Tracing Rays

After retiring from teaching at Georgia Tech, I returned to do one last course on ray tracing in anticipation of work on a book on optical design. In spring 2007, I circulated a notice about the course that I called Analysis of Optical Systems. It provided a short definition of ray tracing, the manipulation, interpretation, and use of optimization of optical systems, and gave a short description of the advancing technology of simulating optical systems. I was surprised by the response of the students. They thought I was going to teach them how to use powerful computers to generate realistic scenes like those that animation studios like Pixar use to create movies like *Finding Nemo* or *Wall•E*. While I realized that ray tracing techniques were used to make computer-generated images (CGI), it never occurred to me to think of ray tracing that way.

I became aware of ray tracing as a separate discipline when I instituted an optics track in the School of Physics at Tech in the early 1970s. After creating a course on lasers, it became clear that an applied optics approach was needed, which included an advanced course in geometrical optics. It might have been called optical engineering, but the College of Engineering might have gotten a bit testy about such a course being taught in the School of Physics, so I called it Optical Design. There was a problem. I had never done any optical design or engineering beyond my thesis work in Raman spectroscopy.

So I signed up for a summer course, Fundamentals of Lens Design, at the Institute of Optics in Rochester. The course was taught by the chief lens designer at Kodak, Rudolph Kingslake [Editorial, "Rudolf Kingslake (1903–2003)," May 2003]. It was clear once the class began that he enjoyed his work in a fascinating field and shared it with anyone who was interested. I took the summer course just after Texas Instruments and Hewlett-Packard began their battle of programming calculators. Kingslake had one of the new calculators and could not contain himself as he bounced across the front of the auditorium narrating with great glee results of a set of ray traces through a series of lenses using one of these new marvels. After all, for those who had punched their cards and brought their offerings to the high priests of the IBM computers of that era, the invention of the programmable calculator was like the removal of shackles.

While the handheld calculator could be purchased for initial ray traces, it was difficult to do a thorough analysis with it. When the minicomputers became widely available, many of the programs that had been available only on mainframes were converted to run on these smaller machines. For a teacher in a technological institute, the transition that mattered was when the handheld calculators morphed into programmable desktop calculators with a miniature magnetic tape reader that served to store programs and a paper tape printer for output. But an HP 9815 calculator was costly, so we only had a single machine for the class. To run the ray traces for assignments, students would sign up for hour-long sessions to use the machine. The day before assignments were due, the students worked around the clock, sleeping under the tables of the optics lab at night.

When the desktop calculator was replaced by the personal computer, students could run their own copies of ray trace programs. Around that time, Mike Harrigan, a researcher at Kodak Labs, and I wrote a series of columns with the same title as this editorial for SPIE's *Optical Engineering Reports*. They were intended as a series of "five finger exercises" for those beginning in optical design. Sadly, we were not able to sustain the columns. But I hope to incorporate such exercises into a new text on optical design.

Now, mainframes are not needed or wanted for lens design. A personal computer can perform the analyses and optimizations using powerful processors. Even the computational demands of CGI, that "other" ray tracing, do not require mainframes. Instead, a large of number of standard personal computers running in massively parallel modes produce an exquisitely detailed CGI as one of a half a million frames of an animated movie.

I wonder about the initial enthrallment of those Tech students for instruction on CGI. All of the optics in reflection, refraction, and scattering of light from surfaces is built into the program. The mastery comes from the knowledge of how to employ the rendering machines to create the most compelling effects. These are graphic design programs. This is not to belittle graphic design, for it is the most intriguing and satisfying discipline that I know of outside of optical engineering.

The main purpose for generating calculated images with a computer is the realization of a motion picture or to create special effects. While the content changes from movie to movie and the challenges for rendering become evermore complex, the output is a computer-generated image sequence. In contrast, goals of a ray trace of an optical design change with each new lens, system, or environment. The disciple requires a designer to have a broad range of knowledge about optics, materials, and environments and to modify his or her approach as the task demands.

At the end of the course I introduced the students to image simulation, an analysis technique that uses a digital picture as the object to be imaged instead of the usual line or grid of separated object points. This type of ray tracing begins to bump up against CGI. This analysis produced a CGI-like rendering of a single frame that put heavy demands on the processing power of their machines. It also gave my physics students a taste of the ray tracing that they thought they were going to study.

Donald C. O'Shea
Editor