Will Yager (WY2527)

Accurately Rendering a Black Hole

Report

Goals

- Render a black hole, a night sky, and an aesthetically pleasing accretion disk
- Accurately factor in the effects of gravity into this rendering

Description of Technique

Path Tracing

This program uses full path tracing to generate an image. That is, the camera shoots out virtual "photons" that interact with the world in reverse order. The photons stop moving once they have passed through so much material that the opacity exceeds 99%. That is, if a photon collides with the black hole or travels into a cloud of very dense gas, it will stop tracing.

Gravitational Lensing

In order to accurately simulate a black hole, the path of the photon has to follow the distorted spacetime around the black hole. Path tracing allows us to numerically compute the path of the photon as it moves in the black hole's gravity well. The computation of the photon's path is rather involved, but the gist of it is that we can pretend the photon is a massive particle moving through a particular $\frac{1}{r^4}$ potential, and this will accurately reproduce the path of a real photon.

Texture Generation

The texture of the accretion disk and the stars are both generated using 3-dimensional Perlin noise. This leads to natural-looking distributions of stars and gas in the accretion disk.

Celestial Sphere

The stars are simulated by placing a textured sphere very very far away from the black hole, so photons are essentially no longer affected by the gravity of the black hole.

Accretion Disk

The accretion disk is composed of a spiral of luminous gas (with decreasing temperature farther from the black hole) with a non-zero thickness (i.e. it's not just a texture).

Technical Details and Engineering

Path Tracing

The program uses leapfrog integration with a dynamically adjusted path segment length so as to dynamically move between speed and accuracy where appropriate. The path length is calculated to be $l_0 * \min(s_{bh}, s_{ad})$, where l_0 is a pre-defined default length, $s_{bh} = \frac{1}{1+e^{10(1.4-r)}}$ is the black hole proximity accuracy factor, $s_{ad} = \frac{1}{1+e^{10(0.4-\frac{y}{20w_{ad}})}}$ is the accretion disk proximity accuracy factor, and w_{ad} is the accretion disk width. Additionally, at large distances from the black hole, where gravitational effects are negligible, the path distance is increased to 10 Schwarzschild radii.

Gravitational Lensing

Every photon is endowed with a pseudo-velocity vector corresponding to the pseudo-acceleration it's accrued as it travels through the black hole's gravitational potential. Initial pseudo-velocity is set to 1 (in Schwarzschild radii per arbitrary time unit) in the direction of photon travel. Photons are accelerated according to the pseudo-force field $-1.5 * h^2 \frac{\hat{r}}{r^5}$ where $h = x_0 \times v_0$ is a constant of motion for each photon. This psuedo-force operates on dimensionless pseudo-mass 1 (thanks to choice of coordinate system where the Schwarzschild radius is 1). This derivation is credited to Riccardo Antonelli.

Texture Generation

I wrote an algorithm to endow any point in 2D or 3D space with a Perlin noise-generated value at arbitrary frequencies and fourier profiles. Implementation details are available in `XY.hs`, `XYZ.hs`, and `PerlinG.hs`. The clever part was embedding the entire space in a series of grids of dimension n with the edge length being a very large integer (2^{64} in this case), with each additional level of cell containing 2^n smaller cells. This obviates the need for any sort of blending across cells, and allows constant-memory lookup for any point in space. Pseudorandom values are generated using the `SipHash` algorithm with arbitrary seed values.