Assignment 3: Path tracing

EDAN30

April 2, 2011

In this assignment you will be asked to extend your ray tracer to support path tracing. In order to pass the assignment you need to complete all tasks. Make sure you can explain your solutions in detail.

Verify your result by comparing your images to those in the seminar.

1 Scene setup

For this assignment you should use the Cornell scene. Download *cornellscene.zip* from the course web page. Unzip the files and add them to your project. Use *buildCornellScene* to build the scene and *setupCornellCamera* to position the camera. This scene is a variation of the classic Cornell box. It is good for debugging global illumination algorithms due to its visual properties and simple geometry.

Task 1: *Start by implementing a whitted ray tracer performing jittered supersampling. Make sure your tracer handles direct illumination.* You should know this from previous assignments. If a ray does not hit anything, the color returned should be black, i.e., the background color should be black.

Task 2: *Extend you direct illumination model to attenuate each point light with increasing distance. To this end, calculate the distance, d, between a light and a receiving point and multiply its contribution by* $\frac{1}{d^2}$.

2 Path tracing

Path tracing is one of the simplest and most intuitive algorithms for global illumination. This part of the assignment requires you to implement the algorithm, step by step.

2.1 Indirect illumination

In real life there is no single light source illuminating the environment. In reality, light bounces off surfaces making each surface a new light source in turn. In order to better approximate the rendering equation, we need to consider this indirect illumination.

Indirect illumination can be written as:

$$L_{indirect}(x \to \Theta) = \int_{\Omega} L_{in}(x \leftarrow \Psi) f_r(x, \Psi \leftrightarrow \Theta) cos(N_x, \Psi) \, \mathrm{d}\omega_{\Psi} \qquad (1)$$

where L_{in} is the incident radiance at x in direction Ψ. This can be approximated using a finite sum of discrete samples:

$$L_{indirect}(x \to \Theta) \approx \frac{1}{n} \sum_{i=1}^{n} \frac{L_{in}(x \leftarrow \Psi_i) f_r(x, \Psi_i \leftrightarrow \Theta) cos(N_x, \Psi_i)}{p(\Psi_i)} \qquad (2)$$

where Ψ_i are samples with probability distribution function (pdf) $p(\Psi)$.

The incoming radiance in direction Ψ_i is the same as the outgoing radiance from the hit point when tracing a ray in that direction. Using the ray casting function, $r(x, \Psi)$, this can be written as:

$$L_{in}(x \leftarrow \Psi) = L(r(x, \Psi) \to -\Psi) \qquad (3)$$

Intuitively, to approximate equation (1) we sample the entire hemisphere using rays and treat the illumination at each hit point as a light source. This is a recursive integral, requiring the illumination at each sample point to be recursively evaluated.

We could do this by sampling illumination from the environment using more rays at each recursive hit. This is not tractable as we need hundreds of rays

at each recursion for good results, each new ray contributing less to the image. In path tracing we stick to a single ray and randomize its path. Given enough rays through each pixel, the result will converge to the correct solution.

Task 3: *Implement multiple bounces of indirect lighting by tracing recursively in a random direction on the hemisphere.* Use russian roulette to terminate the recursion. In order to get good results you need a large number of rays per pixel. Increase your supersampling to at least 100 rays per pixel.

In this first implementation you should use uniform samples, i.e., $p(\Psi) = k$, where k is constant. Start by calculating the value of k using the identity:

$$\int_{\Omega} p(\Psi)\,\mathrm{d}\omega_{\Psi} = 1 \Leftrightarrow \int_{\phi=0}^{2\pi}\int_{\theta=0}^{\pi/2} p(\theta,\phi)sin(\theta)\,\mathrm{d}\theta\mathrm{d}\phi = 1 \tag{4}$$

The sample directions, Ψ_i, should be uniformly sampled using rejection sampling.

Proceed by sampling the surroundings using equation (2) recursively. If a ray does not hit anything, the color of the background (black) is used as illumination.

2.2 Cosine weighted sampling

In the previous task, the BRDF was multiplied with the geometric term $cos(\theta)$. This causes some rays with lower geometric term to contribute less to the resulting image, which reduces the efficiency of the rays. In order to remedy this, we send rays in a cosine weighted direction, allowing each ray to contribute equally.

Task 4: *Implement cosine weighted importance sampling for better efficiency.* Start by setting $p(\theta,\phi) = k \cdot cos(\theta)$ and use equation (4) to calculate the constant k. The geometric term can then be eliminated from equation (2) when sampling.

In order for this to work, the sample directions, Ψ_i, need to be sampled

using that pdf. Start by calculating the cumulative distribution function:

$$F(\theta, \phi) = \int_0^{\phi}\int_0^{\theta} p(\theta, \phi) sin(\theta)\, \mathrm{d}\theta \mathrm{d}\phi = \frac{\phi}{2\pi}(1 - cos^2(\theta)) \tag{5}$$

Separate into:

$$F_\theta = 1 - cos^2(\theta)$$
$$F_\phi = \frac{\phi}{2\pi}$$

Set each separated part to a uniform random number $u_\theta, u_\phi \in [0, 1]$ and solve for θ and ϕ:

$$\theta = cos^{-1}\sqrt{1 - u_\theta}$$
$$\phi = 2\pi u_\phi$$

In order to create a direction vector, Ψ_i, pointing in the direction of θ and ϕ, we need a basis around the sample point x with the normal, N_x, as z-coordinate axis. Once such a basis is established, the direction in that basis is given by:

$$x = cos(\phi)sin(\theta)$$
$$y = sin(\phi)sin(\theta)$$
$$z = cos(\theta)$$

2.3 Reflection and refraction

Reflection and refraction can be implemented using russian roulette. This is preferred over continuing two rays from the surface; one reflection ray and one refraction ray. It is much more effective to trace a single ray with full contribution to the image, than tracing two less contributing rays. The reflectivity and transparency of the material are used as probabilities for their respective events. If none of these events occur, direct and indirect illumination is performed instead.

Task 5: *Add a 50% reflective material to ball1 and a 50% refractive material with* $n = 1.5$ *to ball2. Implement reflection and refraction using russian roulette.*

2.4 Area lights

Path tracing cannot capture caustics caused by direct illumination from the point light (why?).

Task 6: *Remove the point light from the scene and add a area light.* This can be accomplished by adding a sphere with a emissive material. Use the same location as the point light and set the sphere radius to 20. Set the emissive color of the sphere to $(r, g, b) = (10, 10, 10)$.

Observe the caustics from the glass sphere.

Task 7: *Reduce the sphere radius to* 1 *and set the color to* $(r, g, b) = (4000, 4000, 4000)$. *Explain what happens.*

2.5 Image based lighting

If we have an open scene, many rays end up sampling the background. If a color is assigned to the background, it can be used as light source.

Task 8: *Remove the area light, the roof and the sides from the scene. Set the background color to* $(r, g, b) = (1, 1, 1)$.

To get more interesting illumination, a light probe can be used instead of a constant background color.

Task 9: *Make the ground 50% reflective. Load a light probe using the provided class. Sample it in the direction of the ray to get the background color.* Different light probes can be downloaded from the course web page.