CS443/543 HW#2

Due: Midnight on February 20

You may use any programming language or API to implement and display images, but you must implement the algorithms by yourself and should not use existing similar functions to generate the output images. For part 1 and 2, use three color images in HW2_sample_images.zip as your input images.

You must not use any exisiting gamma correction functions or rgb2ycbcr() for this homework. You must implement each one by yourself.

Part 1:

Implement and apply *Gamma correction* to correct image intensity for color images. To deal with the discrepancies between color as seen by people and sometimes different color displayed on our monitor, the input signal to the monitor (the voltage) can be "gamma corrected." To reproduce the image faithfully on a computer screen, you will apply the inverse power function (1/gamma) to the target input image before displaying it, compensating for the non-linearity which will be introduced. The trick is to modify the brightness of the pixels without changing their perceived color.

For each of your 3 images, your program will generate a Gamma-corrected image. In your report, you must describe which gamma value you used.

What to hand in:

- Your code.
- Three output images: all of your files must include your last name, part number, input or output, and image input number (chung_part1_[input|output]_image1.png).
- YOU MUST WRITE A REPORT. The report should basically show the input and output results and short description (your observation) about how images are changed after the gamma correction (based on your gamma value).

Part 2:

In this assignment, the goal is to visualize and understand YCbCr color model.

$$\begin{bmatrix} Y' \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.168736 & -0.331264 & 0.5 \\ 0.5 & -0.418688 & -0.081312 \end{bmatrix} \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} + \begin{bmatrix} 0 \\ 0.5 \\ 0.5 \end{bmatrix}$$

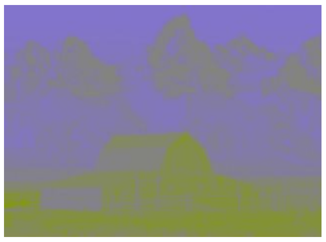
Write a program that reads an image file and does the following:

a) Convert your RGB image to YCbCr.

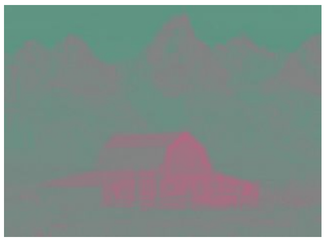
b) Generate output images from the 'luminous' component.

c) Generate Cb image as a combination of blue and yellow; and generate Cr as a combination of red and cyan image (see chap4 slide). For example, set the range of Cb (Cr) in [0..255]. If Cb value is greater than 127, display pixel in blue tones, otherwise display it in yellow tones. As the distance from 127 increases the saturation should increase. For example, the low values should be represented with high yellow values whereas high values should be represented with high red values.

e.g.,



Cb



Cr

What to hand in:

- Your code.
- Three output images: all of your files must include your last name, part number, input or output, and image input number (chung_part2_[input|output]_image1.png)

Enjoy!