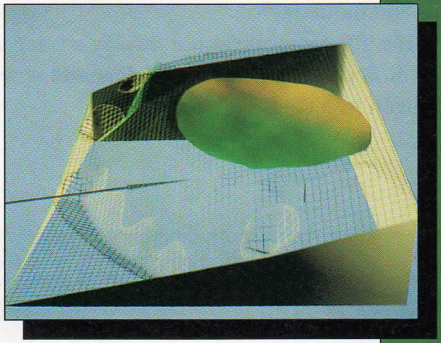
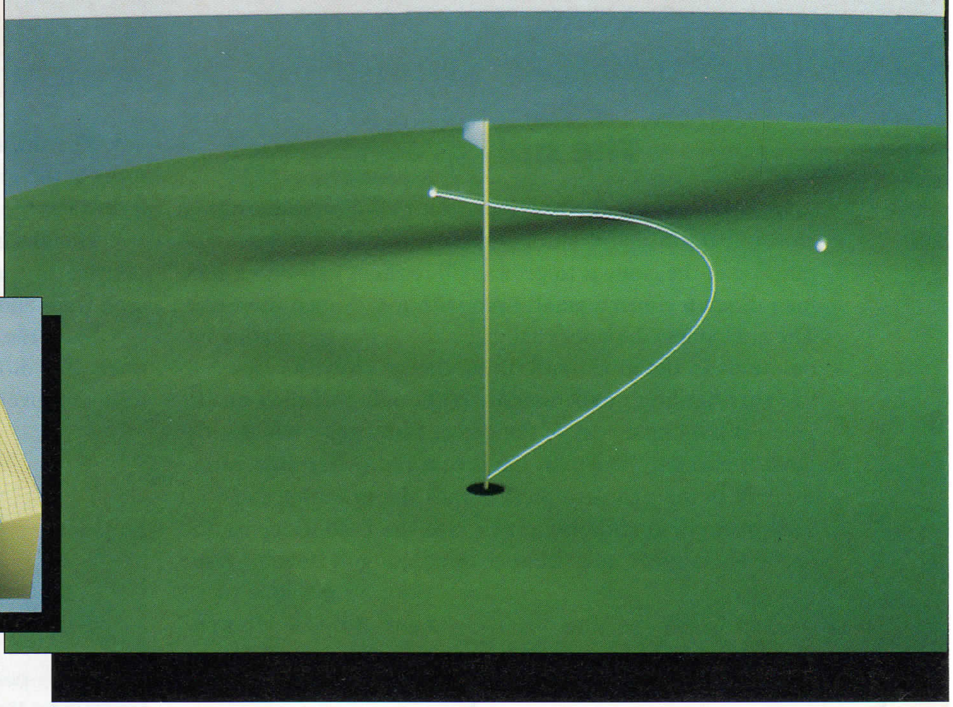




Golf Green Visualization

William E. Lorensen and Boris Yamrom
General Electric Company

Our system uses shaded polygonal models to help television viewers appreciate the complex topography of tournament golf greens. The system also models golf ball trajectories and putting difficulty.

Spectators of sporting events often have difficulty perceiving the movement of players, cars, horses, or balls. Television has successfully enhanced viewer perception of some sports by providing unusual camera placements, close-up views, instant replays, and slow-motion replays. But television enhancements can also distort a viewer's perception. For example, restricted camera placements in baseball, most notably the center-field camera, fail to communicate the trajectory of a baseball pitch. In a horse race, the head-on camera shot of the stretch run compresses the distances between horses. And in golf, diffuse overhead sunlight washes out the undulations in a green.

Computer graphics can help. Although much computer graphics research focuses on the creation of realistic models, some recent work has focused on nonrealistic rendering techniques that enhance the understanding of complex phenomena.[1,2]

In this article we address viewer perception of golf green topography and how that topography affects putting. We combined conventional computer graphics and numerical analysis techniques with application-specific modeling and analysis algorithms to enhance a viewer's understanding of the golf green. The presentations our system produces make watching a golf match on television more entertaining.

The golf green

Golf greens are the portions of a golf course that have closely cut grass and undulations surrounding a hole. The purpose of the game is to get a small resilient ball with a 1.68-inch diameter into a small hole with a 4.25-inch diameter. The golf course designer challenges the golfer by creating undulations in the green and tilting the green with respect to the surrounding environment. Although golf play on the green is but one aspect of the game, many golfers find it the most frustrating. Once on the green, the golfer must strike the ball in the proper direction with the proper speed so it rolls over the undulating green into the hole. Because the challenge is great, television coverage of golf devotes more than 50 percent of its time to the players' putting.

For several reasons, television fails to bring viewers an appreciation of putting's complexity:

- Camera placement is restricted to locations that will not distract the golfer. Portable cameras can be moved behind but not in front of the golfer. Other cameras, placed on towers, present a high view of the green but are not high enough to give a full overhead view.
- The diffuse overhead sunlight washes out the undulations in the green. Artificial lighting cannot be used because it distracts the players.
- On television the often subtle tilt in a green's orientation relative to the horizon is camouflaged by the surrounding sand traps and mounds.
- Since the TV cameras cannot display a trace of the ball's track, it is difficult to see how or understand why a golf ball follows a given trajectory.

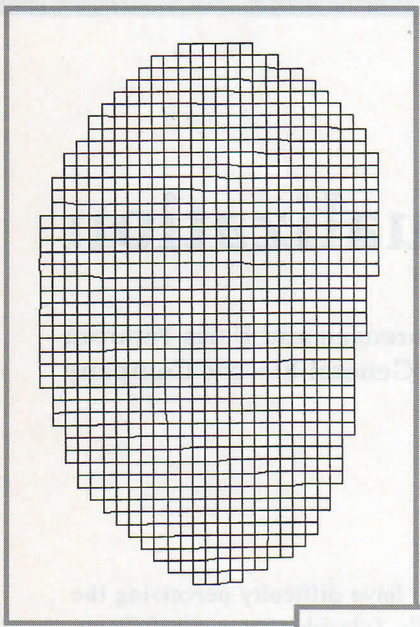


Figure 1. Interior polygons generated by connecting neighboring points on the square grid.

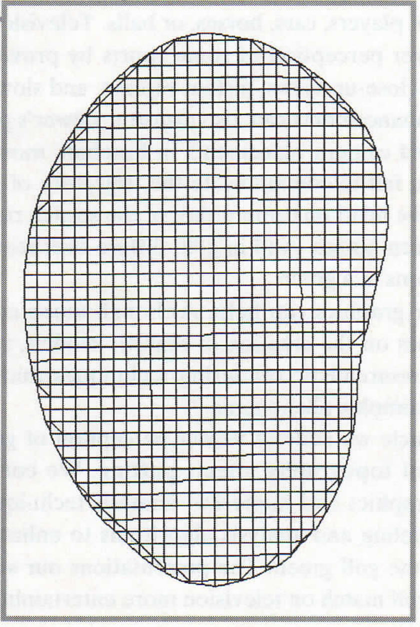


Figure 2. Interior and perimeter polygons.

Golf terms

birdie—one stroke under par

caddy—person who assists a golfer, especially by carrying the golf clubs

par—the standard number of golf strokes needed to complete a hole or course in expert play

pin—the pole bearing a flag to mark a hole in golf; always located on a green

Golf green visualization

Our approach to enhancing a viewer's perception of a golf green uses geometric models of greens, mathematical models of putting, and computer graphics presentations of the greens and putts.

First, to build an accurate model of the green topography, we have surveyors measure elevations for the greens that we want to present to the television audience. Then, we use modeling software to create polygonal models of the greens. A surveyor obtains the pin placement for the day's round and the location of each player's ball. Mathematical models of the golf ball motion predict its path along the green and into the hole. During the tournament broadcast, an operator moves the computer graphics camera, adjusts lighting, and controls the putt simulations. The sports director broadcasts the computer-generated sequences that will interest viewers and enhance the announcers' commentary.

Surveying the green

Before the tournament, a licensed surveyor uses a theodolite to acquire green perimeter data and green elevation data within the green. We use two approaches:

1. Gather the elevation data on a uniform *x-y* grid. For our experiment, we sampled the green at two-foot intervals. Although this simplifies the geometric modeling step, it takes up to a full day to survey one green.

2. Gather the elevation data on a nonuniform *x-y* grid, taking more samples in undulating areas. This approach takes about half the time of the uniform method, but it complicates modeling.

After the survey is complete, the surveyor furnishes a computer disk with *x-y elevation* data for the green's interior and ordered *x* and *y* coordinates of the green's perimeter. The surveyor retains records of reference points to locate the hole and ball on the modeled green.

In the future, we might use more sophisticated methods, such as close-range photogrammetry.[3]

Modeling the green

The system requires models for the ball, hole, pin, and green. The polygonal description of the golf green includes three pieces:

• Interior: For uniform data, the system generates the polygons by connecting neighboring points on the square grid, as shown in Figure 1. For nonuniform data, a Delaunay triangulation[4] of the data provides the necessary polygons. This triangulated data can be resampled to provide a uniform distribution of data.

• Perimeter: The surveyor always includes points outside the perimeter so the polygons can be trimmed by the perimeter data. We fit parametric cubic splines to the perimeter data and use these splines to clip the polygons that straddle the perimeter. Figure 2 shows the results.

• Skirt: To help viewers appreciate the green's inclination, we produce a skirt around it. The skirt consists of vertical polygons that start at the green perimeter and extend downward.

We use the generated geometric model, shown in Figure 3, for rendering and retain the uniformly spaced survey data for later putt simulations. The greens we have modeled are typically 60 × 90 feet, requiring 1,000 to 2,000 polygons.

Locating the ball

Accurate locations for balls on the green are required during the tournament. We could compute the position of a point (a ball) in the environment using two measurements from two different theodolite positions. Alternatively, we could use a measurement of two angles, vertical and horizontal, plus range data obtained from the ultrasound reflected from the object back to the theodolite.

Unfortunately, using two surveyors is expensive. Also, golf balls scatter ultrasound in all directions, reflecting too little energy back to the surveyor for reliable measurements. However, we have the green represented in the computer as a polygonal surface. Thus, we need only one measurement of vertical and horizontal angles from a known position to calculate the intersection point between the ray from this position and the green. Although the precision of this method

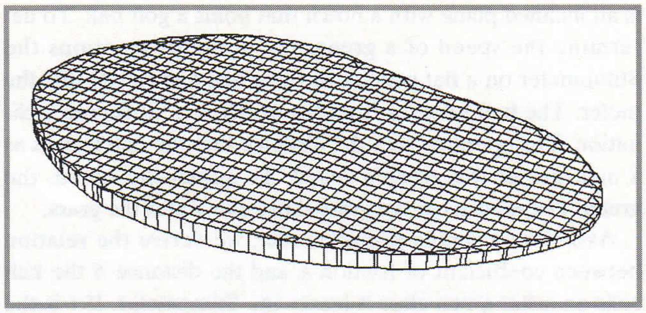


Figure 3. Wireframe model of a green.

decreases as the elevation of the observation point decreases, the surveyor is normally located on a television tower some 20 feet above the ground and 60 feet from the center of the green.

Modeling the putt

In providing a putt trajectory model, we want to help viewers understand how the green's topography affects the ball—not create a putting machine. Therefore, we made trade-offs to predict the ball trajectories using a reasonable amount of computer time.

When putted, a golf ball exhibits two types of interactions with the green surface: sliding and rolling. Sliding occurs during the early portion of the putt. Gradually, the friction force applied to the golf ball in the direction opposite to the motion direction causes the ball to rotate. The sliding speed decreases and the rotation speed increases until the ball starts rolling without sliding. The rolling coefficient of friction is much lower than the sliding coefficient of friction. Both values depend on many parameters, such as moisture content, height, and growth direction of the grass. For a perfectly uniform ball, the rolling begins when the ball speed decreases to 5/7ths of the initial speed independently of the value of the coefficient of friction.[5]

In reality, the modern golf ball is not uniform. It is composed of several (usually three) layers of materials, with the most dense material in the center. Therefore, its moment of inertia is slightly less than the moment of inertia of the uniform ball of the same weight, and it starts pure rolling at a speed greater than 5/7ths of the initial speed. Because balls are produced by different manufacturers, we cannot know ahead of time the moment of inertia of a particular ball. The United States Professional Golf Association (USPGA) regulations state only that the ball should be 1.68 inches in diameter and weigh 1.62 ounces.[5]

For our analysis, we assume uniform golf balls. To further simplify the simulation, we assume that the ball always slides with two coefficients of friction: one for the first, sliding phase of motion and another for the second, rolling phase.

To simulate a putt, the system needs the coefficient of friction of the green surface. Golf course managers use a Stimpmeter to measure the speed of a golf green. The Stimpmeter is an inclined plane with a notch that holds a golf ball. To determine the speed of a green, the operator positions the Stimpmeter on a flat portion of the green and slowly lifts the meter. The ball releases from the notch at a 20-degree inclination. The operator uses the distance that the ball travels as a measure of the green's speed. To adjust the speed, the greens keeper can cut the grass shorter or water the grass.

Assuming a simple sliding model, we derive the relation between coefficient of friction k and the distance S the ball rolls on a flat green after it leaves the Stimpmeter. If $\mathbf{v}$ is the initial velocity of the ball, then

$$k = \frac{|\mathbf{v}|^2}{2gS}$$

where $|\mathbf{v}|$, the speed of the ball as it leaves the base of the Stimpmeter, is 6.0 feet per second,[6] and g is the acceleration of gravity.

To simulate a putt, we must solve two problems. The first, an initial-value problem, uses an initial position, a velocity, and direction to predict the ball's path. The second, a boundary-value problem, uses an initial location, final position, and final speed to calculate the initial speed and direction. Both problems require the formulation of differential equations for the ball sliding on a faceted surface.

Figure 4 shows how the sliding of a body on an inclined plane is controlled by three forces: the body's weight $\mathbf{P}$ directed down vertically, the reaction of the plane $\mathbf{N}$ directed along the normal to the plane n, and the friction force $\mathbf{F}_{fr}$ directed opposite to the body's velocity $\mathbf{v}$. Because Figure 4

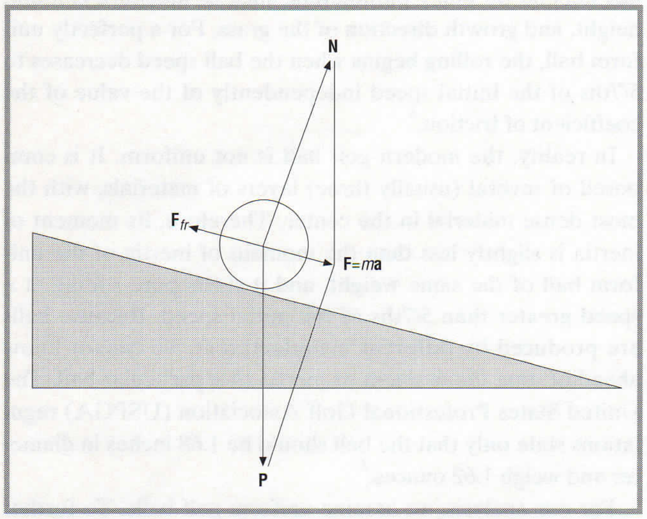


Figure 4. Forces controlling sliding on an inclined plane.

shows a 2D projection of a 3D scene, it hides the fact that $\mathbf{v}$ is not necessarily collinear with $\mathbf{P} + \mathbf{N}$. Since there is no motion in the direction perpendicular to the inclined plane, the normal reaction $\mathbf{N}$ is equal to $-(\mathbf{P} \cdot \mathbf{n})\mathbf{n}$. The friction force $\mathbf{F}_{fr}$ is equal to $-k|\mathbf{P} \cdot \mathbf{n}|\mathbf{v}/|\mathbf{v}|$, and $\mathbf{P} = m\mathbf{g}$, where m is the mass of the body and $\mathbf{g}$ is the acceleration of free fall. From Newton's second law it follows that the acceleration $\mathbf{a}$ of the body is defined from the relation

$$m\mathbf{a} = m\mathbf{g} - (m\mathbf{g} \cdot \mathbf{n})\mathbf{n} - k|m\mathbf{g} \cdot \mathbf{n}|\mathbf{v} / |\mathbf{v}|$$

or

$$\mathbf{a} = \mathbf{g} - (\mathbf{g} \cdot \mathbf{n})\mathbf{n} - k|\mathbf{g} \cdot \mathbf{n}|\mathbf{v} / |\mathbf{v}|$$

where $\mathbf{g} = (0, 0, -g)$, and the z axis is up. For

$$\begin{aligned}\mathbf{v} &= (v_x, v_y, v_z)\\ \mathbf{a} &= \mathbf{v}' = (a_x, a_y, a_z)\\ \mathbf{n} &= (n_x, n_y, n_z)\end{aligned}$$

we can rewrite the last vector equation in coordinate form as a system of three equations:

$$\begin{aligned}a_x &= gn_zn_x - kgn_zv_x / |\mathbf{v}|\\ a_y &= gn_zn_y - kgn_zv_y / |\mathbf{v}|\\ a_z &= gn_zn_z - kgn_zv_z / |\mathbf{v}| - g\end{aligned}$$

We introduce auxiliary variables to transform this system of nonlinear differential second-order equations to first-order system. We assume that the independent variable is time t and the position of the body is

$$\mathbf{x}(t) = (x_1(t), x_2(t), x_3(t))$$

For the velocity (v_x, v_y, v_z), we introduce variables x_4, x_5, and x_6. Then, we can rewrite the system as a system of six equations:

$$\begin{aligned}x_1' &= x_4\\ x_2' &= x_5\\ x_3' &= x_6\\ x_4' &= gn_zn_x - kgn_zx_4 / \sqrt{x_4^2 + x_5^2 + x_6^2}\\ x_5' &= gn_zn_y - kgn_zx_5 / \sqrt{x_4^2 + x_5^2 + x_6^2}\\ x_6' &= gn_zn_z - kgn_zx_6 / \sqrt{x_4^2 + x_5^2 + x_6^2} - g\end{aligned}$$

The normal $\mathbf{n}$ is a constant along the facet. However, to accommodate smooth transitions from facet to facet, we calculate the normals as an average between corner normals of the facet. The corner normals are the average of the normals of each polygon using the corner. (This is exactly the same averaging done in Gouraud shading.) As a result, we get continuous functions on the right side and can use numerical methods to solve these equations. The exact equations for a body sliding on a curved surface are more complicated, but the precision of our geometry model does not warrant more precise dynamic modeling. Visual comparisons between sim-

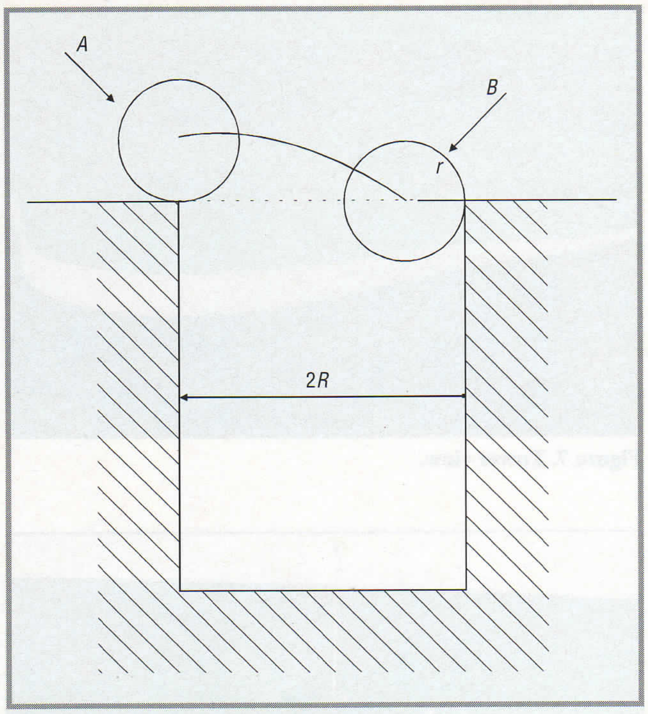


Figure 5. Ball dropping into a cup.

ulated trajectories and actual putts demonstrate that our approximation is satisfactory. Given the ball's initial position on the green and its velocity, we can solve the above system. The solution represents a ball's trajectory $\mathbf{x}(t)$ on the green.

It is more interesting and challenging to calculate what the right velocity should be if the ball's initial position and its final destination, the cup, are given. We can solve this boundary-value problem using a shooting method.[7] Before we describe the procedure in more detail, we must clarify the terminal boundary condition. The ball can fall into the cup if its trajectory passes close to the cup's center and its speed is small enough to allow the ball to drop inside. We use a heuristic procedure to specify the speed threshold. Figure 5 shows a ball approaching the edge of the cup with speed $|\mathbf{v}|$. Subject to free fall, the ball will reach the point B with speed

$$|\mathbf{v}| < v_{\text{threshold}} = \sqrt{\left(\frac{g}{2r}\right)}(2R - r) = 0.36 \text{ feet per second}$$

where g is the acceleration of free fall, $2R$ is the cup diameter, and r is the ball radius. If the ball approaches the hole along a chord less than the diameter of the hole, its threshold will be less than $v_{\text{threshold}}$. The threshold goes to zero when the ball approaches the hole at its tangent. A simpler solution uses one average threshold, 0.18 feet per second, for all possible approaches to the hole.

We can use the initial-value problem to solve the boundary-value problem with a shooting method,[7] according to the following algorithm:

1. Using the initial-value problem solver, make a first guess by shooting directly at the cup. Take two more shots, varying the speed by 1 percent and direction by 1 percent. In all three cases, calculate the trajectory up to the point where the speed becomes less than the threshold value.
2. Compute the distances from the center of the cup to the endpoints of all three trajectories. Use these three distances together with the three speed and direction values to compute improved speed and direction values. The Newton-Raphson method[8] is used here.
3. Repeat steps 1 and 2 with the new guess and direction until the trajectory approaches the cup center within a distance less than the cup radius.

This algorithm does not guarantee convergence to a solution, so we devised a more robust procedure. It too has limitations. Some combinations of ball and cup position preclude a solution. However, USPGA rules limit cup placement to nearly flat regions of the green, thus reducing the number of impossible putts.

In the robust procedure, we denote the distance from the end of the trajectory to the cup's center by D_m, where m is the iteration number in steps 1 and 2. We let s be the scaling of the green surface $z = s$ * elevation(x, y) in the z direction. The real green has the scaling factor $s = 1$. For $s < 1$ the green is "flattened," and for $s > 1$ the green is "stretched."

The revised algorithm starts with the scaling set to 1. If it fails to converge to a solution in less than a fixed number of iterations (60, for example) or if D_m exceeds a threshold (10 feet, for example), the algorithm flattens the green by one half between its current scaling s_{current} (initially set to 1) and previous scaling s_{previous} (initially set to 0). The procedure tries to solve the problem for these conditions. If it succeeds, it uses the previous larger scaling with the initial conditions equal to the found ones; if not, it continues to flatten the green. This procedure works even if the ball leaves the green. The elevation of points off the green is set to 0, smoothly varying at the boundary of the green.

Visualizing the green and putts

The system provides a variety of graphics capabilities to enhance the viewer's perception of golf green action:

- Shaded rendering: Gouraud shading applied to the polygonal facets of the green model, along with adjustable light sources, provides an enhanced view of the green topography. Low-elevation light sources increase the perception of subtle undulations.
- Camera controls: Simple camera controls allow the operator to move either the look-at or look-from points in a circle. Figure 6 shows the six ways to move a camera by rotating its look-from, look-at, and view-up vector. Azimuth and elevation are rotations of the look-from point, while yaw and pitch are rotations of the look-at point. Roll rotates the view-

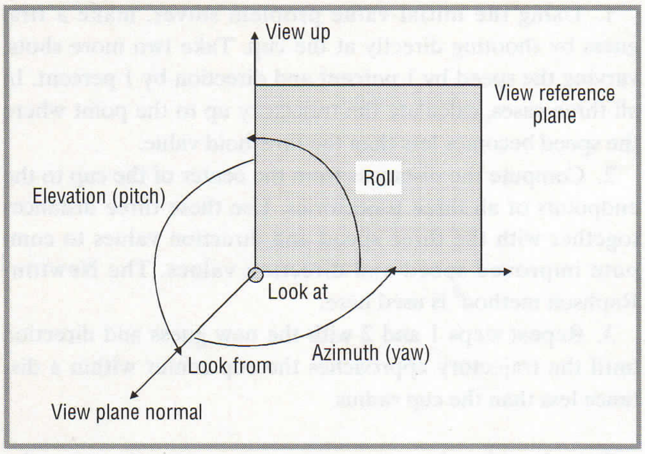


Figure 6. Camera controls.

up vector about the view plane normal. In this application, we withhold roll from the user interface. For rapid camera changes during a televised match, we provide buttons that move the camera to green-specific tower and fairway views, as shown in Figure 7. The tower view parameters are calculated by superimposing the synthetic green over the TV camera view from the camera tower. Zooming controls let the operator move the camera in and out along the view plane normal by a specified ratio.

• Camera location: A prominent feature of any 3D computer graphics application is the flexibility and usability of the viewing mechanism.[9] We chose a camera model because of its versatility and intuitive user interface. However, the camera model is so flexible that an operator can lose the green. To prevent this, the system calculates a default view any time a ball or the pin moves. The default view, shown in Figure 8, places the camera behind the ball, looking at the pin, about three feet above the green. With the look-at point halfway between the ball and the hole, the operator can move quickly from the golfer's point of view to the caddy's behind the pin.

• Animation: The operator can start a continuous azimuthal movement of the camera and change other viewing and analysis parameters while the animation runs. Typically, the operator zooms in and out and changes the camera elevation. This feature simulates a fly around the green, much like the helicopter camera shots often seen on television.

• Exaggeration: Scaling the green's elevation exaggerates any undulations in the green. Figure 9 shows how even the slightest change in elevation creates a mountain range.

• Trajectory display: The simulated putt is displayed as a line traveling along the green's surface, as a series of balls tracking the trajectory, or as an animation of the ball rolling, as shown in Figure 10. While solving the boundary-value problem, the system shows the intermediate guesses. Although this feature would not interest a television viewer, the adaptation of the shooting method solutions to the green's topography is interesting to the scientist.

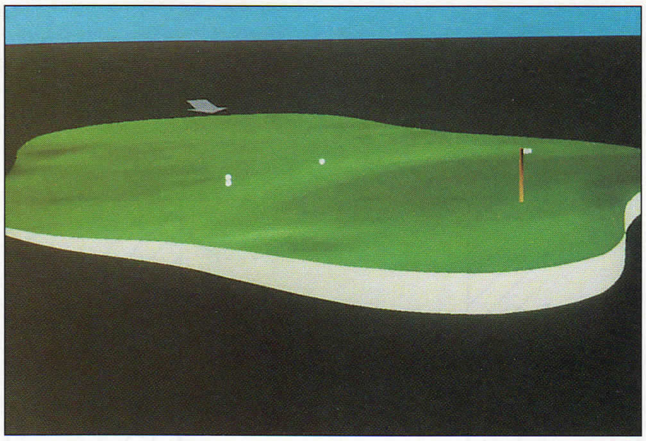


Figure 7. Tower view.

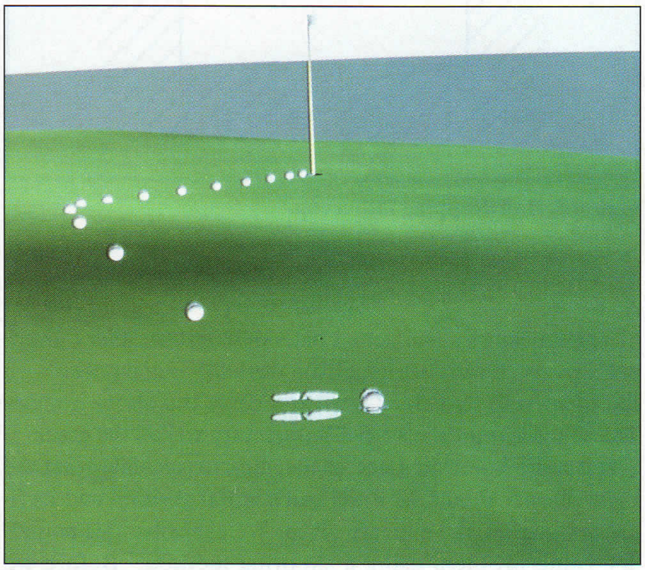


Figure 8. The golfer's view.

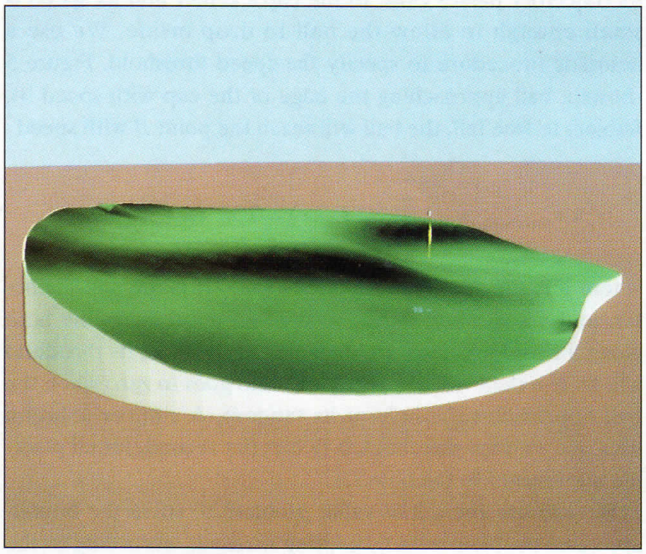


Figure 9. Exaggerating the green.

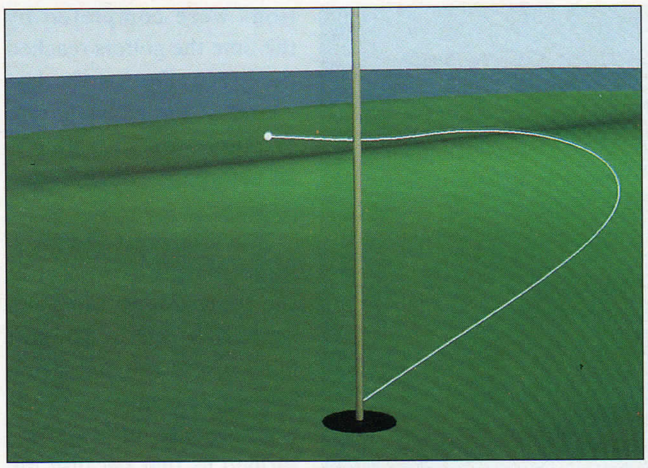


Figure 10. Trajectory of a simulated putt.

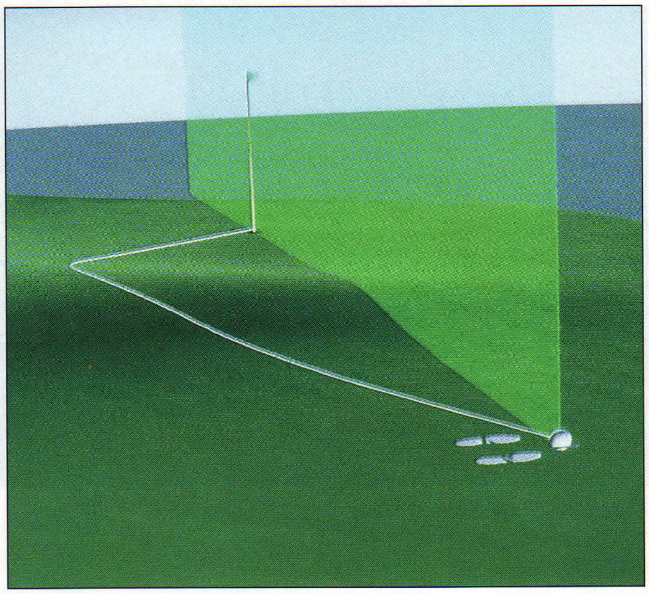


Figure 11. Vertical plane.

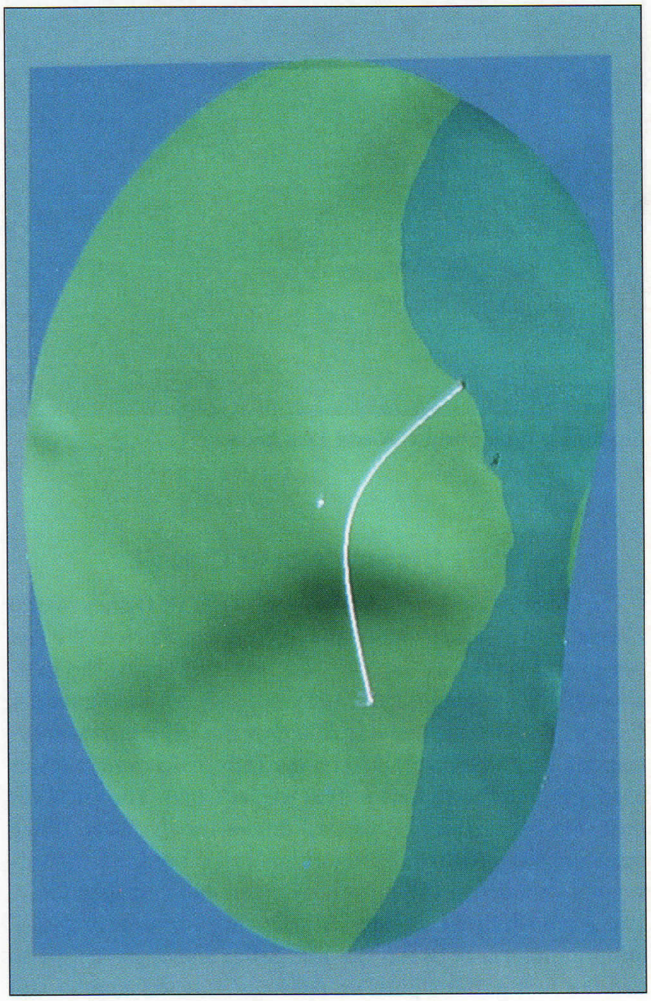


Figure 12. Horizontal plane.

• Vertical plane: Figure 11 shows a transparent vertical plane passing through the ball and hole positions to enhance the viewer's appreciation for the break of the ball.

• Horizontal plane: To highlight the relative elevations of different portions of the green, the operator can raise and lower a transparent horizontal plane, as shown in Figure 12. This flooding effect is especially useful to show subtle elevation changes.

• Putt difficulty maps: A putt difficulty map shows the relative difficulty of putting from each point on the green to a given pin location. The system assesses the difficulty of putting from a point as follows: First it solves the boundary-value problem for that point. Then it varies the initial speed and direction by small amounts and solves the initial-value problem for each variation. Difficulty is the average distance these putts lie from the hole. This amounts to a sensitivity analysis of the putt: If slight changes in the initial-value problem result in large variations from the perfect putt (the boundary-value solution), then a putt from this point is more difficult than another that has less variation. As Figure 13 shows, we apply this variation as a color to the vertex of each point in the geometric model and render the Gouraud-shaded polygons with interpolated colors. Difficulty varies from easy (green) to hard (red). Of course, a pin location change requires a new difficulty map.

• Quantitative results: As a by-product of the analysis, we obtain distance to the hole, elevation above or below the hole, and amount of break for each ball. These accurate values in themselves are interesting to the viewer. The system calculates the break of the ball along its path by finding the maximum deviation of the ball's path from a straight trajectory.

Results

We have experimented with the system at two golf tournaments broadcast by the National Broadcasting Company (NBC).

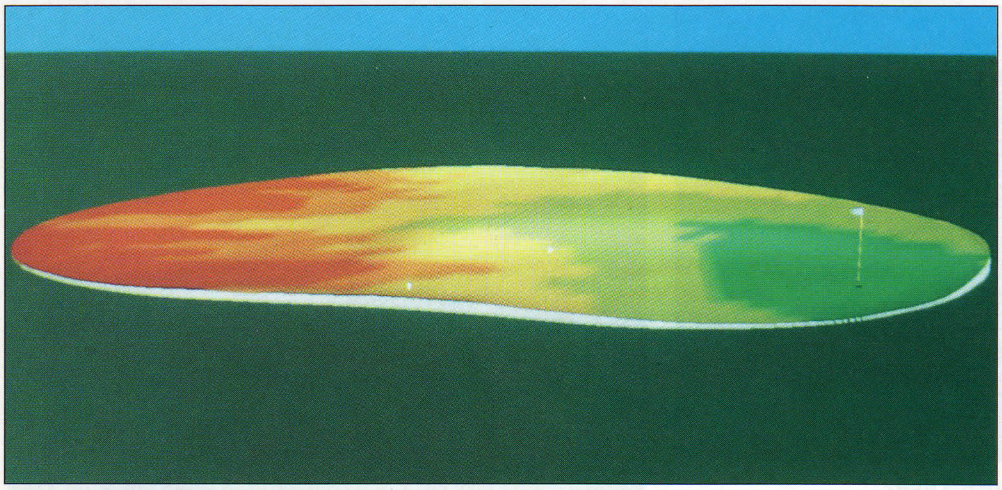


Figure 13. Putt difficulty map.

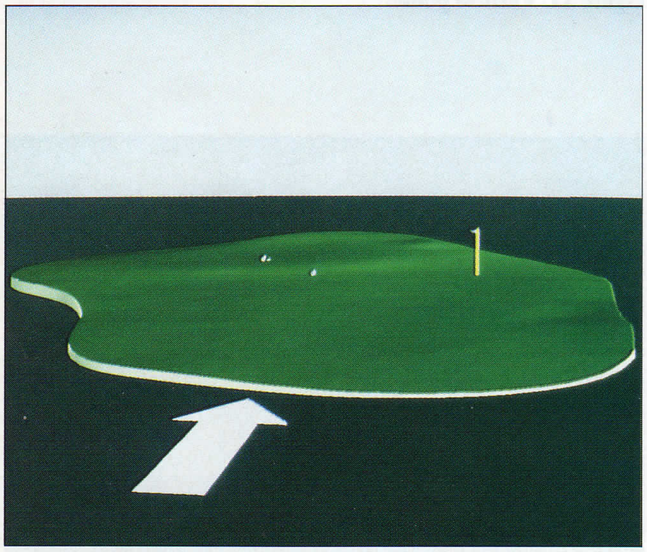


Figure 14. Walt Disney World 18th hole.

Walt Disney World Classic

On October 20, 1990, we attended the Walt Disney World Classic, a PGA tournament held in Orlando, Florida. We had two greens surveyed, the 15th and 18th, but during the tournament we used only the 18th green, shown in Figure 14. Each green required one day of survey time. During the tournament, a surveyor stationed on the 18th green camera tower used a theodolite to locate each player's ball. The surveyor provided horizontal and vertical angles for the balls. Using these angles, a ray-tracing algorithm cast a ray from the surveyor's location at the polygonal green and calculated *x, y,* and *z* coordinates for the ball. During the nearly three-minute interval between the last golfer's shot to the green and the first golfer's putt, an operator selected camera and light locations and started the putt simulations. All calculations were completed by the time the golfers reached the green. During the last round of the tournament, 59 putts were simulated—all before the golfers began putting.

Tim Simpson led by a large margin throughout the tournament, but John Mahaffey birdied eight holes in a row to pull within one stroke coming to the final hole. The producer informed us that because the match was so close, he was not comfortable about introducing our new technology. However, he invited us to try our system at a future tournament.

JAL Big Apple

Our next opportunity to use the golf visualization system was in July 1991 at the Wykagyl Country Club in New Rochelle, New York. This was a Ladies Professional Golf Association Tournament. Our experience in Orlando taught us to avoid the last hole of the tournament, so here we chose the par 3 16th hole, shown in Figure 15. This green is almost perfectly flat, but it does have a large slope from back to front. Also, this time we had the pleasure of showing our system to two NBC sports commentators, Charley Jones and Bob Trumpe. Both were excited about the system and suggested how we might improve it. For example, Trumpe asked us to color the higher elevations brown and the lower elevations green, to help the viewer understand how the green tilted.

During the two days of the telecast, the system performed superbly, and our simulations were shown live several times during the broadcast. We had our Silicon Graphics workstation in a trailer about 100 feet from the producer. During the broadcast, the producer told us when to show the simulation. Typically, we showed the putts from the golfer's point of view and occasionally from the caddy's viewpoint.

Implementation

We coded the system in LYMB,[10] a C-based object-oriented system that supports message passing and inheritance. LYMB applications are written using scripts that create instances of classes and change instances' states. The classes are written in C. LYMB has more than 400 classes that support a variety of applications: scientific visualization, industrial inspection, computer animation,[11] and molecular modeling.

Our golf application uses many existing classes such as ren-

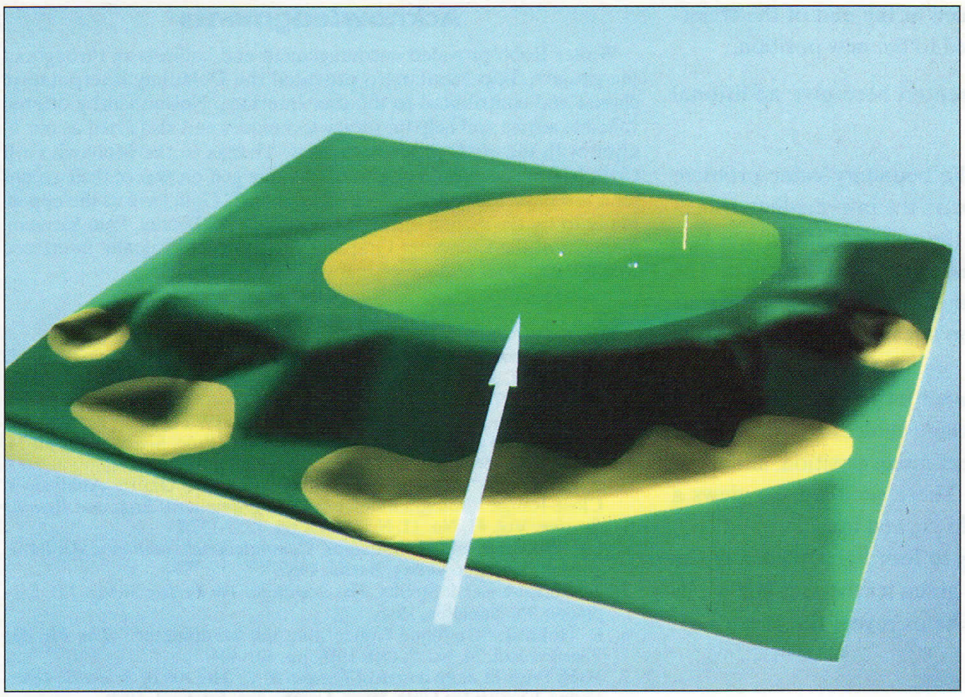


Figure 15. Wykagyl Country Club 16th hole.

derers, cameras, lights, actors, models, and splines. We also developed classes specifically for golf modeling and analysis. One set of analysis classes implements the initial- and boundary-value solution algorithms in *Numerical Recipes*.[7] User-interface classes for the HP widgets[12] permitted us to create a custom and portable X Window user interface.[13] The interpreted LYMB environment let us customize the user interface for easy operator interaction during the broadcast.

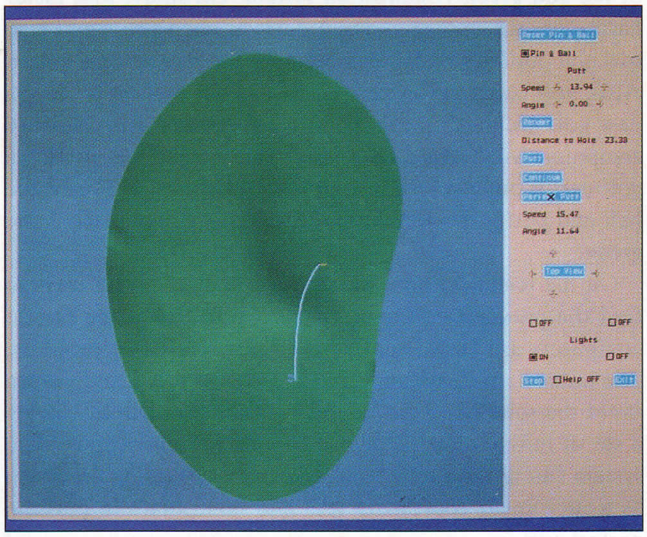
The Golf Green class reads survey data and creates a polygonal model of the green, perimeter, and skirt. It also responds to *@(x,y)z?,* returning the value of the elevation at the requested Cartesian location on the green surface. Another message, *@(x,y)normal?,* returns the surface normal at the point (x, y). The initial-value problem solver uses these messages. Golf Green has a scale factor that scales the elevations and normals so the shooting method can control the flatness of the green. We implemented the shooting method in a LYMB script that uses loops, the initial-value solver, and Golf Green classes.

The system runs on Sun 3/4, Hewlett-Packard 9000, Stardent GS2000, Digital Equipment DS5000, and Silicon Graphics 4D workstations. Rendering classes for vendor-specific hardware permit fast response on these systems. For instance, the Silicon Graphics workstation renders a typical green at 5 frames per second. Initial-value problems on this machine take less than a second. Boundary-value solution times depend on the green topography, coefficient of friction, and distance from the hole. For distances less than 10 feet, solution times are under 3 seconds. Distances longer than 30 feet require an average of about 20 seconds.

User interface

By integrating user-interface classes with existing applications, including the parser and animation classes, we created a powerful visualization environment. Figure 16 shows one user interface to the golf simulation. The green can be viewed from different points in space with a virtual camera controlled by a pointing device (a mouse). The user performs the rest of the interactions using various widgets in the control panel on the right:

- Pin & Ball pops up sliders to change the position of the pin and the ball on the green. Whenever the ball or pin position is changed, the system places the virtual camera in a natural player's position, so the ball and the hole are in the field of view. After the user changes the ball or hole position, three widgets immediately show suggested speed, direction, and distance from the ball to the hole.
- Arrow widgets change the initial putt speed and direction. Footprints and a putter head are displayed beside the ball each time the direction is changed. Zero direction corresponds to the straight line from the ball to the hole.
- The Putt selection shows the ball's trajectory on the green calculated on the basis of the defined initial speed and direction.



Speed 13.94
Angle 0.00
Distance to Hole 23.38
Speed 15.47
Angle 11.64

Figure 16. Golf simulation user interface.

• Continue shows the ball placed at the end of the trajectory. The virtual camera is adjusted to the new position.

Other pop-up panels not presented here give additional options:

• Perfect Putt shows solving the boundary-value problem using a shooting method and displays the intermediate trajectories. When the system reaches the solution, it displays two text widgets with the chosen initial speed and direction. The user can change the speed and the direction data for the perfect putt and analyze the solution's sensitivity to the initial conditions.

• Arrows in the viewing subpanel let the user select four predefined viewing positions. Possible choices are one step to the left, one step to the right, a bent-knees position, and a position looking down from above. These simulate the views a golfer gets by moving left, right, down, and up.

• On-off toggles corresponding to four lights placed in the corners of a rectangle around the green let the user change illumination. Appropriate lighting helps reveal the green's topography.

We modified the interface substantially before and during the two golf tournaments. The major changes were to accommodate fast data entry during the hectic broadcast times. Because LYMB is interpreted, no compilation changes were required.

Summary

Although we started our golf green visualization project in 1987, we did not realize its significance until 1990 when advances in hardware speeds and LYMB permitted its revival and success. Computer graphics has a vast repertoire of techniques that can be applied to sports. Our system blends mathematics, graphics, and computer science to enhance viewer entertainment. There are other potential applications of the golf green visualization system. Certainly, with faster hardware and innovative input devices, our work could lead to a golf putting simulation system for the recreational golfer. Also, golf course designers could use our system to design new golf greens and rehabilitate old greens. And because computers will become even faster and smaller, we envision a golf-cart-mounted system that uses telemetry to locate the ball and perform a simulation before the golfer reaches the green.

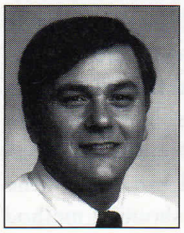
On the technical side, we are investigating better ways to find the parameters for the perfect putt. Even the fastest computers are challenged by the boundary-value problem. The initial-value problem is much easier to solve, and we can avoid the solution of the boundary-value problem by solving a set of initial-value problems before the tournament. The system can use the set of ball trajectories covering the green as a curvilinear coordinate grid and interpolate to find the perfect putt. ❑

Acknowledgments

Walter Robb provided encouragement and enthusiasm throughout the project. Skip Montanaro provided the Delaunay interpolation classes and contributed to the user interface. Nelson Corby offered valuable advice and help on survey techniques and also acted as our liaison with the NBC production crew. Thanks to the Mohawk Golf Club, Niskayuna, New York, for disrupting golf on two of their greens for a few days. Larry Cirillo, producer of NBC Golf, gave us the opportunity to try the system during a live broadcast. Finally, Pete Meenan, manager of the Computer Graphics and Systems Program, interfaced with the customer and solved many logistical problems.

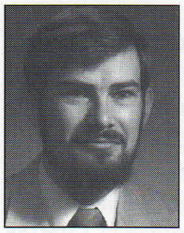
References

1. T. Saito and T. Takahashi, "Comprehensible Rendering of 3-D Shapes," *Computer Graphics* (Proc. Siggraph), Vol. 24, No. 4, Aug. 1990, pp. 197-206.
2. P. Hanrahan and P. Haeberli, "Direct WYSIWYG Painting and Texturing on 3D Shapes," *Computer Graphics* (Proc. Siggraph), Vol. 24, No. 4, Aug. 1990, pp. 215-223.
3. K.W. Wong, "Mathematical Formulation and Digital Analysis in Close-Range Photogrammetry," *Photogrammetric Eng. and Remote Sensing*, Vol. 41, No. 11, Nov. 1975, pp. 1355-1373.
4. F.P. Preparata and M.I. Shamos, *Computational Geometry: An Introduction*, Springer-Verlag, Berlin, 1985.
5. A. Cochran and J. Stobbs, *The Search for the Perfect Swing*, J.B. Lippincott, Philadelphia, 1968.
6. B. Holmes, "Dialogue Concerning the Stimpmeter," *The Physics Teacher*, Vol. 24, No. 7, Oct. 1986, pp. 401-404.
7. W.H. Press et al., *Numerical Recipes in C: The Art of Scientific Computing*, Cambridge Univ. Press, Cambridge, England, 1988.
8. A. Ralston and P. Rabinowitz, *A First Course in Numerical Analysis*, McGraw-Hill, New York, 1965.
9. J.D. Mackinlay, S.K. Card, and G.G. Robertson, "Rapid Controlled Movement Through a Virtual Workspace," *Computer Graphics* (Proc. Siggraph), Vol. 24, No. 4, Aug. 1990. pp. 171-176.
10. W.E. Lorensen et al., Object-Oriented Software Development in a Non Object-Oriented Environment," in *Object-Oriented Geometric Modeling and Rendering*, Siggraph Course Notes, P. Sabella, ed., ACM, New York, 1987.
11. W.E. Lorensen and B. Yamrom, "Object-Oriented Computer Animation," *Proc. IEEE NAECON*, Vol. 2, IEEE, New York, 1989, pp. 588-595.
12. *Programming with the HP X Widgets, Version 11*, Hewlett-Packard, Corvallis, Ore., 1988.
13. B. Yamrom and W. Lorensen, "X, Golf and Object-Oriented Programming," *TOOLS 90, Proc. Second Int'l Conf.: Technology of Object-Oriented Languages and Systems*, Angkor, Paris, 1990, pp. 443-454.



William Lorensen joined the GE Research and Development Center as a graphics engineer in 1978. He is currently working on algorithms for 3D medical graphics and scientific visualization. His other interests include computer animation, color graphics systems for data presentation, and object-oriented software.

Lorensen holds a BS in mathematics (1968) and an MS in computer science (1971) from Rensselaer Polytechnic Institute.



Boris Yamrom joined the GE Research and Development Center as a graphics scientist in 1984. His interests include applied and pure mathematics, object-oriented programming, computer graphics, computer animation, geometric modeling, and simulations.

Yamrom holds a BS (1969), an MS (1970), and a PhD (1973) in mathematics from Leningrad University and is a member of ACM (Siggraph, Sigplan).

The authors can be reached at GE Research and Development, 1 River Rd., Bldg. KW, Schenectady, NY 12345. Lorensen's e-mail address is lorensen@crd.ge.com, and Yamrom's e-mail address is yamrom@crd.ge.com.