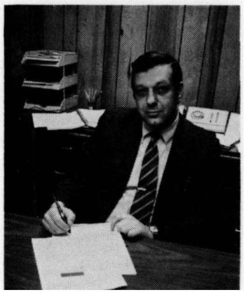

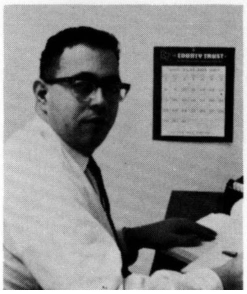

3-D Visual simulation

by

ROBERT A. GOLDSTEIN
Manager, Computer Animation
Mathematical Applications Group, Inc.
White Plains, New York

and

ROGER NAGEL
Faculty Research Assistant
University of Maryland and
Systems Analyst
Mathematical Applications Group, Inc.
White Plains, New York

ABSTRACT

This paper describes a visual simulation technique by which fully computer-generated perspective views of three-dimensional objects may be produced. The method is based on a relatively simple geometric modeling technique for the mathematical representation of the three elements essential to the picture-taking process, namely, a camera, a light source, and the object or objects to be photographed. Once these three basic components have been defined, geometric ray tracing is employed to compute a "picture" of the object as it appears in the simulated camera. In essence, individual light rays are traced from their source to the surface of the object. The reflected component of each ray is computed and traced to its point of intersection with the film plane. Thus, each reflected ray provides the intensity at a single point on the picture, and, when a sufficient number of points have been computed, the entire area of intensity data may be displayed on a cathode ray tube. Several examples of the pictorial output of this process are shown, and the application to the computer-generated films is discussed.

INTRODUCTION

Visual simulation is, as the name implies, an attempt to construct a mathematical model of the physical process of "seeing" a three-dimensional object. Although both visual simulation and the more conventional techniques of computer graphics have as their goal the production of pictorial output, there is a basic difference between the two approaches. Computer graphics systems display an object as a collection of line segments which serve to distinguish the boundaries between surfaces. This approach produces line drawings having a flat, two-dimensional quality which, although geometrically accurate, are lacking in realism.

The simulation approach treats an object as a set of three-dimensional surfaces that reflect light, and it is this reflected light impinging on photographic film (or the retina of the eye) that forms an image of the object. The result is, therefore, a fully toned picture, closely resembling a photograph of the real object. It is this added degree of realism that makes the simulation approach attractive for many applications.

The success of any visual simulation system rests largely on its ability to provide an accurate model of the basic ingredients involved in the photographic process. These are a source of light, a camera, and the object to be photographed. In addition, the phenomena of light transmission and reflection must be realistically simulated. The system described in this paper has been under development at MAGI for several years and provides a unique approach to the problem. As is the case with most complex simulation models, compromises between mathematical accuracy and computing efficiency have been introduced, but, on the whole, the visual outputs from this system are quite striking.

Probably the most important — and certainly the most difficult aspect of any visual simulation system — is the method by which three-dimensional geometries are represented. For this reason, the

3-D modeling technique, referred to as the Combinatorial Geometry Method, which forms the heart of this system, is discussed at some length.

GEOMETRIC MODELING

Combinatorial Geometry is a technique for representing, in a computer, a mathematical model of a three-dimensional object or a group of objects. One of the prime considerations in the development of this technique was to make it easy to use so that a three-dimensional model can be constructed using little or no mathematics.

With this in mind, a user-oriented building-block approach was selected as offering an excellent combination of simplicity and flexibility. This approach is based on the assumption that most objects, although quite complex in their entirety, can be subdivided into simpler, easy-to-define geometric figures. Starting with a set of these figures or building blocks, the model builder can construct an almost endless variety of complicated objects.

To illustrate the method, consider the simple examples shown in Figure 1. Two basic figures, a sphere and a cylinder, will be combined to form several more interesting shapes. To avoid confusion, the term *body* will be used to denote a basic figure, while a *region* is defined to be any combination of one or more bodies. In Figure 1a, the sphere, defined by its radius and the location of its center, has been labeled Body 1. The cylinder (Body 2) is defined by the coordinates at the center of either base, a radius, and the magnitude and orientation of the axis. Note that the two bodies have been positioned so as to overlap and occupy the same volume in space.

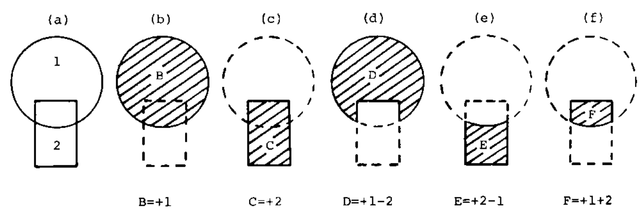

Figure 1 - Examples of the Combinatorial Geometry method

Figures 1b through 1f show the five possible regions which can be formed from these two bodies. Below each figure is an equation which defines how the region is to be constructed. A region equation defines a volume in space, such that all points which satisfy the equation lie within the volume, while points which do not satisfy the equation are external to the volume. The general form of a region equation is given by a combination of the boolean operators *and* (+) and *not* (-), *e.g.*,

$$R = \pm B_1, \pm B_2 \ldots\ldots\ldots\ldots\ldots \pm B_n$$

where R denotes a particular region and the B's represent body numbers. The + or - signs serve as boolean operators having the following significance: $A(+B)$ implies all points previously defined and those points in Body B, while $A(-B)$ implies all points previously defined and not in Body B. Thus

$R_1 = +A-B$ is the region consisting of all points in Body A and not in Body B. If effect, we are constructing a region by the systematic addition or subtraction of the volumes occupied by a set of bodies.

Returning to the examples, consider Region B in Figure 1b. It has been defined as +1 meaning that all points in B lie within Body 1. The result is a region of spherical shape. Note that no mention need be made of Body 2 in this equation since the cylinder is not needed to define Region B. Similarly, Region C, defined as +2, lies within Body 2, thereby forming a cylindrical region. Region D is a bit more interesting since it involves the process of body subtraction. The equation states that points within D must lie within the sphere (+1) but not within the cylinder (-2). Thus, we have constructed a sphere pierced by a cylindrical hole. Region E, on the other hand, lies inside the cylinder (+2), but outside the sphere (-1), giving a cylinder with one concave base. Finally, Region E is defined as all points which lie simultaneously within the sphere and the cylinder, giving a cylindrical region with one convex base.

Although the above examples dealt only with a sphere and a cylinder, the following nine body types are available for geometric modeling:

1. Sphere
2. Right circular cylinder
3. Right elliptic cylinder
4. Truncated circular cone
5. Truncated elliptic cone
6. Ellipsoid
7. Rectangular parallelepiped
8. Right-angle wedge
9. Arbitrary polyhedron (any convex figure bounded by up to 6 plane surfaces)

Each of these bodies is described by a set of parameters defining its dimensions and its location and orientation in space. The model builder may employ as many of each body type as necessary to create the desired geometry. The construction of a complete model is shown in Figure 2. This sequence of computer generated photographs illustrates how individual bodies are systematically "pieced together" to form a rather complex object. In all, 21 bodies and 24 regions were employed in the final model. For a more detailed discussion of geometric modeling see Reference 1.

CAMERA AND LIGHT SOURCE

As noted earlier, this technique is basically a simulation of the physical process of photographing an object. We have described how the subject to be photographed can be modeled, and all that remains is to provide a model of the camera and a source of light.

The camera is represented by a simple pinhole camera containing an imaging or film plane and a focal point (pinhole) through which all light rays must pass before reaching the plane. Since the distance between the pinhole and the image plane is arbitrary,

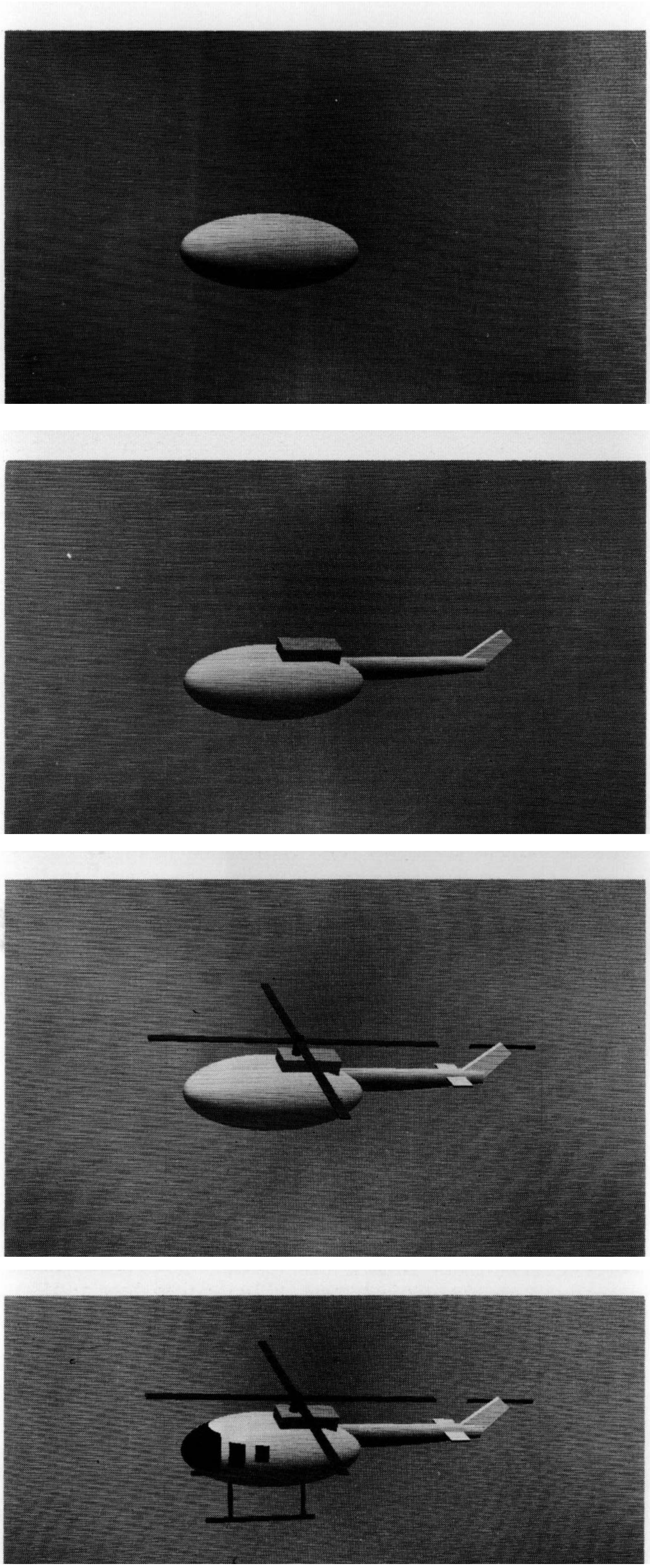

Figure 2- Four Steps in Modelling a Helicopter

the camera has, in effect, a variable focal length lens. That is, an increase in this distance will narrow the field of view (telephoto lens) while a decrease in distance enlarges the field of view (wide-angle lens). The image plane is subdivided into a rectangular matrix of points, and the ultimate goal is to define the intensity of light reaching each point on this matrix. The density of points in the matrix determines the resolution or image quality of the final picture.

The source of light may be a uniform plane parallel source, such as sunlight, defined by its direction and intensity. Alternatively, a point source can be used which decays with distance according to a R^{-n} law. The value of n, as well as the location and intensity of the source, are all variables in the model. The basic effect of the point source is to spread the light, causing surfaces which are distant from the source to receive less illumination than those which are close to the source. In addition to a primary light source, the model makes provision for a general or ambient source of light, which illuminates all surfaces equally, regardless of their orientation. This source, analogous to scattered sunlight, is included so that surfaces which are not directly illuminated by the primary source will not all appear black.

LIGHT TRANSMISSION

Before proceeding to a description of how the picture is actually computed, one final item must be defined, namely, the mechanism for reflecting light from the surfaces of the three-dimensional object. To simplify a basically complex phenomenon, the following approximation is made. The fraction of incident light reflected from a point on the surface of a region is a function only of (1) the angle θ between the normal to the surface at that point and the direction of the incident light and (2) the coefficient of reflection, k, assigned to the region. Thus, for a plane parallel source of light, the intensity of reflected light, I, is given by the equation:

$$I = I_0 k \cos\theta + A_0,$$

where I_0 and A_0 are the intensities of the primary and ambient light sources, respectively. Note that a negative $\cos\theta$ occurs when the surface does not face the light source and, in this case, I is set equal to A_0. The major drawback to this treatment is that it assumes all surfaces to be perfectly diffuse, when, in reality, most materials exhibit some specular (or mirror-like) component in the reflected beam. Thus, a certain amount of realism will be lost when dealing with metallic or other highly reflecting surfaces.

One further optical property that is worth mentioning is the ability to assign the property of transparency, rather than reflectivity, to any region in the geometry. Regions given this property will transmit all incident light and, therefore, not appear in the final picture. This provides the system with a kind of "x-ray vision" in that the camera can be made to look through the outer shell of an object and view its internal structure.

SIMULATION VIA GEOMETRIC RAY TRACING

We are now in a position to describe the procedure used to define the light intensity at every point on the image plane matrix (that is, how we simulate the action on the camera film plane). Basically, this is accomplished by geometric ray tracing, a technique which tracks individual rays of light from the light source to the object to the camera. In practice it is more efficient to perform the calculation in the reverse order, so that each point on the image plane serves as an origin for a ray. The ray is projected from its origin, through the camera's focal point, and out into space until it intersects one of the bodies which make up the geometry. The outward normal to the body at the point of intersection is then computed as is the reflectivity of the region which exists at that point.

With this information the intensity of the reflected ray can be calculated from the equation given above. Thus, ray tracing is basically a method of determining the first body and region to be encountered by a ray after it leaves the camera.

The procedure is actually quite simple and is illustrated by the example in Figure 3. A geometry consisting of three bodies and three regions is shown, along with the appropriate equations of the regions. The camera is located at P_0 in a transparent region C, which encloses the entire geometry. A ray is emitted from the camera and the problem is to determine what part of the geometry lies in the path of the ray. What is done, in essence, is to solve the equation of the ray simultaneously with the equations describing each of the bodies. Since the allowed body types are all convex, there will be two possible solutions for each body, corresponding to the point at which the ray "enters" the body and the point at which it "leaves" the body. There will, of course, be no solutions for those bodies which do not lie in the path of the ray.

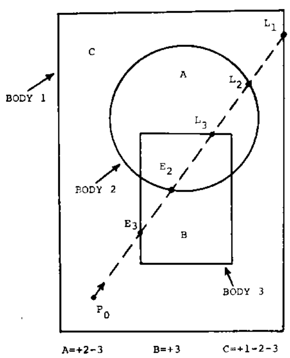

Figure 3 - Geometric Ray Tracing

In this example, however, there will be six possible solutions, and the required intersection point will be the smallest, non-negative solution. This is obviously the entering point of body 3 (E_3). Having determined what body has been hit, we must now define which region exists at point E_3. This is done by advancing the ray to E_3 and investigating each of the region equations until one is found that is satisfied by this point.

Consider Region A, which requires the current position of the ray to lie within Body 2 but outside of Body 3. Since the ray is known to be entering Body 3, the equation for A cannot be satisfied. A similar argument eliminates Region C. Equation B, however, is satisfied since it requires only that the ray lie within Body 3. Thus the ray has been found to intersect Body 3 and Region B at the point E_3.

Next we consider the possibility that Region B has been defined as transparent. In this case, the ray must be continued until it reaches a solid surface. The next point along the path is E_2, which places it within the sphere as well as the cylinder. Once again only Region B can be satisfied at this point and the ray must be advanced to the next intersection, L_3. The ray has now left Body 3, but, since it is still within Body 2, it must have entered Region A. We can carry the example one step further by assuming that Region A is also transparent. The ray would be advanced and ultimately reach the point L_1, which lies outside of the geometry. When this occurs, the corresponding point on the image matrix is known to receive no reflected light and is assigned a specified background intensity.

The above example illustrates how one ray from the camera is analyzed. When every point on the image plane has been defined by this method, the picture is ready to be displayed.

PICTURE OUTPUT

The end result of a calculation for a single picture is a raster of intensity levels on the image plane. This can either be recorded on magnetic tape for subsequent display or, with suitable interface hardware, be fed directly from the computer to a display system. Without going into detail on display hardware, the basic requirement is an intensity-modulated cathode ray tube (CRT) in which the electron beam voltage can be varied with position on the face of the tube as in television. The stored intensity levels in fact control the beam voltage so that each point is displayed at the required brightness, producing the continuous tones in Figure 2. Other examples of the output of this process are shown in Figure 4.

The photographs shown here were made at the Computer Science Center of the University of Maryland. The CRT display system, designed and built at the University, is capable of displaying an image containing up to 4096 x 4096 raster points, in 64 shades of grey. The display can be photographed with either a still camera or a 16mm movie camera operating in single-frame exposure mode. A detailed discussion of the Maryland equipment is available in Reference 2.

APPLICATION TO MOTION PICTURES

The area where visual simulation will find its greatest application is in the field of computer-generated motion pictures, or, as it is more commonly called, computer animation. Although the term *animation* has traditionally been synonymous with simple hand-drawn cartoons, we are concerned here with the more realistic, fully shaded pictures obtainable with visual simulation techniques.

Animated films of this type are rare, and the reasons are purely economic. It might take an artist an hour or more to draw one accurately shaded, perspective view of a complicated object. When one considers that even a short animated sequence could

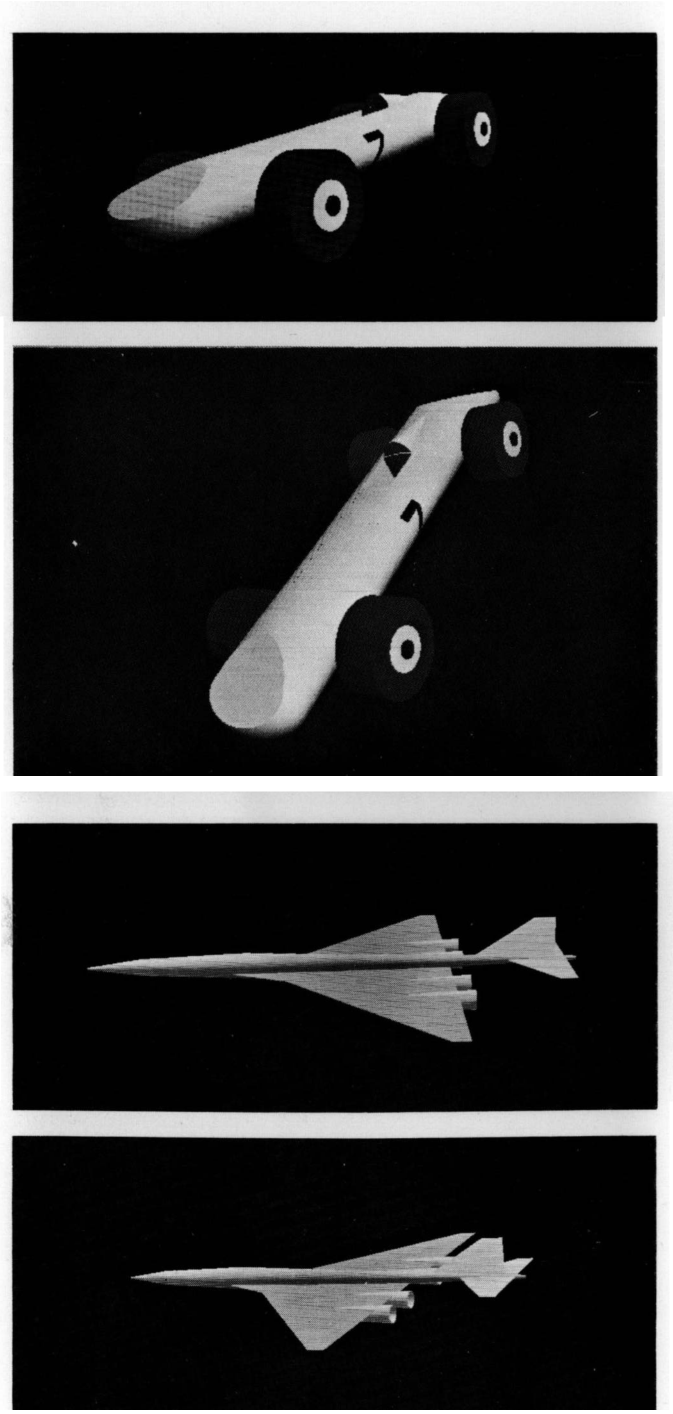

Figure 4 - Examples of Visual Simulation Output
Side and Top View of Racing Car
Two Views of Airplane

require hundreds of such drawings, each slightly different from the other, the task becomes monumental. The computer, on the other hand, can generate each new view in a matter of seconds.

DIRECTOR'S LANGUAGE

The previous discussion has been limited to geometric modeling techniques and how they may be applied to the generation of a single view. If this technique is to be at all useful for motion pictures, however, the work required to define the model initially must be eliminated for all subsequent views. For example, the parameters which describe the locations and orientations of each body used to model an object should not have to be respecified every time that object changes position. Similarly, there should be complete freedom to move and aim the camera as well as the illumination source. What is needed is a shorthand way of making these changes or, in effect, a "director's language."

A computer input language has been designed to serve just this purpose and has been incorporated into the system. The language is constructed so that the user can, with very little effort, produce the input for a sequence of views and then generate the entire sequence in a single computer run. Basically, it consists of a series of commands, the most powerful of which serve to relocate objects in space. Suppose that the object in the last photo of Figure 2 had to be moved by a distance 3, 0, and -1 units in the X, Y, Z directions, respectively. The name HELICOPTER could be supplied for that object and a command issued to

MOVE HELICOPTER (3,0,-1)

The data defining the location of every body associated with HELICOPTER would then be automatically incremented by the above amounts. If one also wanted to rotate the object around a pre-defined axis of rotation (labeled AXIS 1, for example) through a 15^{o} angle, the command

TURN HELICOPTER (AXIS 1,15)

would be used. The orientation of each body would then be adjusted to provide a 15^{o} rotation of the entire object. Similar commands exist to translate and rotate both the camera and the light source so that the "director" has complete control over the objects on his "set."

In order to test the flexibility and possible shortcomings of the language, a short (30 second) film was made using the University of Maryland's display hardware outfitted with a 16mm camera. The film showed the helicopter and a simple ground level gun emplacement. The helicopter was programmed to undergo a series of maneuvers including turns, takeoffs, and landings, etc., until it eventually is shot down and crashed. Using the Combinatorial Geometry Method, the initial geometric modeling, required to generate the first movie frame, was prepared in about 8 man-hours. Using the director's language, however, all subsequent frames in the film were defined in a total of only 4 man-hours.

EXPERIMENTAL RESULTS AND FUTURE DIRECTION

The evaluation of the visual simulation system described here would not be complete without an assessment of the computing time involved in generating a typical picture. This is largely a function of (a) the degree of resolution desired, as measured by the number of points on the image plane and (b) the complexity of the geometric model.

While experience to date is insufficient to provide any rule-of-thumb on how these affect computing time, we can at least indicate some representative results for work done in the past. The photographs of Figure 2 were computed for a 720 x 960 image plane matrix or a total of almost 700,00 points. The first picture, consisting only of an ellipsoid, required 5 seconds of computing on the CDC 6600. The complete helicopter shown in the last picture was generated in 20 seconds on the same computer.

Future development work will be aimed primarily at decreasing computer costs and increasing the capabilities of the system. The generation of color images will be of prime concern. Other areas of concentration will include the development of efficient techniques for combining computer-generated images with live photography and for computerized frame editing and manipulation.

COMPARISON TO OTHER SYSTEMS

The idea of visual simulation of three-dimensional objects is not new. The literature contains many references to 3-D models and simulations. Most of these techniques, however, apply to solving the hidden-line problem, but as yet little work has been reported on greyscale or color. The PhD thesis[3] of Philipe Loutrel at New York University is an example of this. The display of wireframe pictures or movies without greyscale toning and even without hidden lines lacks some of the realism available in greyscale output.

Art Appel has produced several papers in this area over a number of years, and he does have greyscale output. His work at IBM Research uses a two-list system to model 3-D objects. The first list contains all the vertices of all objects as specified in cartesian coordinates. The second list describes the combination of these vertices into volumes or "objects." He then employs "quantitative invisibility"[4] to eliminate hidden lines, and a further program to produce greyscale output.[5]

The MAGI technique builds volumes from basic shapes rather than vertices and this, one suspects, leads to less tedious input. MAGI actually simulates light rays. Since light does not penetrate solid objects, no hidden lines are computed. Greyscale is again a direct result of simulated light rays. It would seem, then, that the MAGI technique is a simulation of the end product (picture), as well as the production of the picture. Appel, however, simulates only the end product. One cannot claim either method is "better," unless a basis for choice is defined (*e.g.*, economics, computer resources needed, training of users, etc.) Since no basis has been as yet accepted, no claims of relative merit can be made.

The University of Utah projects in this area have produced some excellent visual simulations with at least two hidden-line algorithms, and have even generated color output.[6,7] There is, however, little basis for direct comparison, because Utah has used special-purpose hardware and striven for real-time production rather than strictly software simulation.

Similar comments can be made about the work reported on by Rodney Rougelot of G.E.[8] Here again special-purpose hardware systems were employed to produce real-time or quasi-real-time color displays.

Roger L. Boyell at PRA raises an important idea in his paper.[9] The basic point is that visual simulation is well suited to a TV output format. That is, the TV industry has had years to produce very powerful electronic gear useful in such important tasks as mixing, adding, color, recording, and playback, etc. One can then, if using a TV output format, make use of this technology. He described a system which takes advantage of the raster format of a TV picture. This system was developed as a next iteration of a radar simulation extended to full video simulation. The system mainly models terrain and uses polar coordinates in calculations as opposed to cartesian coordinates. In addition, it uses some special-purpose analog-digital hardware and is trying for real-time operation.

The technique of this paper could easily be made to work with TV equipment. The use of a raster for output makes this a natural, and in fact all the advantages Boyell points out would apply. In a TV environment one can even use stop-action filming so that only quasi-real-time is necessary. That is, the data can be recorded and then, by normal TV editing, copied and run at standard rates.

The MAGI modeling system was designed for ease of use by non-mathematicians. This forced our choice of a cartesian coordinate system. The modeling of a semi-infinite medium like terrain as opposed to an object of definite size is a major difference between the two techniques.

ACKNOWLEDGEMENT

The authors wish to acknowledge the contributions of Mr. Walter Guber of MAGI, who was primarily responsible for the original formulation of the Combinatorial Geometry method.

REFERENCES

1. Davis, R., Nagel R., Guber, W., *A Model-making and display technique for 3-D pictures, Proceedings 7th UAIDE Annual Meeting, pp. 47-72, 1968.*

2. Pilipchuk, A., *Mechanical scanner for off-line picture digitization*, Technical report 69-92, University of Maryland Computer Science Center, College Park, Maryland, April 1969.

3. Loutrel, P., *An algorithm for eliminating the hidden lines in computer-drawn polyhedra, Proceedings 7th UAIDE Annual Meeting,* pp. 421-440, 1968.

4. Appel, A., *The notion of quantitative invisibility and the machine rendering of solids, Proceedings ACM 22nd National Conference*, MDI publication, Wayne Pennsylvania, pp. 387-393, 1967.

5. Appel, A., *Some techniques for shading machine renderings of solids, AFIPS Proceedings*, v. 32, SJCC, 1968. MDI publication, Wayne, Pennsylvania, pp. 37-49, 1968.

6. Watkins, G. S., *A hardware compatible algorithm for generating visible surfaces from 3-dimensional data, Proceedings 8th UAIDE Annual Meeting*, pp. 389-401, 1969.

7. Wylie, C., Romney, G., Evans, D., Erdhal, A., *Half-tone perspective drawings by computer, AFIPS Proceedings*, v. 31, FJCC, 1967. MDI publications, Wayne, Pennsylvania, pp. 49-58, 1967.

8. Rougelot, R., *Private communication*, General Electric, Syracuse, New York.

9. Boyell, R., *Computer simulation of lunar displays*, Society of Photo-Optical Instrumentation Engineers 14th Annual Symposium, August 1969.

ROBERT GOLDSTEIN is a Systems Analyst at MAGI and directs the development work on the computer animation system described in this paper. Mr. Goldstein received his BS in Physics from Brooklyn College in 1955 and his MS from the University of Maryland in 1960. His interest in geometric modeling stems from earlier work on the analysis of complex nuclear reactor radiation shields. Mr. Goldstein is a member of the Society for Information Display.

ROGER NAGEL is a Faculty Research Assistant and Instructor at the University of Maryland. He received his BS in Mathematics and MS in Computer Science from Stevens Institute of Technology in 1964 and 1968, respectively. He is currently working towards a PhD in Computer Science at the University of Maryland. Mr. Nagel first became interested in computer animation as an employee of MAGI, where he worked on various 3-D graphics systems. He is currently working on a research project sponsored by MAGI at the University of Maryland to develop new techniques for computer animation. He is also serving as co-chairman of the UAIDE Computer Animation Committee. Mr. Nagel is a member of the AMA and ACM.