CS52: Homework 1 Solutions

Out: Sep 26. Due: Oct 5.

Problem 1: Image Storage

(a) To compute the number of the maximum images fitting in memory, we simply solve the equation

$$\begin{aligned} numberOfImages &= \frac{mainMemory}{imageResolution \times colorDepth} \\ &= \frac{768 \text{ MB}}{16Mpixels \times colorDepth} \\ &= \frac{48 \text{ B}}{colorDepth\ in\ bytes} \end{aligned}$$

This gives us the following number of images:

1. 4 for single precision, floating point HDR images (96bpp = 12 Bpp)
2. 8 for high-precision color images (48bpp = 6 Bpp)
3. 16 for standard color images (24bpp = 3 Bpp)
4. 16 for standard color images encoded after gamma correction

(b) 48bpp and 96bpp images will contain all the information found in the original photograph.

(c) When dropping image resolution, the resulting images will be less sharp and blocky artifacts will pop in when zooming into the image. When changing color depth from 96bpp to 48bpp, image values higher than 1 are clamped and the less gradation can be represented in the image. When further reducing the color depth to 24bpp, even less color gradations can be represented. This becomes especially noticeable when further operations are applied to the image, e.g. scaling. If 24bpp images are encoded after gamma correction though, they will look better, since they still maintain relatively large color gradation.

(d) Let us define x the value of a channel of a pixel in the image and x' its the value after conversion.

1. `w = user-defined-value; x’ = 255*pow(min(1,max(x/w,0)),1/gamma);`

```
2. w = pow(2,16); x' = 255*pow(min(1,max(x/w,0)),1/gamma);
3. w = pow(2,8); x' = 255*pow(min(1,max(x/w,0)),1/gamma);
4. x' = x;
```

Problem 2: Image Compositing

(a) Following the formula given in the exercise, we have

1. A **over** $B = (1.0, 1.0, 1.0, 1.0) = A$
2. A **over** $B = (0.5, 0.5, 0.5, 0.75)$
3. A **over** $B = (0.5, 0.5, 0.5, 0.5) = B$
4. A **over** $B = (0.0, 0.0, 0.0, 0.0)$

(b) In the case of non-overlapping areas,

$$A \textbf{ over } B = \left(c'_{(A \textbf{ over } B)} = c'_A + c'_B, \alpha_{(A \textbf{ over } B)} = \alpha_A + \alpha_B\right)$$

where c' indicates pre-multiplied colors; from the formulae it is clear that A **over** B = B **over** A. Since A and B do not overlap, their total area can cover no more than the total size of a pixel, i.e. $\alpha_A + \alpha_B \leq 1$.

Problem 3: Ray-Sphere Intersection

(a,b) The easiest way to solve this exercise is to draw the spheres and the ray. Following are the solution for each sphere position.

1. $\mathbf{O} = (0, 0, 0)-$ intersected at P=(1,0,0) and (-1,0,0). The intersection is from the inside the sphere.
2. $\mathbf{O} = (3, 0, 0)-$ intersected at P=(2,0,0) and (4,0,0).
3. $\mathbf{O} = (1, 1, 0)-$ intersected at P=(1,0,0). The ray is tangent to the sphere, i.e. there is only one intersection.
4. $\mathbf{O} = (-3, 0, 0)-$ intersected at P=(-2,0,0) and (-4,0,0).
5. $\mathbf{O} = (0, 3, 0)-$ not intersected.

Problem 4: Ray-Disk Intersection

To determine the ray-disk intersection, first intersect the ray with the plane passing by $\mathbf{O}$ with normal $\mathbf{N}$. If the ray is not parallel to this plane, let's define $\mathbf{P}$ as the intersection between the ray and the plane given in class. The ray will intersect the disk if $\|\mathbf{P} - \mathbf{O}\| \leq r$.

Extra credit: Ray-Cylinder Intersection

Let $\mathbf{r}(t) = \mathbf{e} + \mathbf{i}t$, then, by substitution, the ray $\mathbf{r}$ intersects the cylinder when

$$\begin{cases} (e_x + i_x t)^2 + (e_y + i_y t)^2 = r^2 \\ -h/2 \leq e_z + i_z t \leq h/2 \end{cases}$$

To find the intersection, expand the quadratic equation

$$(i_x^2 + i_y^2)t^2 + 2(e_x i_x + e_y i_y)t + (e_x^2 + e_y^2 - r^2) = 0$$

and solve it to find the parameter values of t in the same manner as the sphere case given in class. Once the appropriate value of t is found, check if it obeys to the inequalities too.