CURJ

SPRING 2004 EXCERPT

CALTECH UNDERGRADUATE RESEARCH JOURNAL VOL. IV NO. I

ArtCenter College of Design

Original CURJ Design by Aniko Hullner Grau
©2004 CURJ. ALL RIGHTS RESERVED.
THIS ISSUE NOT FOR RESALE

www.curj.caltech.edu

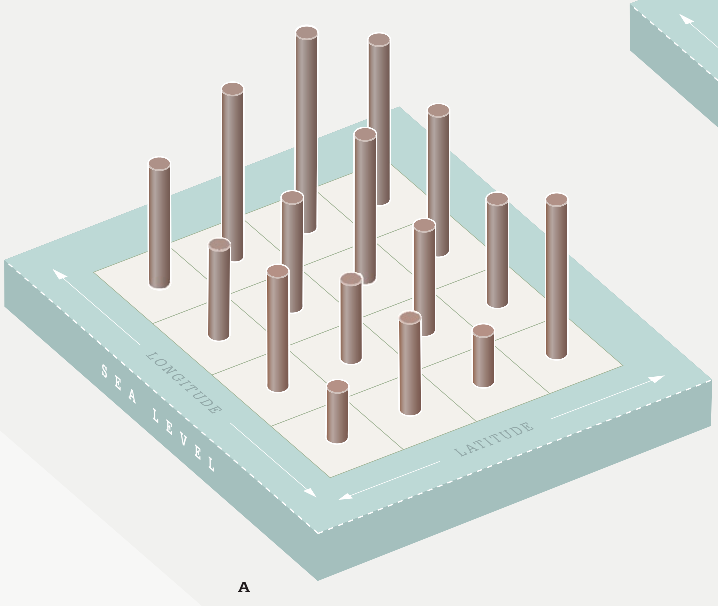
LEAPING OVER CHA

BY JOSEPH E. GONZALEZ

A NEW ALGORITHM FOR EVALUATING LINE-OF-SIGHT ON DIGITAL ELEVATION MAPS

NEW INTELLIGENCE INFORMATION INDICATES that several key operatives in a radical terrorist cell are currently meeting in an abandoned hut deep in the rugged mountains of northern Afghanistan. Five kilometers away, a Delta Force unit is deployed to capture the operatives. However, if the unit is detected by militants patrolling the mountain trails, the operatives will have sufficient warning to avoid capture. Driving Humvees equipped with mobile workstations, the Delta Force unit can rapidly move to the hut while evading the militants using a powerful new line-of-sight evaluation algorithm in development at NASA's Jet Propulsion Laboratory.

Rapidly and accurately assessing visibility between two points on large terrain maps is critical to the success of military simulations, real time terrain visualization, and the placement of wireless communications systems. Each application imposes different requirements on the speed and accuracy of LOS evaluation and the availability of hardware resources. Military simulations use large terrain maps, often spanning several countries at high resolutions (30 data points per kilometer), and require accurate visibility information for thousands of military platforms every second. These high demands push the current desktop computer architecture to its limits in memory and speed. However, using a technique which divides the problem into more manageable chunks, we can quickly and accurately compute line-of-sight. Consequently, a Delta Force unit could continuously determine what parts of the terrain militants can see and adjust their routes accordingly as the militants moved.



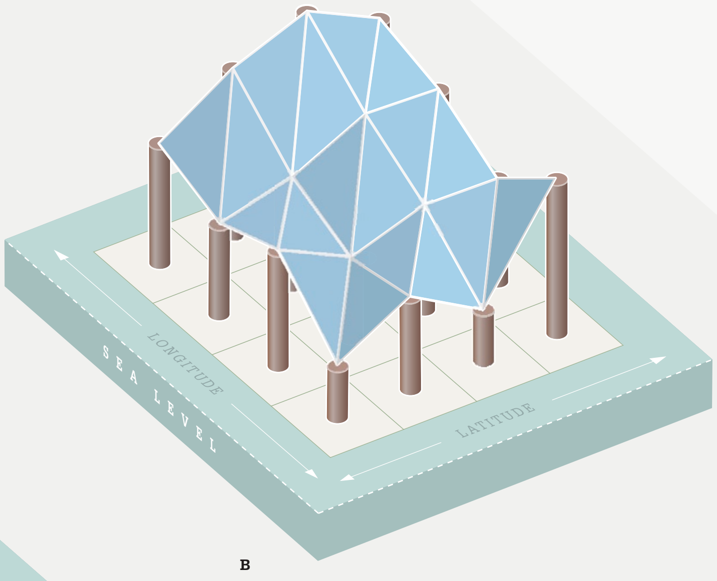


FIGURE 1 A DEM (A) describes the known elevation of the terrain relative to the geoid (sea level) on a regular grid. A regular triangulation (B), is used to define the terrain of unknown elevation between DEM posts.

SQUEEZING THE EARTH INTO A COMPUTER

The topography of the Earth is captured by satellites in the form of a digital elevation map (DEM). A DEM is simply a regular grid of elevation points called DEM posts. While the elevation of the terrain at individual DEM posts is well defined, the terrain between DEM posts is undefined. We chose to use a regular triangulation of the DEM, in which we draw straight lines between adjacent DEM posts. The straight lines form a regular tessellation of triangular planes. These planes compose a regular triangulation and define the terrain between the DEM posts [FIGURE 1].

When it is necessary to reduce the number of DEM posts or edges, an irregular triangulation is often used. Irregular triangulations use large triangles to represents regions of nearly coplanar DEM posts [FIGURE 2]. However, while large triangles reduce the number of DEM post they also may remove significant terrain features. Furthermore, it is difficult to find the optimal set of large triangles that best represents the terrain.

Although more complex interpolation techniques exist, they do not necessarily better represent the true surface of the Earth. Consequently, we assume that a regular triangulation of a DEM is the most accurate interpolation. The new algorithm for evaluating LOS uses the regular triangulation of a DEM as its model of the Earth's surface.

SEEING THE SLOW WAY

Military simulations currently use one of two major algorithms to evaluate LOS. The "telephone pole algorithm" compares the height of the LOS to the height of the surface model at uniformly spaced steps. If the LOS is below the surface at any step then LOS is obstructed. This algorithm has the advantage of faster computation through the reduction of the number of steps at the expense of decreased accuracy. To reduce the frequency of false results, the number of steps must be increased, thereby slowing the algorithm.

The second major line-of-sight algorithm traverses the edges of a piecewise planar model of the terrain. The speed and accuracy of this approach depends on the model of the terrain rather than the algorithm. In this algorithm, LOS is assessed by comparing the line connecting the two points to each edge it crosses. The algorithm indicates that the line-of-sight is blocked if the line falls below any edge. Likewise, it indicates that the line-of-sight is clear if it has reached an endpoint without passing under an edge of the terrain. While this method can be applied directly to a regular triangulation of a DEM, the large number of edge crossings makes it prohibitively slow in most applications. Consequently, an irregular triangulation if often used to increase evaluation speed at the expense of accuracy.

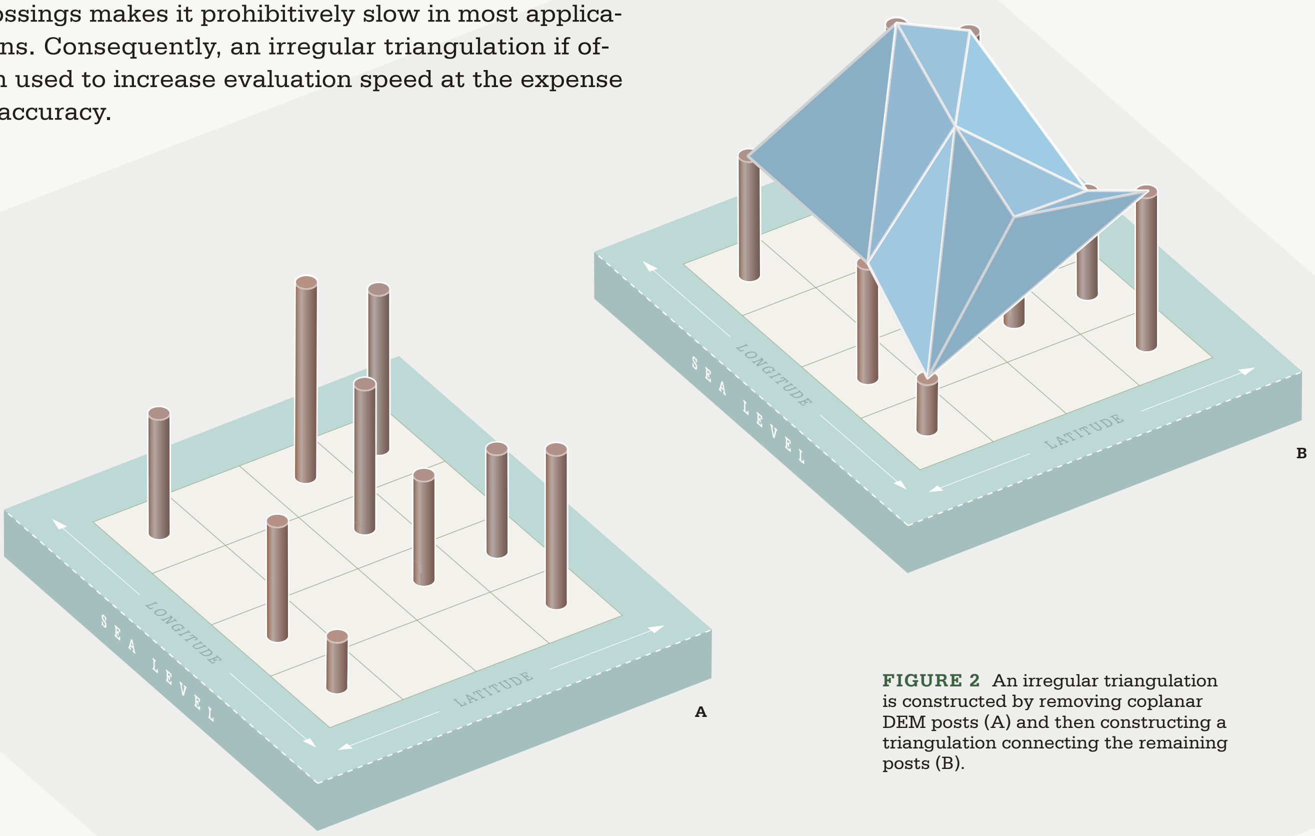
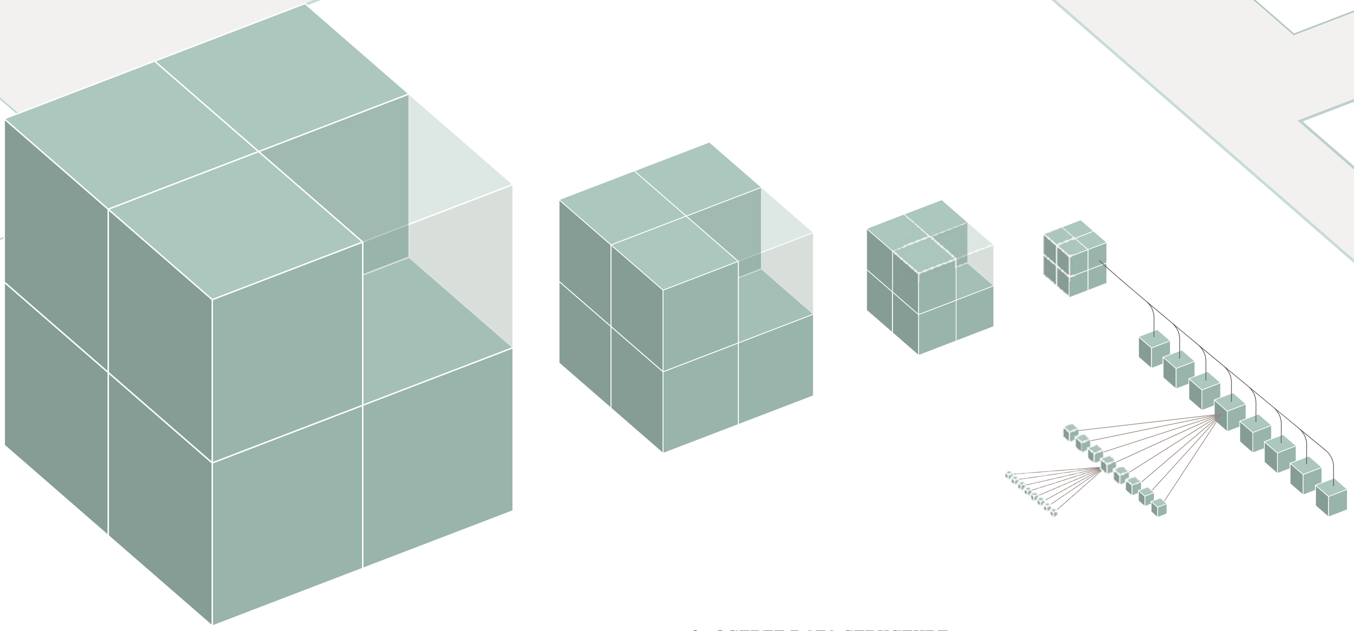


FIGURE 2 An irregular triangulation is constructed by removing coplanar DEM posts (A) and then constructing a triangulation connecting the remaining posts (B).


A OCTREE DATA STRUCTURE

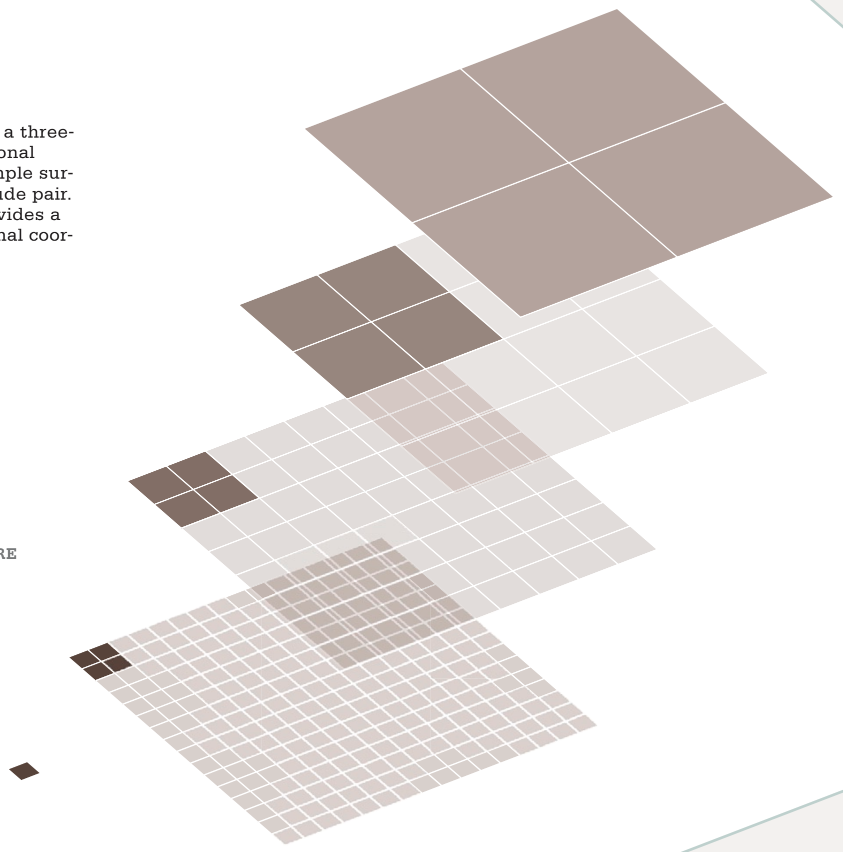
ILLUMINATING BOXES IN BOXES

Our new approach to evaluating LOS applies techniques originally developed to accelerate 3D scene rendering. To obtain exceptionally realistic images, rendering algorithms trace the paths of simulated rays of light through complex three-dimensional scenes containing millions of objects. Without optimization, ray-object intersection must be evaluated for every object in the scene, making a detailed, full feature animation like "Finding Nemo" unfeasible to produce.

Fortunately, there are powerful optimization techniques that reduce the number of ray-object intersection evaluations by grouping objects within simplified bounding boxes. Moreover, a ray that does not intersect a bounding box will not strike any of the objects contained within that box. Consequently, many costly intersection evaluations may be avoided by making one simple box evaluation. Our line-of-sight problem is a special case of the ray tracing problem in which the line-of-sight is the "ray" and we want to determine if that ray intersects with the terrain. Therefore, by draw bounding boxes on our map we can also avoid unnecessary intersection evaluations.

The number of intersection evaluations may be further reduced by grouping the bounding boxes within bounding boxes. In 3D scene rending, the scene space is repeatedly divided into smaller cubical volumes and then reassembled into an octree data structure in which every higher level, representing a box, contains eight smaller levels, each of which contains eight smaller levels and so forth until the data is indivisible [FIGURE 3A]. While this approach could be applied to the LOS evaluation problem, the regular triangulation is a simpler two-dimensional surface with only a single elevation for a given latitude-longitude pair. The two-dimensional analog to the octree is the quadtree, which recursively divides a space into quadrants [FIGURE 3B]. The quadtree exploits the planar form of the DEM by dividing the bounding boxes into four smaller boxes along the latitudinal and longitudinal axes.

FIGURE 3 An octree data structure (A) recursively divides a three-dimensional space into octants. Unlike most three-dimensional scenes encountered in computer graphics, the DEM is a simple surface with only a single elevation for a given latitude-longitude pair. Therefore, a quadtree data structure (B) that recursively divides a two-dimensional space labeled by latitudinal and longitudinal coordinates into quadrants is better suited.



GROWING A TREE FROM THE LEAVES

Quadtree spatial partitioning is applied by dividing the DEM along the posts into progressively smaller quadrants until each quadrant contains only 4 posts. Quadrants that contain four DEM posts (and that are therefore indivisible) are represented by four leaves containing the elevations of the four DEM posts. Any quadrant that may be further divided (that is, contains more than 4 posts) is represented as a node containing the highest elevation in that quadrant.

To ensure that the DEM is always evenly divisible, each dimension (rows and columns) of the DEM is enlarged to the nearest integer that can be expressed in the form $2^n + 1$, where n is an integer. However, no data is stored for leaves that are outside of the DEM or for nodes that do not intersect the DEM.

By starting at the leaves, we only need three comparisons per quadrant to determine the maximum elevation within that quadrant rather than comparing all the DEM posts within the quadrant. (Starting with a square 2 posts by 2 posts wide, we need three comparisons to find the highest post in that square. We can group all of the points on the grid into non-overlapping 2×2 squares and store the maximum elevation in each of them. Then, we can group four neighboring 2×2 squares into a 4×4 block. The highest elevation in the 4×4 block is the highest of the four “maximum elevations” of each of the four 2×2 blocks, and since there are only four numbers to compare, this only takes three comparisons.) As the quadrants get larger, this dramatically reduces the number of comparisons since the maximum elevation of a large quadrant can be determined from the maximum elevations of the four quadrants that compose it.

The need for memory pointers (common to tree data structures) was also eliminated by using a spatial mapping function. This function maps any quadrant of space on the earth to the unique location in memory at which the maximum elevation for that quadrant is stored. Storing the pointers would have resulted in a quadtree data structure 14 times larger than the original DEM. By using the spatial mapping function the final quadtree data structure was only 33% larger than the original DEM.

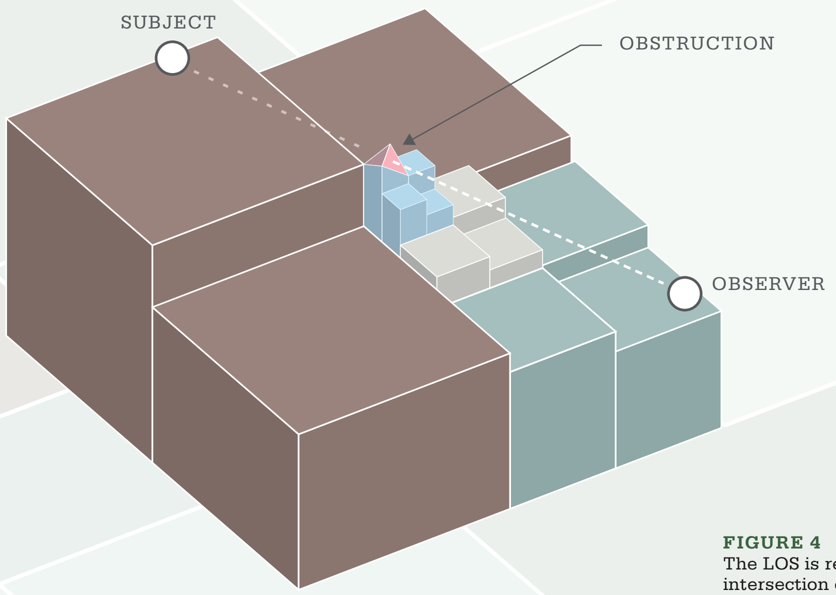


FIGURE 4
The LOS is reported to be obstructed at the the first intersection of its path with a leaf-level triangulation.

IGNORING IRRELEVANT POINTS

The new algorithm quickly evaluates line-of-sight by only checking the points that could affect the answer. If you are trying to evaluate LOS across a valley and you know that every point in the valley is lower than your line, there is no reason to check every point within the valley. By searching the largest bounding boxes first, it is possible to reject LOS-terrain intersection over large regions of a map in relatively few steps. We only search through the more detailed elevation information within boxes that the LOS intersects. Moreover, the terrain is examined at the lowest possible detail necessary to reject intersection.

The algorithm starts by determining if the line-of-sight path intersects any of the top-level bounding boxes. If it does not, then it reports that the path is unobstructed. If the path intersects any of the top-level boxes, then the algorithm repeats for each of the intersecting boxes by checking for intersections between the LOS and each of the lower boxes. If the algorithm ever finds that the LOS intersects a leaf (one of the pairs of triangles in the regular triangulation), then it reports that the path is obstructed; otherwise, it will report that the path is unobstructed. [FIGURE 1]

Because the DEM is always divided into quadrants, the number of computations in this approach is logarithmic in the number of DEM edges along the LOS. In comparison, the complexity of the edge traversal technique is linear in the number of steps and the telephone pole technique has constant time complexity. Therefore, this new technique is faster than the edge traversal technique and slower than the telephone pole technique but doesn't suffer from the telephone pole algorithm's inaccuracy.

To compensate for the curvature of the Earth, we added a few additional values to the dimensions of the bounding boxes. To adjust for the convex top of the bounding boxes, we added a precomputed offset. The distance between a LOS and the surface of the earth increases along the LOS away from the point of tangency. An additional 2nd order approximation of this curvature is subtracted from the top of the box. Prior to subtraction, the second order offset is multiplied by a scaling factor to compensate for the refractive properties of the atmosphere.

Even with this faster LOS algorithm, the size of the maps involved in these computations poses a significant problem. For large data sets the DEM alone can exceed several gigabytes of memory. To make this algorithm practical for large data sets on desktop computers, we need to carefully manage the way we interact with the data. Because each level of the tree is a linear array of binary elevation data describing a relatively smooth surface, compression techniques may be feasible. However, rather than researching techniques to compress the data structure we are focusing on transferring only the necessary information to low-latency memory (RAM) while leaving large portions of the unread data structure on high-latency memory (hard disk). It will often be possible to determine when

“The terrain is examined at the lowest possible detail necessary to reject intersection.”

LOS is unobstructed by reading from the first few levels of the quadtree data structure. These levels take insubstantially small amounts of memory relative to the entire DEM and can easily be stored in RAM. Data from lower levels is needed only when the LOS intersects a bounding box. Because military simulations of events are usually confined to small regions rather than across the whole map, information describing only these regions may be easily copied to RAM.

FASTER AND MORE ACCURATE

We wrote a prototype implementation of the algorithm and the data structure generator. A quadtree data structure was constructed from a 3-second resolution DEM of Korea. By replaying a previous military simulation running the new quadtree algorithm and again with an edge traversal algorithm using a TIN generated from the same DEM, we assessed the discrepancy between the two techniques. Approximately 19.7% of the 65,000 LOS queries resulted in a discrepancy between the two algorithms, which was probably due to data lost in the TIN description of the terrain. There was a 4% loss in performance of the overall simulation when running the new algorithm, but this was probably due to the configuration of our systems and delays in our computer network.

Borrowing from concepts discovered in the field of ray tracing, we have developed a new algorithm for evaluating LOS. By using a quadtree data structure, the quadtree algorithm can rapidly search for the intersection of the LOS and the terrain. We have reduced the complexity of this search from linear time to logarithmic time while only increasing the size of the data structure by 33%, giving an enormous efficiency gain for very large maps. With the future development of compression and memory management techniques that exploit the new quadtree data structure, we believe that it will be possible to use extremely large high-resolution DEMs, which would make it efficient for small portable computers to easily evaluate line of sight on the fly. A computer with a DEM and this algorithm might some day direct a soldier out of harms way, help a military commander plan a strategy, or assist civilian communications engineers in determining the optimal locations for cellular antennas. C

Joseph E. Gonzalez is a second year undergraduate in Computer Science at the California Institute of Technology. He would like to thank his mentor Robert Chamberlain, his colleagues at JPL, Professor Alan Barr, and the SURF program.

FURTHER READING

1 T. Kay, J. T. Kajiya. Ray Tracing Complex Scenes. *ACM SIGGRAPH* **20(4)**, 269–268 (1986).

2 R. F. Richbourg, R. J. Graebener, T. Stone, K. Green. Verification and Validation (V & V) of Federation Synthetic Natural Environments. Proceedings of the 2001 Interservice/Industry Training, Simulation, and Education Conference (2001).

3 S. M. Rubin, T. Whitted. A Three Dimensional Representation for Fast Rendering of Complex Scenes. *Computer Graphics* **14(3)**, 110–116 (1980).