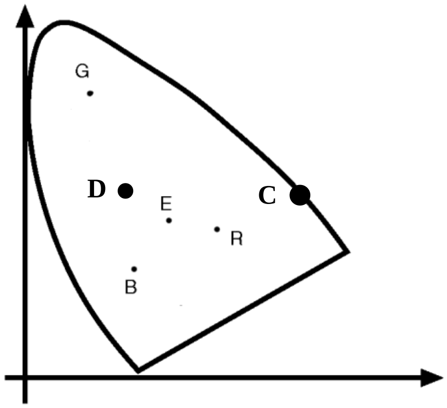
CS576 Assignment 2
Instructor: Parag Havaldar
Assigned – Monday 02/22/2021
Solutions due – Wed 03/10/2021 by 10 am morning
Late Policy - NONE

Question1: Color Theory – 10 points

One of uses of chromaticity diagrams is to find the gamut of colors given the primaries. It can also be used to find dominant and complementary colors –
Dominant color of a given color D (or dominant wavelength in a color D) is defined as the spectral color which can be mixed with white light in order to reproduce the desired D color. *Complementary colors* are those which when mixed in some proportion create the color white. Using these definitions and the understanding of the chromaticity diagram that you have, answer the following.



- In the image alongside find the dominant wavelength of color D. Show this wavelength. *(2 points)*
- Do all colors have a dominant wavelength? Explain your reasoning. *(3 points)*
- Find the color which is complimentary to the color C. Show this color *(2 points)*
- What colors in RGB color space map to the equiluminous point E upon projection into the chromaticity space. *(3 points)*

Question 2: Color Theory (10 points)

- The chromaticity diagram in (*x, y*) represents the normalized color matching functions X, Y and Z. Prove that *(2 points)*

$$Z = [\ (1-x-y)/y\ ]\ Y$$

Here you are tasked with mapping the gamut of a printer to that of a color CRT monitor. Assume that gamuts are not the same, that is, there are colors in the printer's gamut that do not appear in the monitor's gamut and vice versa. So in order to print a color seen on the monitor you choose the nearest color in the gamut of the printer. Answer the following questions

- Discuss (giving reasons) whether this algorithm will work effectively? *(2 points)*
- You have two images – a cartoon image with constant color tones and a real image with varying color tones? Which image will this algorithm perform better – give reasons. *(2 points)*
- Can you suggest improvements rather than just choosing the nearest color? *(4 points)*

Entropy Coding – 20 points

Consider a communication system that gives out only two symbols X and Y. Assume that the parameterization followed by the probabilities are $P(X) = x^2$ and $P(Y) = (1-x^2)$.

- Write down the entropy function and plot it as a function of *x.(1 + 3 points)*
- From your plot, for what value of *x* does the Entropy become a minimum? At what values of *x* is the Entropy a maximum? .*(2 points)*
- Although the plot visually gives you the value of *x* for which the entropy in maximum, can you now mathematically find out the value(s) for which the entropy is a maximum? *(6 points)*
- Can you do the same for the minimum, that is can you find mathematically prove the value(s) of *x* for which the entropy is a minimum? *(8 points)*

Generic Compression – (20 points)

The following sequence of real numbers has been obtained sampling a signal:
5.8, 6.2, 6.2, 7.2, 7.3, 7.3, 6.5, 6.8, 6.8, 6.8, 5.5, 5.0, 5.2, 5.2, 5.8, 6.2, 6.2, 6.2, 5.9, 6.3, 5.2, 4.2, 2.8, 2.8, 2.3, 2.9, 1.8, 2.5, 2.5, 3.3, 4.1, 4.9
This signal is then quantized using the interval [0,8] and dividing it into 32 uniformly distributed levels.

- What does the quantized sequence look like? For ease of computation, assume that you placed the level 0 at 0.25, the level 1 at 0.5P, level 2 at 0.75, level 3 at 1.0 and so on. This should simplify your calculations. Round off any fractional value to the nearest integral levels *(4 points)*
- How many bits do you need to transmit the entire signal? *(2 points)*
- If you need to encode the quantized output using DPCM. Compute the successive differences between the values – what is the maximum and minimum value for the difference? Assuming that this is your range (ie, ignore first value), how many bits are required to encode the sequence now? *(4 points)*
- What is the compression ratio you have achieved (ignoring first value)? *(1 point)*
- Instead of transmitting the differences, you use Huffman coded values for the differences. How many bits do you need now to encode the sequence? Show all your work and how you arrived at the final answer *(5+3 points)*
- What is the compression ratio you have achieved now (ignoring first value)? *(2 points)*

Programming Part (*140 points*)

This assignment will help you gain a practical understanding of analyzing color channels especially as it pertains to image segmentation. Image segmentation is a challenging task in the field of image processing and computer vision. In order to obtain an accurate segmentation performance, user interaction is always used in practical image-segmentation applications. Here you will implement one such application where the image pixels need to be segmented based on two threshold values in the HSV color space, using the hue values as thresholds. All the pixels falling between these two hue thresholds will be displayed in the original color in the output image, whereas all the other pixels outside the threshold will be displayed in gray.

You will be given input images in the usual rgb format. You are free to use extend the display code sample given of assignment 1 to implement this project or you may write your own in any language of your choice (*no scripting languages such as MATLAB or python please!*).

Recall that the HSV color space (Hue, Saturation and Value) describes Hue as a number between 0 and 360, 0 being closer to red
The input to your program will take three parameters where

- The first parameter is the name of the image, which will be provided in an 8 bit per channel RGB format (Total 24 bits per pixel). You may assume that all images will be of the same size for this assignment of size 512 x 512
- The second parameter will be a number between 0-360. This will provide the first hue threshold ***h1*** for deciding your segmentation boundary.
- The third parameter will be a number between 0-360. This will provide the second hue threshold ***h2*** for deciding your segmentation boundary. This number will be greater than the previous number.

To invoke your program we will compile it and run it at the command line as

```
YourProgram.exe C:/myDir/myImage.rgb h1 h2
```

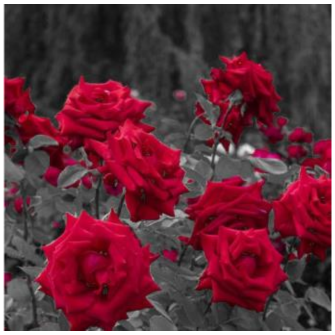
where *h1* and *h2* are the parameters as described above. To run your program you will need to convert your rgb values Example inputs are shown below and this should give you a fair idea about what your input parameters do and how your program will be tested.

1. *YourProgram.exe roses_image1.rgb 0 359*

The left image is your input and the right image is the produced output. Here the thresholds h1 and h2 scan through all colors 0 to 359, so the output will be same as the input.

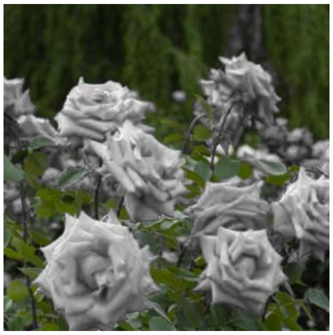
2. *YourProgram.exe roses_image1.rgb 320 359*

The left image is your input and the right image is the produced output. Here the thresholds h1 and h2 scan through 320 to 359 which is the red part of the HSV space. Correspondingly the output is shown below.
 input.



3. *YourProgram.exe roses_image1.rgb 60 120*

The left image is your input and the right image is the produced output. Here the thresholds h1 and h2 scan through 160 to 120 which falls in the green spectrum of the HSV space. Correspondingly the output is shown below.



Now for the details - You will need to convert your image from an RGB representation to a HSV representation. Unlike other color space conversions, there is no matrix to linearly convert to the HSV color space. The HSV color space is represented as a cylindrical coordinate system with H varying from 0 (near violet/blue) to 360 (red), where as H and V both vary from 0 to 1. You can read about the color conversion here:
https://en.wikipedia.org/wiki/HSL_and_HSV
https://www.cs.rit.edu/~ncs/color/

Once in the HSV space, you can appropriately use the thresholds h1 and h2 to perform the assignment.

What should you submit?

- Your source code files, and your project file or makefile, if any – all combined into a zip archive with your name on the archive. ***Please do not submit any binaries or images***. We will compile your program and execute our tests accordingly.
- If you need to include a readme.txt file with any special instructions on compilation, that is fine too.