CS52: Homework 2 Solutions

Out: Oct 8. Due: Oct 22.

Problem 1: Raytracing

(a) We can split the problem into various parts. First, we determine the number of viewing rays required. Since each pixel requires s^2 samples and there are wh pixels in the image, the total number of viewing rays is $n_v = whs^2$. For each viewing ray that hits on object, a shadow ray is needed for each light; since the fraction of viewing rays hitting an object is p, the total number of shadow rays is $n_s = pln_v$.

So, in the absence of reflections, we have

1. number of rays: $n_r = whs^2(1 + pl)$
2. number of object intersection tests: $n_i = n_r o = owhs^2(1 + pl)$.

The basic raytracing algorithm is thus linear in the number of objects, lights and pixels. While the number of lights often remains small, the number of object in complex scene is quite large, thus requiring acceleration structures to avoid this behavior. Later in the class we will define new data structures such that $n_i \approx n_r log(o)$.

(b) To compute the number of required intersection tests, we simply plug the given values in the equation given above.

1. Simple scene (low quality): $143,360$
2. Simple scene (high quality): $917,504,000$
3. Complex scene (high quality): $11,534,336,000,000$

Problem 2: Shading

(a) As given in class, the general Phong shading for one light can be written as

$$C = C_d + C_s = k_d \max(0, \mathbf{N} \cdot \mathbf{L}) + k_s \max(0, \mathbf{R} \cdot \mathbf{V})^n$$

For the scene in this exercise, the angle between the normal and the light direction is the same as the angle between the normal and the viewing direction (light and camera in the same position). This implies that $\mathbf{N} \cdot \mathbf{L} = \cos(\theta)$. Furthermore, since the angle formed between the mirror direction and the normal is the same as the angle between the normal and the light direction, $\mathbf{R} \cdot \mathbf{V} = \cos(2\theta)$. This gives us

$$C(\theta) = k_d \cos(\theta) + k_s \cos^n(2\theta)$$

(b) To solve this, we will solve the equation given above for θ, after plugging the material values. Note that in each of the two cases, $C(0) = 1$, so we can just work on $C(\theta)$.

1. $\theta = \arccos(C(\theta)) = \arccos(0.5) \approx 60°$
2. $\theta = \arccos(\sqrt[n]{C(\theta)})/2 = \arccos(\sqrt[n]{0.5})/2 \approx 3.37°$

Based on this results, it is clear that using the flash on a shiny surface would produce a very bright but small highlight in the center of the picture. But the rest of the surface will be dark making the flash not an effective solution.

Problem 3: Lighting

For a point to be in shadow, a shadow ray cast from a point in the direction of the light should hit an object at a distance shorter than the one from the point to the light source. Since the light and the camera are in the same position, we are guarantee that there cannot be any other object between the surface point and the light.

Problem 4: Transforms

(a)

1. $A \rightarrow B$:

$$M = \begin{bmatrix} 0 & 1 & 3 \\ -1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

2. $A \rightarrow D$:

$$M = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 4 \\ 0 & 0 & 1 \end{bmatrix}$$

3. $A \rightarrow F$:

$$M = \begin{bmatrix} 0 & -1 & -3 \\ 1 & 0 & 5 \\ 0 & 0 & 1 \end{bmatrix}$$

(b)

1 and 2. In these cases, B and D are directly linked to A, so we can use the previous values.

3. In this case, A is first transformed to D, then D to F. This last transform is $D \to F$:

$$M = \begin{bmatrix} 0 & -1 & -3 \\ 1 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

Extra credit: Raytracing Computational Complexity

The easiest way to determine the number of rays shot is to consider that for just one reflective bounce, a fraction pr of rays will have to be shot to check for further intersection; we can then repeat the same operations for checking shadows and further reflection recursively. We can thus write

$$n_r = n_v(1+pl)\sum_{i=0}^{\infty}(pr)^i = n_v(1+pl)\frac{1}{1-pr}$$