Applications of ray tracing in molecular graphics

Thomas C. Palmer

Cray Research Inc., North Carolina Supercomputing Center, Research Triangle Park, NC, USA

Frederick H. Hausheer

The Oncology Center, The Johns Hopkins University, Baltimore, MD, USA

Jeffrey D. Saxe

Cray Research, Inc., National Cancer Institute—Frederick Cancer Research Facility, Frederick, MD, USA

Ray tracing is an elegant and powerful technique for creating computer-generated images. The wide variety of geometric primitives and realistic effects, such as reflections, transparency, and shadows, make it one of the most popular rendering methods in use today. We present a brief introduction to ray tracing and discuss some of the computational issues involved. Examples illustrate ray tracing's effectiveness for producing high-quality visualizations of molecular structures.

Keywords: ray tracing, molecular modeling, raster graphics, visualization, constructive solid geometry

INTRODUCTION

Line-drawing devices have traditionally been the preferred medium for molecular graphics. The advantages of these systems are well known and widely utilized. Wire-frame and dot surface models enable one to focus on intramolecular topology and relative atomic positions. In addition, the ability to manipulate these models in real time is crucial in building an understanding of the three-dimensional (3D) structure of a molecular system.

Several trends are responsible for the increasing popularity of shaded image visualizations for displaying molecular data. Sophisticated raster graphics workstations developed over the last few years are capable of displaying both antialiased lines and shaded images. Shaded images provide the means to display complex molecular models based on primitives other than lines or dots. In the absence of animation, static shaded images are often superior for conveying 3D information for presentation or publication.

Many techniques have been developed to address the chief question of raster rendering: What color is assigned to each pixel? Classical methods answer by finding the object closest to the eye (usually by sorting) and computing the shade based on surface orientation and incident illumination from light sources. With this approach, objects are effectively projected onto the 2D viewing plane prior to shading. This *local illumination model* is incapable of capturing many significant aspects of surface-light interaction.

Ray tracing assigns colors to pixels by answering a more fundamental question: What surfaces interact with the photons that pass through a pixel and into the eye? Rather than following photons forward from light sources, we trace backward from the eye (through each pixel) along possible paths that a photon could have traveled. An individual arc in this backward tracing is called a *ray* and is represented by an origin point and a direction vector. A ray is fired from the eye point through each pixel in the image to be generated. These rays can be reflected from the closest surface or transmitted (with refraction) through the surface (see Figure 1). Each reflected or transmitted ray can, in turn, strike other surfaces, resulting in a *ray tree* with binary branching (Figure 2). In theory, a ray will terminate only when it either leaves the scene or is absorbed. In practice.

Address reprint requests to Mr. Palmer at Cray Research Inc., North Carolina Supercomputing Center, P.O. Box 12732, 3021 Cornwallis Rd., Research Triangle Park, NC 27709, USA.

Received 28 February 1989; accepted 14 March 1989

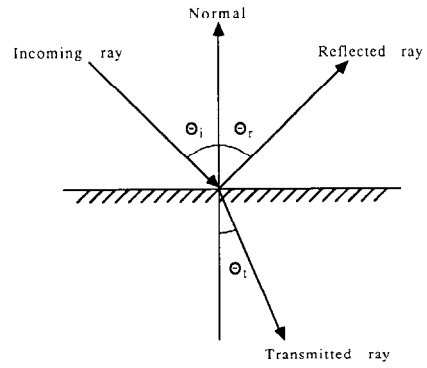

Figure 1. Geometry of reflection and refraction. The angle of incidence equals the angle of reflection: $\Theta_i = \Theta_r$. Snell's Law relates the angle of incidence with the angle of refraction: sin Θ_i/sin Θ_t = the index of refraction. (Borrowed with permission from Ref. 1.)

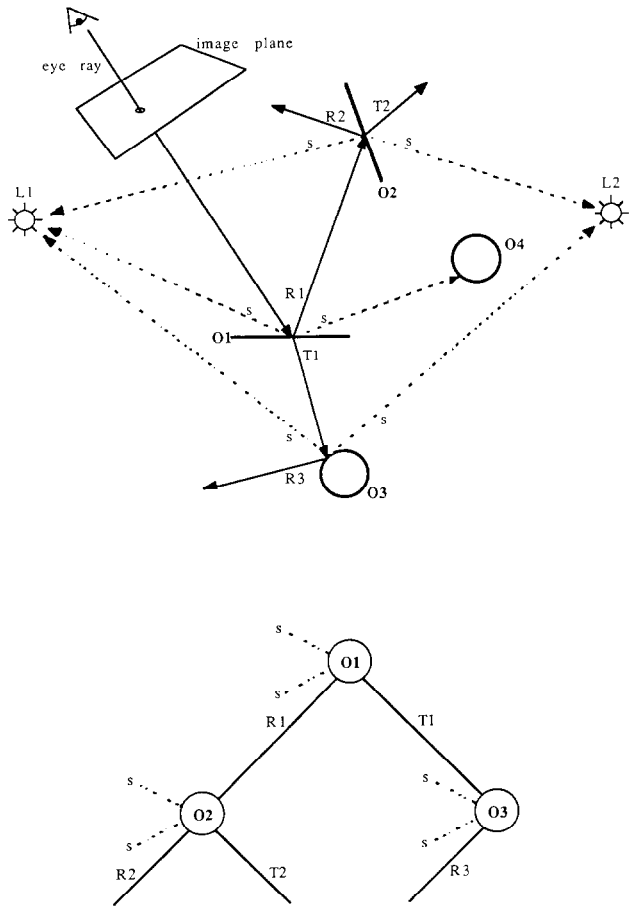

Figure 2. (Top) An eye ray propagated through a scene creating a binary ray tree. The dashed rays are shadow feelers sent from an intersection point toward each light source. If the shadow feeler hits any object closer than the light source, the intersection point is in shadow. (Bottom) Schematic ray tree. (Borrowed with permission from Ref. 2.)

the depth of the ray tree is limited by computer time and memory constraints.

Each node in the ray tree is assigned a color based on the properties of the intersected surface. This color is then added to the weighted colors of the reflected and transmitted ray tree branches evaluated recursively. Evaluating the ray tree to determine pixel color invokes a *global illumination model*. Surface-light interactions can be simulated, since visibility and shading are computed in 3D model space.

Ray tracing was first described by Kay.[3] However, Whitted's algorithm is more general and has been widely implemented.[4] Only the briefest of introductions has been given here; for additional information, refer to the excellent introduction provided by Rogers.[5] Hanrahan has compiled an exhaustive survey of ray-primitive intersection algorithms.[6]

STRENGTHS AND LIMITATIONS

Realistic visual effects that are difficult or impossible to achieve with other rendering techniques are handled naturally by the ray-tracing algorithm. The basic algorithm provides reflections, transparency with refraction, shadows, intensity depth cuing, texturing and multiple local light sources.[4] With various extensions, ray tracing can simulate motion blur, soft shadows, depth of field,[7] diffuse interreflections and caustics.*[8]

Another advantage of ray tracing is the availability of a wide variety of geometric primitives. The most fundamental operation of ray tracing is the calculation of ray-primitive intersections. This calculation must yield the distance to the intersection point, the surface normal at that point and the reflected and refracted rays. The range of objects for which an intersection algorithm can be derived defines the set of available primitives. Algorithms exist for quadrics (spheres, cylinders, cones, ellipsoids, hyperboloids and paraboloids), superquadrics, polygons, fractals, parametric surfaces, surfaces of revolution, height fields and volume densities. Ray tracing is also the most efficient technique for rendering objects created with *constructive solid geometry* (CSG) methods. CSG objects are built by combining primitives using binary set operations such as union, difference, or intersection. Figure 3 shows some basic primitives.

Ray-primitive intersection algorithms vary considerably in their complexity and run-time efficiency. Relatively efficient analytic solutions exist for many primitives (e.g., quadrics). For other primitives (e.g., parametric surfaces), numerical methods must be used.

The advantages of ray tracing are not obtained without substantial computational costs. Image-generation time is measured in minutes or hours for medium to complex models rendered at reasonable resolution. Two factors are primarily responsible:

(1) The number of rays can be extremely large. Over one million are required in a 1024 × 1024 image for the eye rays alone.

*A bright area on a surface caused by focused light is a caustic. The image of the sun cast by a magnifying glass is an example

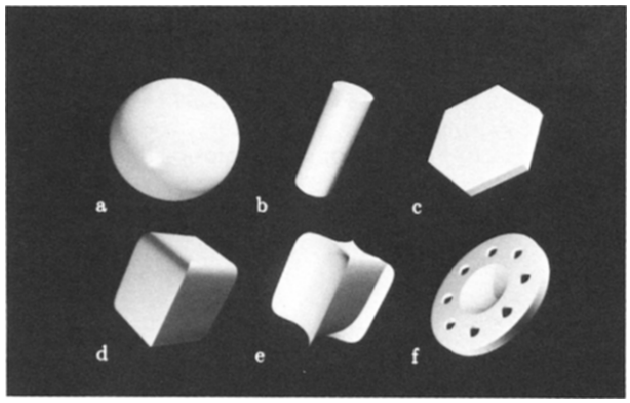

Figure 3. Some basic ray-tracing primitives: (a) sphere, (b) cylinder, (c) set of polygons, (d) convex superquadric, (e) concave superquadric, (f) constructive solid geometry composite created by subtracting eight cylinders and a sphere from parts of a cylinder

(2) Many ray-primitive intersection calculations are required to find the nearest object. In effect, each ray sorts a portion of the object database to find this object.

Much of the research in ray tracing over the last decade has been invested in accelerating the process. Most strategies work by attacking the two factors mentioned above. The number of spawned rays can be reduced in at least two ways. One approach is to adaptively prune the ray tree when the contribution of a ray falls below a certain threshold.[9] The other method involves using more than one eye ray per pixel to antialias the image. If each ray tree yields approximately the same color, then the assumption is made that all rays hit the same set of objects. If this condition is not met, more rays are fired in the area of conflict. This technique is called *adaptive supersampling* and tends to fire rays where they will do the most good.[4,10]

A widely used method for reducing ray-primitive intersections is to surround complex objects with a *bounding volume*, such as a sphere. If a ray misses the bounding sphere, it will also miss the enclosed object, and the more expensive ray-primitive intersection calculation is obviated. A *hierarchical bounding tree* can be constructed in which volumes contain either other volumes or primitives. If a ray misses a volume in the bounding tree, the entire set of volumes and primitives enclosed by the subtree can be bypassed safely.[4,11,12]

APPLICATIONS TO MOLECULAR GRAPHICS

With the enhancements developed over the last decade, ray tracers are capable of producing images that approach photorealism. But of what use is realism in an application where objects of interest can be two orders of magnitude smaller than a wavelength of visible light?† In general, realism (good cues for depth, size and shape) will lead to more rapid and complete comprehension of the underlying geometric model. If the model accurately depicts a familiar physical or intellectual construction, then realism will enhance its visualization.

There is, however, a limit to the level of desirable realism in scientific visualization. Care must be taken not to produce overly complex images. Excessive use of reflection and transmission can produce images that are visually confusing and difficult to interpret. There is certainly room for research into how different levels of realism affect the speed and depth of comprehension.

Color Plates 1–7 present several examples of the application of ray-tracing methods and modeling to molecular graphics. Table 1 provides additional information on each image.

Color Plate 1 is a schematic representation of an interstrand crosslink complex of DNA and phosphoramide mustard. This time-averaged structure is the result of a 50-picosecond molecular dynamics simulation with counterions and water. The drug and the backbone of the DNA are modeled with spheres and cylinders. Sugar and base rings are represented with extruded polygons. The image permits rapid, qualitative determination of local denaturation in DNA and the conformation of the crosslink complex. The visualization provides strong structural cues from occlusion and intensity depth cueing while maintaining some of the see-through quality of line drawings.

Spheres can be used to model structures other than atoms. Color Plate 2 is a stereo pair of an energy-refined *Cro* repressor-DNA complex. The *Cro* repressor is a sequence-specific DNA binding protein involved in regulation of gene expression.[13,14] A cubic spline curve was computed using only the alpha carbon atoms. Spheres were placed along the curve to yield the smooth tube-like

†In daylight, the eye is most sensitive to light at wavelengths of about 5500 Å

Table 1. Image-rendering summary. For stereo pairs, the time given is that required to render one view. All images were rendered at four samples (eye rays) per pixel. *RayT* is a locally developed ray tracer with enhancements for rendering molecular data. *Oasis* is a graphics package from Cray Research

Image	Primitives	Resolution	Time(hh:mm:ss)	Hardware	Software
Color Plate 1	547	768 × 768	00:21:20	Cray X/MP 24	RayT
Color Plate 2	4100	768 × 768	00:51:16	Iris 4D/70GT	RayT
Color Plate 3	1	1024 × 768	00:05:00	Cray X/MP 48	Oasis
Color Plate 4	58	768 × 768	03:56:43	Iris 4D/70GT	RayT
Color Plate 5	1032	768 × 768	00:23:21	Iris 4D/70GT	RayT
Color Plate 6	2649	768 × 768	22:09:01	Iris 4D/70GT	RayT
Color Plate 7	111	1280 × 1024	00:24:37	Cray X/MP 24	Oasis

appearance. This visualization emphasizes the secondary structure and the spatial and conformational relationships of the various parts of the complex. The DNA is modeled using only the phosphorous atoms, thereby emphasizing the conformational changes in the DNA backbone induced by protein binding.

Visualizations of volumetric data (scalar functions of x, y and z) have become extremely popular over the last few years. Molecular volumetric data can result from X-ray crystallography or electron microscopy. Data can also be produced by electron density or electrostatic potential simulations. A volume visualization of electron density appears in Color Plate 3. The system on the left is an iso-electron density surface of a tetrathiafulvalene (TTF) trimer.[15] TTF is an organic metallic conductor. Electron density in the frontier orbital of a TTF trimer is displayed on the right. The electron density was simulated by *ab initio* quantum mechanical calculations using DMOL.

Color Plate 4 displays a transparent, enhanced-radius CPK surface surrounding a ball-and-stick model. The molecule is the nucleotide guanine. The electrostatic potential is color-coded on the CPK surface. The shading function used to produce a color at each node of the ray tree does not have to be based on a color that is constant across the surface of a primitive object. In this instance, a function returning an electrostatic potential given x, y and z values is used to select a color. Normally, transparent spheres interior to other transparent spheres would be visible. A software modification was made to the ray-sphere intersection routine that checked for this condition and eliminated those intersections. The resulting visual effect is a transparent shell surrounding the molecule.

Another modification was made to the ray tracer to support a dot primitive. A dot is implemented as a sphere with two special properties. First, the dot's radius varies directly with its distance from the eye point. Thus, all dots appear the same size regardless of position. Second, a dot's surface normal is a constant specified in the input along with position. Therefore, a set of dots representing a surface can give additional perceptual cues (via shading) to the shape of that surface. Color Plate 5 replaces the CPK surface of the previous model with a solvent-accessible dot surface computed with Connolly's MS program.[16] The dots are color-coded by electrostatic potential.

Color Plate 6 combines many of the shading and modeling techniques previously discussed. The source for the data is the molecule *perforin*, which is liberated from cytotoxic T-lymphocytes and associates in circles to punch holes in the membrane of a target cell.[17] The image shows the portion of one such protein that is in contact with the cell membrane. The protein backbone is modeled with spheres on a cubic spline curve, and the surface is represented with transparent spheres with color indicating electrostatic potential. Only those residues contributing to the charge on the visible side are modeled. Note that the strongly positive portion on the upper right will fit into the upper left negative region of the next protein in the circle.

Constructive solid geometry techniques are especially well suited to building schematic models. Color Plate 7 shows a schematic diagram of *perforin* in action. Each instance of the protein is modeled with two cylinders (for each α helix) with a connecting loop. The red "cell" is a sphere with another, smaller sphere subtracted from its interior, thus leaving a "membrane" with a thickness of 30 Å. The hole in the membrane was created by subtracting a cylinder from the cell sphere. The radius of the hole is about 50 Å. Eighteen instances of *perforin* line the lesion.

CONCLUSIONS

Ray tracing is an effective and flexible technique for producing high-quality visualizations of molecular structures. The variety of available geometric primitives proves useful in modeling various aspects of molecular systems. The realistic visual effects contribute significantly to rapid comprehension of the underlying geometric model. High-quality shaded images also provide an excellent complement to traditional vector and dot displays.

The computational expense inherent in ray tracing is a significant problem. However, hardware is being developed that exploits the inherent parallel nature of ray tracing. In the near future, it will be possible to produce ray-traced images of large models in real time.

ACKNOWLEDGEMENTS

We thank Erich Wimmer and Gray Lorig of Cray Research for the volume-rendered electron density image. The image was created using Cray's Oasis graphics package.

Dr. Y. Takeda of the National Cancer Institute's Laboratory of Mathematical Biology provided the *Cro* repressor data. Dr. Manuel C. Peitsch (also of the Laboratory of Mathematical Biology) provided the *perforin* data and collaborated on the visualizations.

We also thank our colleagues at the Advanced Scientific Computing Laboratory for reviews and encouragement.

This project has been funded, at least in part, with federal funds from the Department of Health and Human Services under contract number NO1-CO-74102. The content of this publication does not necessarily reflect the views or policies of the Department of Health and Human Services, nor does mention of trade names, commercial products, or organizations imply endorsement by the U.S. Government.

REFERENCES

1 Glassner, A.S. Surface physics for ray tracing. *Introduction to Ray Tracing* (SIGGRAPH '88 course notes), August 1988

2 Glassner, A.S. An overview of ray tracing. *Introduction to Ray Tracing* (SIGGRAPH '88 course notes), August 1988

3 Kay, D.S. *Transparency, Refraction, and Ray Trac-*

ing for Computer Synthesized Images. Master's thesis, Cornell University, 1979

4 Whitted, T. An improved illumination model for shaded display. *Comm. ACM* 1980, **23**(6), 343–349

5 Rogers, D.F. *Procedural Elements For Computer Graphics*. McGraw-Hill, New York, 1985

6 Hanrahan, P. A survey of ray-surface intersection algorithms. *Introduction to Ray Tracing* (SIGGRAPH '88 course notes), August 1988

7 Cook, R.L., Porter, T. and Carpenter, L. Distributed ray tracing. *Comp. Graph.* 1984, **18**(3), 137–145

8 Kajiya, J.T. The rendering equation. *Comp. Graph.* 1986, **20**(4), 143–150

9 Hall, R.A. and Greenberg, D.P. A testbed for realistic image synthesis. *IEEE Computer Graphics and Applications* 1983, **3**(10), 10–20

10 Catmull, E. *A Subdivision Algorithm for Computer Display of Curved Surfaces*. Technical Report UTEC-CSc-74-188, University of Utah, Dept. of Computer Science, Salt Lake City, December 1985

11 Clark, J.H. Hierarchical geometic models for visible surface algorithms. *Comm. ACM* 1976, **19**(10), 547–554

12 Weghorst, H., Hooper, G. and Greenberg, D.P. Improved computational methods for ray tracing. *ACM Trans. Graph.* 1984, **3**(1), 52–69

13 Takeda, Y. Specific repression of *in vitro* transcription by the *Cro* repressor of bacteriophage λ. *Journal of Molecular Biology* 1979, **127**, 177–189

14 Ohlendorf, D.H., Anderson, W.F., Fisher, R.G., Takeda, Y. and Matthews, B.W. The molecular basis of DNA-protein recognition inferred from the structure of *Cro* repressor. *Nature* 1982, **298**, 718–723

15 Wimmer, E. and Mertz, J. Chemistry by simulation. *Cray Channels* 1988, **10**(2), 8–11

16 Connolly, M.L. *Protein Surfaces and Interiors*. Ph.D. thesis, University of California, 1981

17 Tschopp, J., Masson, D. and Stanley, K.K. Structural/functional similarity between proteins involved in complement- and cytotoxic- T-lymphocyte mediated cytolosis. *Nature* 1986, **322**, 831–834