CS 559: Computer Graphics

Homework 2

This homework must be done individually. Submission date is Tuesday, February 19, 2001, in class.

Question 1:

The $L^*u^*v^*$ space (defined below) is approximately perceptually uniform. Hence, one way to decide whether two pairs of colors in RGB space, say a, b and c, d, are separated by the same perceptual distance is to first convert all the colors into LUV space then compute their relative distances there using a standard distance metric. To get from RGB to XYZ, use the following matrix:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0.73467 & 0.27376 & 0.16658 \\ 0.26533 & 0.71741 & 0.00886 \\ 0.00000 & 0.00883 & 0.82456 \end{bmatrix} \begin{bmatrix} r \\ g \\ b \end{bmatrix} \tag{1}$$

LUV coordinates, (L^*, u^*, v^*) are computed in several steps. First compute (X_n, Y_n, Z_n) which are the XYZ coordinates of white. Then compute the following four values:

$$u' = \frac{4X}{X + 15Y + 3Z} \tag{2}$$

$$v' = \frac{9Y}{X + 15Y + 3Z} \tag{3}$$

$$u'_n = \frac{4X_n}{X_n + 15Y_n + 3Z_n} \tag{4}$$

$$v'_n = \frac{9Y_n}{X_n + 15Y_n + 3Z_n} \tag{5}$$

Finally, compute:

$$L^* = 116\left(\frac{Y}{Y_n}\right)^{\frac{1}{3}} - 16 \tag{6}$$

$$u^* = 13L^*(u' - u'_n) \tag{7}$$

$$v^* = 13L^*(v' - v'_n) \tag{8}$$

When $Y/Y_n < 0.01$, $L^* = 903.3Y/Y_n$, rather than the equation above. Note that when $r = g = b$, $u' = u'_n$ and $v' = v'_n$ and hence $u^* = v^* = 0$. In other words, the L^* component of $L^*u^*v^*$ encodes intensity. You might like to write a program to do the color conversions.

For each of the following pairs of RGB colors, transform them into $L^*u^*v^*$ space. Then, for each pair, compute the distance between the two colors using the standard Euclidean distance function:

$$d = \sqrt{(L_0^* - L_1^*)^2 + (u_0^* - u_1^*)^2 + (v_0^* - v_1^*)^2}$$

a. (1.0, 0.0, 0.0) and (0.9, 0.0, 0.0)

b. (0.0, 1.0, 0.0) and (0.0, 0.9, 0.0)

c. (0.0, 0.0, 1.0) and (0.0, 0.0, 0.9)

Question 2: Consider an image format that uses indexed color (it stores a table of colors and then each pixel is an index into the table). Let n be the number of pixels in the image, let k be the number of bits for the index at each pixel, and assume that the table uses b bits for each color it stores.

a. How many bits are required to store the color table?

b. How many bits are required to store the pixel data?

c. How many bits are required in total? (Ignore the requirement to store the width and height or other information.)

d. Consider an image with 500000 pixels (about 800×600). Fix $b = 32$. How many bits are required if $k = 8$? How many are required for $k = 16$?

e. For the same 500000 pixel image, fix $k = 8$. How many bits are required for $b = 64$?

f. Is the size of a GIF file more sensitive to the number of colors in the color table, or the number of bits used for each color in the table?

Question 3: Consider a variant of Floyd-Steinberg dithering in which the error at a pixel is distributed to only three of its neighbors:

e	+3/8
+3/8	+1/4

a. What happens if you run this version on an image with constant intensity of 0.5? What are the artifacts?

b. One possibility is to only distribute error to the pixel ahead of the current one on the same row. Why is this a bad idea? (Hint: Consider an image with a white left edge and a black right edge, with a gray ramp in between.)

c. Another possibility is to only distribute error to the pixels below the current one. Why is this a bad idea?