Programming Club

Ray Tracing

Hugh Leather

13^{th} March 2015

1 Problem description

Build a simple ray tracer in any language you like.

You are allowed to look at the API docs for your language, but nothing else except what's in this document without asking for permission.

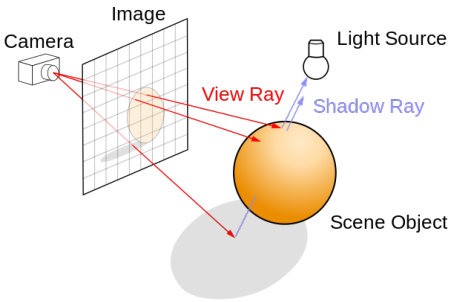
Time 1 hour. You may make teams.

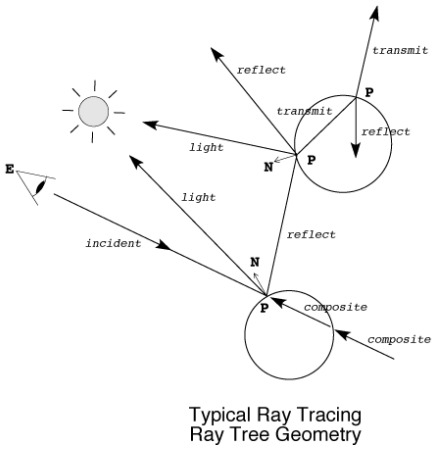
Winners for:

- Prettiest pictures
- Quickest to get it working
- Smallest code base

2 Introduction

Reverse ray tracing





Typical Ray Tracing
Ray Tree Geometry

3 Some maths

3.1 Dot product

$$\begin{aligned}\mathbf{a}\cdot\mathbf{b} &= (a_x, a_y, a_z)\cdot(b_x, b_y, b_z)\\ &= a_x b_x + a_y b_y + a_z b_z\\ &= \|\mathbf{a}\|\|\mathbf{b}\| cos\theta\end{aligned}$$

3.2 Length of vector

$\|\mathbf{x}\| = \sqrt{\mathbf{x}\cdot\mathbf{x}}$

3.3 Reflection

Incident vector, $\mathbf{x}$, surface normal, $\mathbf{n}$.

Reflected vector is $\mathbf{r} = \mathbf{x} - 2\mathbf{n}(\mathbf{x}\cdot\mathbf{n})$

3.4 Refraction

Incident vector, $\mathbf{x}$, surface normal, $\mathbf{n}$. Refractive index of original medium, γ_1, of new medium, γ_2.

Let

$$\gamma = \gamma_1/\gamma_2$$
$$c_1 = \mathbf{x} \cdot \mathbf{n}$$
$$c_2 = \sqrt{1 - \gamma^2 * (1 - c_1^2)}$$

Refracted vector is $\mathbf{r} = (\gamma * \mathbf{x}) + (\gamma * c_1 - c_2) * \mathbf{n}$

3.5 Intersection of ray and sphere

Sphere is centred at $\mathbf{p_c}$, radius r. Ray origin at $\mathbf{p_0}$, direction $\mathbf{d}$.

Let:

$$a = \mathbf{d} \cdot \mathbf{d}$$
$$b = 2\mathbf{d} \cdot (\mathbf{p_0} - \mathbf{p_c})$$
$$c = (\mathbf{p_0} - \mathbf{p_c}) \cdot (\mathbf{p_0} - \mathbf{p_c}) - r^2$$

Intersection at $t = \dfrac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

3.6 Intersection of ray and plane

Plane has normal $\mathbf{n}$ and c (gives implict eqn $\mathbf{n} \cdot \mathbf{p} + c = 0$)

Ray origin at $\mathbf{p_0}$, direction $\mathbf{d}$.

Intersection at $t = -\dfrac{c + \mathbf{n} \cdot \mathbf{p_0}}{\mathbf{n} \cdot \mathbf{d}}$

3.7 Lighting and shading

Energy from point source is inversely proportional to distance squared. Tends to be very dark, consider lower exponent than 2.

Energy per unit area proportional to $cos\theta$, where θ is angle between surface normal and light source.

Note, this simple model will be a bit rubbish. Consider adding ambient light

Shadows if object intersects ray from object to light.

Reflection and refraction can spawn a new ray each - have a depth threshold.

Note simple checkerboard patterns are easy, based on where the point is in space.

4 Output

The simplest picture format you might use is PPM. You can, however use any way to show the ray traced image you like.

The PPM format is:

- A "magic number" for identifying the file type. A ppm image's magic number is the two characters "P6".
- Whitespace (blanks, TABs, CRs, LFs).
- A width, formatted as ASCII characters in decimal.
- Whitespace.
- A height, again in ASCII decimal.
- Whitespace.
- The maximum colour value (Maxval), again in ASCII decimal. Must be less than 65536 and more than zero.
- A single whitespace character (usually a newline).
- A raster of Height rows, in order from top to bottom. Each row consists of Width pixels, in order from left to right. Each pixel is a triplet of red, green, and blue samples, in that order. Each sample is represented in pure binary by either 1 or 2 bytes. If the Maxval is less than 256, it is 1 byte. Otherwise, it is 2 bytes. The most significant byte is first.
- A row of an image is horizontal. A column is vertical. The pixels in the image are square and contiguous.