

Ray Tracing vs Scan Line Rendering



These two graphics techniques are quite different. Both have been in use for several decades now but have been used in different ways. Here we'll cover some of those differences as they appear in HolliDance.

Ray Tracing

[Ray Tracing](#) uses an optical approach to rendering. Rays are mathematically created for each [pixel](#) in the final image. A ray starts in the camera and enters the scene through one of the pixels. This is the opposite direction that light would take, but eliminates having to calculate all the rays that don't reach the camera.

After a ray is created, its path is compared to the space that all the scene's objects occupy. Optimization eliminates the need to do an explicit intercept calculation for each [polygon](#). If intercepts are found, the intercept for the shortest ray length is shaded based on the polygon's color and the scene's lights.

If the object is transparent or reflective, new rays can easily be generated from that point. Shadows are also easy to support by making rays that test intercepts between the original intercept point and the light sources.

Ray tracing sounds pretty easy, but it is computationally very expensive. In 1987, my first ray traced image took several days to render. A little over a decade later, I rendered this more complex scene in *only* 30 minutes!



Ray traced scene

Scan Line

In order to render graphics in realtime, a different approach has been used. In [Scan Line rendering](#), each 3D polygon is transformed to 2D screen coordinates. It's corners are shaded based on the polygon's color and the scene's lights and the polygon is scanned into the frame buffer. Each rendered pixel's distance from the camera (Z value) is compared to a [Z-buffer](#) that has the same dimensions as the window. If the new pixel is closer to the camera, that pixel is written to the frame buffer and the Z buffer is updated.

This type of rendering is fast, but does not support photo realistic features like shadows or reflections. Those effects can only be simulated with fairly complex code.



Scanline rendered scene

GPU Enhanced Scan Line

A few years ago, graphics card (GPU) manufacturers made their cards [programmable](#). More sophisticated effects like [per pixel lighting](#) and [full 3D shadowing](#) became easier to support yet the results always lacked the quality of ray tracing. Ray Tracing was a luxury that took too long to calculate in real time ... until recently.



GPU enhanced scan line rendered scene

Real Time Ray Tracing

Now graphics cards are so powerful, that all the additional calculations required for ray tracing can be performed in a fraction of a second. HolliDance employs a simpler ray tracing technique that results in hard shadows and ignores [reflected light contributions](#). As graphics cards get even faster, I'll add these more [sophisticated](#) rendering options to the program for even more photo realistic results.



Real time ray traced scene

It's a great time to be a 3D graphics artist!

Contact

I can be reached via at my web page at <http://www.cool3dworld.org>. You can email me at dwhite6011@charter.net. Please reference HolliDance in the email subject line.