Those of us who can remember punch cards and paper tape readers may find capabilities like Graphical User Interfaces, abundant memory, and desktop computing to be a dream come true. Fourteen years ago, I asked a friend why raytrace codes were not much used in lighting design, and he responded that you had to tell the raytrace code the exact order in which each ray struck each surface. Why not have the computer calculate this automatically? The common wisdom of the time was that the effects of partial reflection at transmissive surfaces created a $2^n$ complexity problem, which was unsuitable to a computational solution. That challenge was the birth of unconstrained non-sequential raytracing and OPTICAD®. This article summarizes the progress I have observed since the early 1980s.

There was, however, relevant work in progress toward non-sequential raytracing occurring in another field entirely. In the field of computer graphics, Turner Whitted had reported a method for synthetically generating...
ating imagery using raytracing. In this method a ray was traced backwards from the detector, through the simulated scene, and ultimately to a light source. This is possible because Fresnel equations work the same way following a ray backwards and forwards.

The trick, however, was not in calculating the light paths at refractive or reflective surfaces, but rather in managing the formation of secondary (ghost) rays at surfaces. Imagine a system of two flat mirrors facing each other, with a glass plate between them, and a ray starting between them, normal to the faces. The ray would bounce back and forth. Each time it passes through the glass, ghost rays would be generated. First one ray creating two, which create four, and so on, generating ray segments, until the computer (which has to keep track of each segment) ran out of memory. So, sophisticated adaptive techniques were developed that allowed the user to manage the formation of ray segments by controlling parameters such as the maximum number of ray segments, maximum number of ghost rays, and minimum transmittance. We implemented this technology as the OPTICAD® program, and for the last decade it has found an ever increasing variety of applications.

In the intervening years, non-sequential raytracing has been used to describe a variety of systems. However, our use of the term refers to programs that automatically trace rays through a design space, based strictly on the physical size, optical characteristics, and location of the component objects in space. The user does not have to coax the program into non-sequential raytracing by setting up “regions” with input and output apertures. A true non-sequential raytrace program must be able to trace a 4π steradian bundle of rays at once, as shown in Figure 1.

For the illumination engineer, non-sequential raytracing technology has been a tremendous time and money saver. It permits analysis of complex parts like automobile dash panel lightpipes—predicting a part’s lighting performance before the part is even built. Prior to the advent of these technologies, lightpipe designers (and other illumination engineers) found themselves designing a part, having to build and test the part in the lab, and then improving the design. This was slow and inefficient—but non-sequential raytracing changed that by allowing the designer to create a part in a CAD program, then import it for analysis to see how well light would travel through it. Where did the light escape? Where did it concentrate? Where is the best location for a bulb? The design was thus refined before the first lightpipe was built. This tool proved to be a quantum leap in lighting system design.

Monte Carlo raytracing

So far we have talked about specular reflection and transmittance, but much of lighting design is concerned with diffuse light. Diffuse light occurs when a light ray strikes a surface that does not produce one reflected and one transmitted ray segment, but rather produces a statistical distribution of possible ray paths. For example, a light ray striking sand-blasted glass may reflect and transmit in any number of directions. So how can we use raytracing to model a statistical distribution of light rays?

An effective solution is Monte Carlo simulation. Taken from the statistical technique of the same name, Monte Carlo raytracing uses a statistically significant number of rays, and analyzes system performance by random surface normal realizations at each diffuse surface. Surfaces can be modified to permit Lambertian, Gaussian, and x-y power law statistical surface normal distributions at each diffuse component. The process may seem computationally intensive, but modern desktop computers easily perform the task. Thus non-sequential raytracing was adapted to meet the challenge of diffuse reflection and transmittance.
**The interface revolution**

"All evolution in thought and conduct must at first appear as heresy and misconduct," said George Bernard Shaw, and so it was in optical software interface design. Many of the major raytracing codes in the early 1980s were written in FORTRAN and were the legacy of government development contracts, university research, and specific optical design projects. They were (and are) powerful, but each had its own unique interface, which was typically a command or script language that the user typed in. Plots could be sent to plotters. Output tables were sent to the terminal or line printer. Most software was expensive, leased rather than sold, and a short course was really required for a user to learn how to use the software. But that was about to change.

In 1989, when a young University of Arizona graduate named Ken Moore developed ZEMAX™, he designed it from the ground up with the goals of being extremely easy to use, reliable, and affordable. Moore believed that software, even optical design software, should be easy to learn—like the commercial spreadsheets and word processors available on personal computers. He created a user interface that combined graphical output with a handy spreadsheet-like data input. He made the program truly interactive so that a user could make a design change and immediately see the result.

Moore continued to improve the interface and usability of ZEMAX™, and it seems his insights about ease-of-use have paid off. ZEMAX™ raised the bar several notches for optical engineering software, and user’s expectations for both usability and the quality of user interface were going to be higher, henceforth, because they had seen what was possible. Illumination software would have to follow suit, and we developed a Windows™ version of OPTICAD® that aimed at superior user interface.

**The challenges**

With the advent of non-sequential raytracing, the illumination engineer was now equipped with a tool which could be used to address three major challenges:

- Predicting design lighting efficiency performance,
- Predicting design lighting uniformity performance, and
- Scattering prediction and stray light control.

Simple lighting efficiency is a basic challenge for many illumination engineers. Consider the engineers who design lighting for undersea search equipment, or engineers designing radiative heaters for semiconductor drying. They must be concerned with efficient delivery of light to a particular region of space. Non-sequential raytrace tools were ideally suited to this because they could follow a ray striking a reflector, once or perhaps bouncing several times, and compute the transmittance losses due to absorption at the surfaces and in the media (volume absorption) along the path.

Lighting uniformity is a major goal for the illumination engineer. Figure 2 shows two designs for an illuminated sign, which uses a lightpipe. What the designer needs to know, in addition to how bright the sign will be, is how evenly or uniformly illuminated it will be. Anyone who has tried to read a digital watch in the dark can relate. The left half of the watch display is extremely bright and the right half is too dim to read. Modern tools provide a mechanism to quantify this problem and solve it in the design stage. The upper sign in Figure 2 is not evenly illuminated—too much of the light is concentrated in the center, and people are likely to have difficulty reading the sign (as evidenced by the variation in pseudocolors representing intensity levels). Armed with awareness of this problem, the illumination engineer concludes that the light should be diffused, and elects to
frost the back of the plastic lightpipe. The result is shown in the bottom sign, which is much more evenly illuminated. The software can show the result either as naturally polychromatically shaded as black and white, or use pseudo color to enhance subtle changes across the field.

Too much light in the wrong place can be as undesirable as having too little. As a result, stray light analysis is an important function of lighting software.

Telescope and celestial instruments are often designed with baffles to control the movement of stray light, as ghost rays can completely eclipse the signal the instrument is intended to collect. These ghost rays can form as Fresnel reflections from refractive surfaces, or simple reflections from interior surfaces of the instrument. Laser systems are also good candidates for stray light analysis. Figure 3 shows a case, analyzed by William Swantner, in which a ghost focus occurred just above a CO₂ laser. This focus still possessed enough energy to cause harm to a person standing in the wrong place.

**Applications**

Maximizing the efficiency of a lamp reflector is often desirable. Figure 4 shows a desk lamp with rays traced. Note that some of the rays strike the surface multiple times and others do not. A polar plot from the desk lamp (See Fig. 5) is extremely useful because it shows how light is distributed in angle space, and in this case the design produces strong lobes just off the Z-axis, which might be desirable for its application. As is often the case, it is desirable to ensure that the design places light in a particular angular distribution.

The automobile industry is highly competitive, and cost-consciousness is part of every engineering decision. If a part, like a headlamp, can be made even a few cents less expensively while still meeting design performance, the manufacturer becomes more competitive in the marketplace by improving the design.

Here's how the process works. First, for a given application, figures of merit are established—in this case the engineer determines from specifications exactly how much energy should be deposited on the surface in front of the car and in angle space. The basic design for the part (in this case a bulb/reflecter assembly) arrives, usually in an auto company specific CAD format. The designer converts it to an exportable form—typically IGES. This design is then imported into the illumination software for analysis. A performance profile (probably consisting of the polar plot and intensity map) is built for the system, which becomes the baseline. The engineer then modifies the design, either to reduce cost, increase performance, or both, and re-evaluates the design. This iteration occurs until design goals are met. The modified design is made into a prototype, tested, and then sent to production.

**The market**

The illumination software market is an emerging market, because easy-to-use software tools have become available only in the last few years. Unlike other areas such as lens design, there are relatively few specialists, but many general engineers who find themselves analyzing and improving illumination system performance. My experience with OPTICAD® has introduced appli-
The future of lighting design software

The future of lighting design software is, forgive the pun, very bright indeed. Users expect to see an ever increasing level of integration between lighting software and other applications. Already, better software products are offering a high degree of document export and import capability. Technologies like Object Linking and Embedding (OLE) offer an approach for a higher degree of integration between software applications, which is desirable. OLE is a means of allowing programs, and the data within programs, to interact in a useful way. A simple example is the ability to place a spreadsheet in a word processor’s document, and yet retain the calcula-
tional capabilities of the spreadsheet program within the new compound document. There are, however, other approaches to integration. The technology is still maturing, but holds promise.

A subtle, but important advantage, for Windows™ users, is that a good Windows™ interface follows the Microsoft guidelines for Windows™ Interface—things like what happens when the File Open menu item is selected, and what functions are assigned to the left and right mouse buttons. By adhering to the Windows™ interface, programs are much more user friendly. Every day more lighting software becomes available under Windows™. While it is true that some of these programs are simply using a bolt-on-front-end to their command driven program, others are truly integrated Windows™ products, supporting all standard Windows™ functionality. The easy way to judge the quality of software interface is to “fly it before you buy it,” by obtaining a demo copy of the software either from Internet homepages or requesting by mail. The most useful demos are actually working models of the software that permit users to test features of the code, albeit in a restricted way, so you can see what it is like to actually use the code.

For 3-D model interchange, IGES has been a de facto standard for the past few years. However, it is a redundant standard, and most vendors do not support all of the hundreds of IGES entities. STEP will eventually replace this standard, depending on how rapidly mechanical CAD manufacturers accept and implement it. Other proprietary standards offer their own advantages, but broad acceptance would require industry-wide input and accessibility. The Non-Uniform Rational Bicubic Spline (NURBS) is seen as a good hope for a single, unified entity for general 3-D surfaces within other standards like IGES, however, support is still spotty.

This article has presented a discussion of lighting design software, based on the author’s experience. However, there are other good software tools offered by many companies including Lambda Research, Breault Research Organization, Optical Research Associates, and others. All in all, look for more features, better integration, and improved price-performance, as market competition works to the benefit of the end user. Finally, you should expect to be able to buy a digital watch that you can actually read in the dark!

References

Michael Abernathy is co-founder of OPTICAD Corp., and principal author of the OPTICAD® optical analysis program.