors will be found on two edges of the triangle. A line connecting these points approximates the topographical line, and an additional edge splits the remaining quadrilateral. Finally, if a triangular patch corresponds to a rapidly fluctuating surface, many points are likely to have significant errors. Splitting this type of triangle on all edges segments the high-frequency regions which may then be further refined.

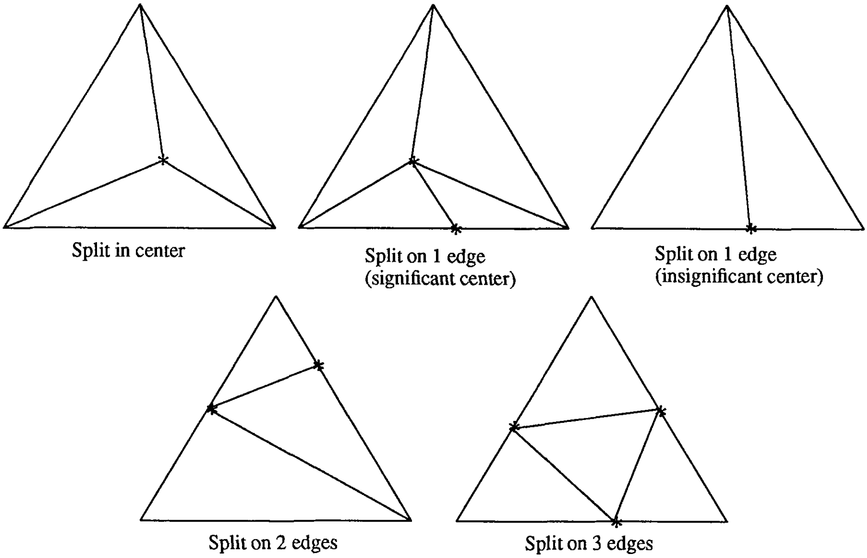


Figure 2. Split strategies for preserving cartographic coherence.

We repeatedly split the triangles until they all meet the given accuracy requirements for the current level of detail. Intermediate triangles, used to produce but not included in the final triangulation for the current level of detail, are discarded. This reduces the number of levels in the hierarchy and the number of triangles within each level, making faster search, display, and processing possible. If polygon constraints are more important than the level of error, we can easily check the polygon count and terminate a level when the limit is approached.

IMPLEMENTATION OF THE ALGORITHM

We implemented our adaptive hierarchical triangulation algorithm as follows. A main program retrieves the input data, calls the appropriate triangulation routines, and writes out the results to a data base. Input parameters include an initial triangulation, the number of levels to create in the hierarchy, and a tolerance for each level. A main loop generates each level of detail. At the start of the loop, the current triangulation represents level i in the hierarchy. At the conclusion of the loop, the current triangulation represents level $i+1$ in the hierarchy. The body of the loop splits triangles in the current triangulation until all errors lie within the given tolerance for that level.

Data Structures

We generate our adaptive hierarchical triangulation from a digital elevation matrix which covers a rectangular area of interest. The region outside the area of interest is represented by four neighboring "triangles". These extend infinitely to the north, east, south, and west of the area of interest. Points within the area of interest provide the endpoints of -- and are entirely covered by -- an initial triangulation. Each point may therefore be associated with zero, one, or two triangles. Points acting as triangle vertices have no triangle associations. If the distance from a point to a triangle edge is less than the distance between grid posts in the original matrix, then that point is considered near that edge. A point on or near a triangle edge is associated with the two triangles that share that edge. Otherwise, the point is within a single triangle. A Membership list contains records of each point's two associated triangles and a distance to their shared edge. When a point is near more than one edge, the membership records form a linked list in order of increasing distance values.

A triangle in the hierarchy is defined by three points from the original elevation matrix. Each triangle is associated with a level of detail and contains pointers to its parent, its children, and three neighboring triangles that share its edges. In addition, triangles have temporary structures keeping track of their splitting points, the maximum error found within them, and the number of edges to be split. A flag indicates whether the triangle meets the accuracy standards of the current level.

Splitting Triangles

For each specified level of detail, our program repeatedly splits triangles until the triangular mesh approximates the surface within the given tolerance. We find errors within a triangle by taking all grid points within the boundaries of that triangle, projecting them to the surface of the triangle, and comparing the results to the original elevation values. Errors are found in four regions on a triangle: on each of the three edges, and within the triangle. These errors determine if and how the triangle will be split.

Next, we find the point producing the maximum error in each of the four regions for each triangle. Notice that the point with maximum error on one triangle's edge will also be the point with maximum error for the other triangle sharing that edge. If the error is significant, then that point will split the triangle(s) it belongs to. Significance may be calculated in two ways. First, if the given value is greater than the threshholded error for the current level of detail, then that point is significant. Alternatively, if the given value is more than some percentage of the maximum error found within a triangle, then that point is significant. In either case, a point is insignificant if its error falls at or below the threshhold for the current level of detail.

After all splitting points have been found, we ensure that the splitting point on an edge of one triangle is also a splitting point for the triangle sharing that edge. Then we split all the triangles. Although each of the five regular triangulation algorithms is different, they all follow the same pattern of steps. First the outer edges of the triangle are split. If the splitting point does not lie exactly on the outer edge, this will introduce a minor bend in the triangle. Extremely thin triangles produce special cases which must be handled separately. Our technical report discusses the necessary special triangulation in depth. In the next step we add all the new interior edges. As we modify and add edges, we update the point membership list indicating what triangle(s) each point belongs to. Finally, new triangle records are added, and triangle neighbor values are updated.

Data Base Structure

This algorithm produces all of the information required to both render the 3D surface and search for spatial relationships. A header record includes information such as a ground position for the lower-left corner of the triangulation; spacing between posts in the original grid; elevation ranges in the triangulation; number of levels. This is followed by the level records. Each level has a threshhold of allowable error, used to produce the triangulation. It also has a number of points, number of triangles, and a list of triangles. Each triangle is defined by 3 point indices, and has a parent pointer, child pointers, and neighbor pointers. All this is followed by a single point list. Only points that appear in the triangulation are written to the data base; all others are unnecessary. Points are ordered such that if level L uses N points, then it uses points $1, \cdots, N$. This reduces retrieval time for a level of detail.

RESULTS

We tested our algorithm on eight (8) areas of interest (AOI) representing four very different types of terrain. AOI 1-2 contains numerous plateas; AOI 3-4 is a relatively flat region; AOI 5-6 contains mountains rising out of the foothills; and AOI 7-8 represents a portion of the Cascade mountain range. Our data comes from the Defense Mapping Agency's Digital Terrain Elevation Data (DTED) Level 1 which has three seconds of an arc between posts. Each test AOI covers 75x75 elevation points. A triangulation employing all 5625 points in an AOI would contain 10,952 triangles.

We implemented the adaptive hierarchical triangulation algorithm with varying parameters to see which behaved best. The first parameter is how we determine the significance of point p 's error e_p. Error e_p may be considered significant compared to 1) tolerance value t_i for level i, so that $e_p \geq t_i$, or 2) a percentage N of the maximum error $e_{t_{max}}$ found for current triangle t, so that $e_p \geq e_{t_{max}}$. The second parameter determines when we split a triangle at one edge and a significant center (as shown in Figure 2). Center point c may be considered significant compared to 1) the error e_v of splitting point v on the edge of the triangle, so that $e_c \geq e_v$, or 2) the significance value used to determine the significance of all other points, as determined by the first parameter. Hence we ran four optional programs. Option 1 uses tolerance to determine significance, and requires a center point to be at least as significant as an edge point in order to be used. Option 2 uses 75% of the maximum error within a triangle to determine significance, and also requires a center point to be at least as significant as an edge point. Options 3 and 4 are like options 1 and 2 respectively, except that a center point's significance is determined by the usual measures. As a basis of comparison, we implemented DeFloriani's first algorithm (DeFloriani et al 1984) and ran it with the same test data.

We executed DeFloriani's algorithm and all four options for our algorithm using the eight AOIs as input, producing triangulations with a minimum error of 10 meters. All tests demonstrated that adaptive hierarchical triangulation works well. Tables 1-4 show some of our results.

A better triangulation will produce fewer slivery triangles. The table shows how slivery the resulting triangles were. We measured sliveriness with the following ratio, calculated for each triangle in the triangulation: $\frac{Perimeter^2}{Area}$. The best possible ratio is approximately 20.78 for an equilateral triangle. Larger values represent thinner triangles, so smaller numbers are better. We summed all of these ratios together and divided by the total number of triangles to get an average sliveriness figure. We then divided that result by the sliveriness ratio for an equilateral triangle. Note that on the average most of the triangles have much sharper angles than sixty degrees. Using DeFloriani's algorithm, some angles are as small as 0.25 degrees. Notice how much better adaptive hierarchical triangulation performed, using all four options. Options 1 and 2 seem to work about equally well, indicating that the best measure of point significance is determined by data characteristics. Options 3 and 4 consistently performed a little worse. This leads us to conclude that a center point should only be included in the division of a triangle if it is more significant than the edge point.

Table 1

Measures of Sliveriness*					
AOI	DeFloriani-1	Option 1	Option 2	Option 3	Option 4
1	32.294	5.086	5.113	6.301	6.578
2	52.487	9.645	11.882	11.074	11.107
3	35.889	5.934	5.672	6.398	5.998
4	50.682	11.315	10.969	12.581	12.854
5	56.835	14.843	8.757	14.329	8.676
6	40.932	5.305	5.371	7.376	7.437
7	51.261	4.352	4.932	6.089	7.367
8	39.925	5.949	6.200	6.805	7.153

* normalized to 1 for an equilateral triangle

Table 2

Comparison of Hierarchies				
AOI	Number of Levels*		Average Number of Children**	
	DeFloriani-1	Option 1	DeFloriani-1	Option 1
1	15	5	2.5	2.8
2	17	5	2.5	2.4
3	17	5	2.5	3.6
4	17	5	2.5	3.6
5	19	5	2.5	2.4
6	18	5	2.5	2.5
7	17	5	2.5	2.4
8	18	5	2.5	2.3

* number of levels specified for new algorithm

** number of children assumed to be 2.5 for old algorithm

A better triangulation will permit fast spatial search. The time required for a search is determined by the number of levels that must be searched, and the number of child nodes that must be examined at each level. DeFloriani's algorithm, which always splits a parent triangle into 2 or 3 children, has an average of about 2.5 children per parent node. The number of levels in a hierarchy depend on the number of iteration levels required to build the triangulation. Adaptive hierarchical triangulation, on the other hand, guarantees a fixed number of levels in the hierarchy, but can split a parent triangle into any number of children. Although one may presume that a very large number of children will be produced, table 2 shows that this is not the case. Table 2 shows that search times using an adaptive hierarchical triangulation will be as fast as, or faster than, the other. Additional results can be found in our technical report.

A better triangulation will result in fewer triangles. Table 3 shows the total number of triangles in the hierarchy. Notice that the options that produced the fewest total triangles also produced the least slivery triangles. Table 4 shows the number of triangles at the highest level of detail, with a maximum error of 10 meters at each point. Compare this to 10,952 triangles for the original grid. Although the difference in values is not striking, the adaptive hierarchical triangulation usually produced fewer triangles than DeFloriani's algorithm.

Figure 3 demonstrates the significance of the improvements made by the adaptive hierarchical triangulation. Figure 3 a shows a view of AOI 1 using the original grid data. Figure 3 b shows the same view of the data triangulated with DeFloriani's algorithm for a maximum error of 10 meters. Figure 3 c shows the same view of the data triangulated with our algorithm (using option 1) for a maximum error of 10 meters. All three views were rendered with Gouraud shading. Notice the severe artifacts caused by very thin triangles in the DeFloriani model.

While Delaunay triangulations have been proposed as means for reducing the number of very sharp triangles within hierarchical structures (DeFloriani 1989), Delaunay triangulations have serious drawbacks as discussed in (Christensen 1987). In some cases, using Delaunay triangulation to add points can actually increase error levels in the model, even though the model contains more triangles. The algorithm of Baker et al (1988) while it avoids generating obtuse triangles, it generates far too many points and triangles for our purposes.

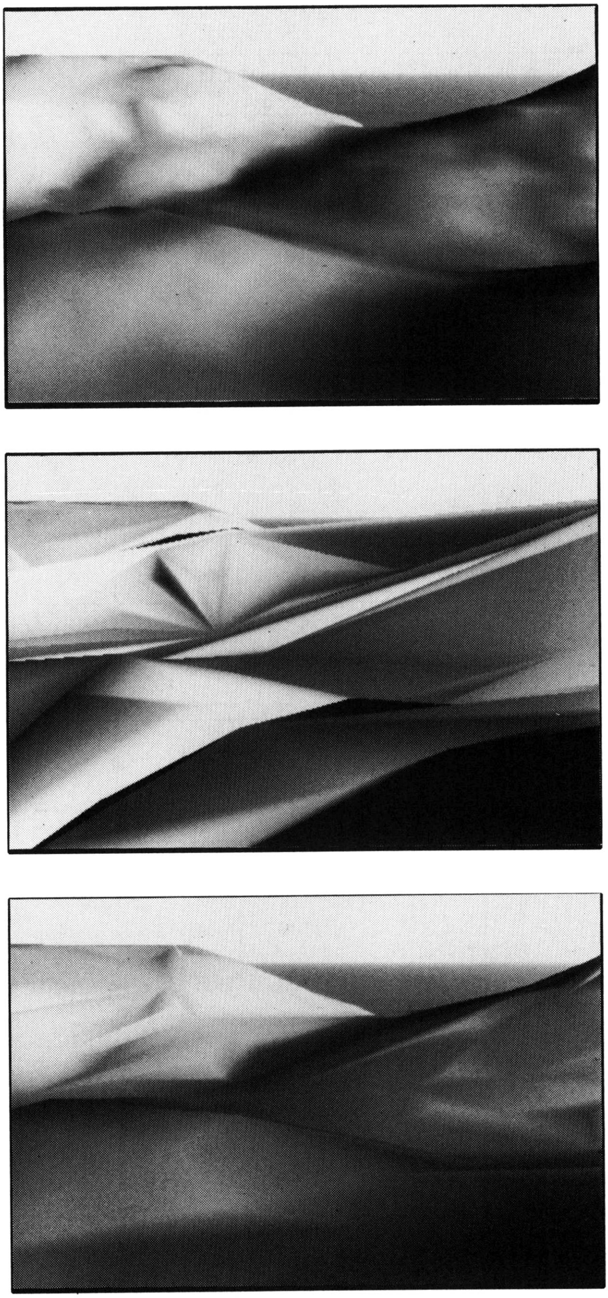
Table 3

Total Number of Triangles in the Hierarchy					
AOI	DeFloriani-1	Option 1	Option 2	Option 3	Option 4
1	2918	2866	2876	3208	3195
2	4198	3964	4124	4344	4364
3	2576	1862	1806	2022	2055
4	2007	1551	1525	1737	1757
5	5433	5339	5179	5508	5372
6	3935	3283	3289	3624	3604
7	4908	4655	4735	4995	5109
8	7962	7927	8125	8312	8516

Table 4

Number of Triangles* in Highest Level of Detail (Tolerance = 10 meters)					
AOI	DeFloriani-1	Option 1	Option 2	Option 3	Option 4
1	1741	1852	1858	1942	1935
2	2474	2330	2380	2442	2452
3	1547	1353	1309	1439	1470
4	1211	1123	1111	1196	1237
5	3185	3072	3062	3127	3135
6	2318	1979	1992	2167	2137
7	2883	2745	2769	2899	2901
8	4568	4418	4414	4436	4586

* compare to 10952 triangles in grid



(a)

(b)

(c)

Figure 3. Perspective views of AOI 1 modeled with (a) DTED, (b) DeFloriani's algorithm, (c) Adaptive Hierarchical Triangulation.

CONCLUSIONS

Adaptive hierarchical triangulation, presented in this paper, has the following advantages over other algorithms currently used. First, because our algorithm focuses on the topology of a surface, it reduces the number of triangles required to accurately approximate the surface and produces fewer long and slivery triangles within each level of detail. Second, our structure guarantees the accuracy of each level of detail. This may be easily extended to impose a polygon limit at each level. Third, our structure only retains important triangles, thereby reducing the total number of triangles that must be stored and searched. Fourth, the tree-like structure of our hierarchy is well-adapted to multiple resolution views, allowing smooth transitions between resolutions in animation. Because adaptive hierarchical triangulation pays attention to surface topology, this transition from low to high levels of detail will cause only minor terrain features to appear. Finally, adaptive hierarchical triangulation algorithm is fully automated, requiring only the area of interest and a series of tolerance levels to be defined. This algorithm can also be shown to run in $O(n \ln n)$ time. These advantages add up to a triangulation that provides great accuracy in a model that can be rapidly searched, rendered, and otherwise manipulated.

REFERENCES

Aho, A.V., Hopcroft, J.E. and Ullman, J.D., 1974. *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, Mass.

B. S. Baker, E. Grosse and C. S. Raferty, 1988. Nonobtuse triangulation of polygons, *Discrete Computational Geometry*, 3, 147-168.

Chew, L.P., 1989. Constrained Delaunay triangulations, *Algorithmica*, 4, 97-108.

Christensen, A.H.J., 1987. Fitting a triangulation to contour lines, *Proceedings of AUTO-CARTO 8*, 57-67.

Christiansen, H.N. and Sederberg, T.W., 1978. Conversion of complex contour line definition into polygonal element mosaics, *Proceedings of SIGGRAPH '78*, 187-192.

Clarkson, K.L., Tarjan, R.E. and Van Wyk, C.J., 1989. A fast Las Vegas algorithm for triangulating a simple polygon, *Discrete and Computational Geometry*, 4(5), 432-432.

DeFloriani, L., Falcidieno, B., Nagy, G., and Pienovi, C., 1984. A hierarchical structure for surface approximation, *Computers and Graphics*, 8(2), 183 - 193.

DeFloriani, L., 1989. A pyramidal data structure for triangle-based surface description, *IEEE Computer Graphics & Applications*, 9(2), 67-78.

Dennehy, T.G., and Ganapathy, S., 1982. A new general triangulation method for planar contours, *Proceedings of SIGGRAPH '82*, 69-74.

Dwyer, R.A., 1987. Faster divide-and-conquer algorithm for constructing Delaunay triangulations, *Algorithmica*, 2(2), 137-151.

Fekete, G., 1990. Rendering and managing spherical data with sphere quadtrees, *Proceedings of Visualizaion '90*, 176-186.

Fournier, A., and Montuno, D., 1984. Triangulating simple polygons and equivalent problems, *ACM Transactions on Graphics*, 3(2), 153 - 174.

Fowler, R.J. and Little, J.J., 1979. Automatic extraction of irregular network digital terrain models, *Proceedings of SIGGRAPH '79*, 199-207.

Garey, M. R., Johnson, D. S., Preparata, F. P. and Tarjan, R. E., 1978. Triangulating a simple polygon, *Information Processing Letters*, 7, 175-180.

Goodchild, M.F., 1989. Optimal tiling for large cartographic databases, *Proceedings of AUTO-CARTO 9* , 444-451.

Manacher, G.K. and Zobrist, A.L., 1979. Neither the greedy nor the Delaunay triangulation of a planar point set approximates the optimal triangulation, *Information Processing Letters*, 9, 31-34.

Mirante, A. and Weingarten, N., 1982. The radial sweep algorithm for constructing triangulated irregular networks, *IEEE Comuters Graphics & Applications*, 2, 11-21.

Preparata, F.P. and Shamos, M.I., 1985. *Computational Geometry*, Springer-Verlag, New York.

Scarlatos, L.L., 1989. A compact terrain model based on critical topographic features, *Proceedings of Auto-Carto 9*, 146-155.

Scarlatos, L.L., 1990(a). An automated critical line detector for digital elevation matrices, *Proceedings of the 1990 ASPRS/ACSM Annual Convention*, 43-52.

Scarlatos, L.L., 1990(b). A refined triangulation hierarchy for multiple levels of terrain detail, *Proceedings of the IMAGE V Conference*, 115-122.

Watson, D.F., 1981. Computing the n-dimensional Delaunay tessellation with applications to Voronoi polytopes, *The Computer Journal*, 167-172.

Watson, D.F. and Philip, G.M., 1984. Survey: systematic triangulations, *Computer Vision, Graphics, and Image Processing*, 26, 217-223.