CPSC 4040/6040
Computer Graphics Images

Assigned: Sept. 24, 2015
Due: Oct. 1, 2015 (start of class)

Quiz 2 *(Grading: 0–10 points)*

1. `http://www.brucelindbloom.com/index.html?ColorCalculator.html`
   provides a calculator for converting colors from RGB to CIE spaces. It also gives instructions and technical details for how the conversion works. In class we learned about the CIE xyY color space and how there are colors that we can see which cannot be described by the RGB color system.

   RGB can be expressed as a triangle overlaid on the CIE xyY space. Part 1 of this question: using the calculator, determine the xy coordinates of the triangle's corners. To do so, first make sure you specify the correct reference white of "D65" as well as the RGB model and Gamma of "sRGB". To find the corners of the triangle, you will have to convert the primary RGB colors to xyY, specifying each channel as a number between 0.0 and 1.0. Note that, for any other valid RGB color you specify, they will all fall as some point within the triangle (you may want to use this fact to verify correctness).

   Next, given a fixed Y value of 1.0, identify a xy pair such that when it is converted to RGB the R channel has a negative value. Also identify a xy pair such that the R channel has a value greater than 1.0.

2. Draw and label a diagram of the HSV color space. Include a brief description of each variable, its meaning, and its range.

3. Following are two RGBA colors specified as floating point values between 0.0 and 1.0 for each color channel. Give the associated colors for the following pixels labeled A and B (specifying all numbers as decimal values between 0.0 and 1.0). Finally, give the color of the pixel described by A **over** B.

   Color A: $(R, G, B, \alpha) = (0.5, 0.25, 0.25, 0.25)$

   Color B: $(R, G, B, \alpha) = (0.2, 0.8, 0.2, 0.75)$

4. Given an 8-bit, greyscale image of width `W` and height `H`, declared as

   ```
   unsigned char image[H][W];
   ```

   Write a small block of C++ code that computes the histogram for this image. You must properly allocate an array of *sufficient size* and using the correct *data type* to store the histogram as well as *populate it correctly*.

5. The code in Listing 1 is intended to perform histogram equalization as two steps given an input histogram (the variable `histogram` below). First, a lookup table, `lut`, is computed that stores the transformation from input color values to output color values based on scaling relative to the input histogram. Next, the lookup table is applied to each pixel in the image. However, this code contains at least three bugs. Please identify at least two bugs and correct them.

Listing 1: Where is it broken?

```
1  //you may assume that W,H are set to the correct values
2  const int W,H;
3
4  unsigned char lut[W*H];
5
6  for (int bin = 0; bin < 256; bin++) {
7     int sum = 0;
8     for (int k=0; k<=bin; k++) {
9        sum += histogram[k];
10    }
11    //make sure the lut stores unsigned chars
12    lut[bin] = ( sum / (W*H) ) * 255;
13 }
14
15 for (int row = 0; row < W; row++) {
16    for (int col = 0; col < H; col++) {
17       image[row][col] = lut[image[row][col]];
18    }
19 }
```