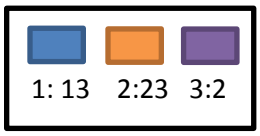
CS/ECE 545 Digital Image Processing Homework 4, Spring 2016 (Due Tue May 3rd, by 11:59 PM)

1. (25 Points) Burger & Burge Exercise 12.2 (page 311). Create an ImageJ plugin called **Indextable_Display** that shows the color table of an 8-bit indexed image as a new image with 16 × 16 rectangular color fields. Use the following ImageJ function to label each color with the **frequency** and **index color value** in the image. Mark all unused color table entries with the value 0. The new image should look similar to the image below but with 16 x 16 squares (white background, each color represented by a square of that color, and below the colored square a label with the index # and frequency). Look at Prog. 12.3 as a starting point.

C.13.1 ImageProcessor (Class) – page 499.

void drawString (String *s*, int *u*, int *v*): Draws the string *s* at position (*u*, *v*) using the current fill/draw value and font.



2. (50 Points) Create an ImageJ plugin called **Color_Transfer** that implements the Reinhard et al. 2001 paper (IEEE Computer Graphics and Applications). Please read the paper thoroughly before attempting the implement it. Test your plugin using the images on the class site of "*Vincent van Gogh's Cafe Terrace on the Place du Forum*" as the color profile to apply to the fuller labs photo, which obviously needs some color.

The paper is located on the class site (http://users.wpi.edu/~clindsay/teaching/CS545/ColorTransfer.pdf)

3. (25 Points) Burger & Burge Exercise 14.1 (page 366): Implement the two-dimensional DFT using the one-dimensional DFT in a plugin named **DFT_2D**, as described in Sec. 14.1.2. Apply the 2D DFT to real intensity images of arbitrary size and display the results (by converting to ImageJ **float** images). Implement the inverse transform and verify that the back-transformed result is identical to the original image. You plugin should display the 3 images (Orig, DFT image, Inverse DFT image).

4. (50 Points) Motion Deblur via Deconvolution: Read the following paper "COMPARISON OF MOTION DEBLUR ALGORITHMS " found on the class site. Then implement an imageJ plugin named **Motion_Deblur** that performs the direct deblurring method in this paper (**Section 3.1 in paper**). Assume there is **no noise**, which simplifies the method. Assume the **motion is 2D**. Make sure to pay attention to padding. Your job is to find the blur kernel that best represents the motion blur by searching the kernel's space and repeatedly de-convolving the image with the filter. Use the methods from the previous question to implement your plugin. In your search, use the error metric Peak signal-to-noise ratio (PSNR) defined in Section 4.3. Search should terminate when you reach a suitable error (see Table 2 in paper for

possible values). Write all methods from scratch (ie don't use predefined frequency domain methods in imageJ), except of course the Java math functions of the math functions in the imageProcessor class.

Important Note: Some of the equations from the book contains errors especially the color Image chapter. Before using any equations, please check the book's errata and use the correct form of each equation. The book's errata is located at:
https://imagingbook.files.wordpress.com/2013/06/burgerburge-en1-errata.pdf

Submit (Indextable_Display , Color_Transfer, Motion_Deblur, DFT_2D) and your README file in Word or PDF with all your **answers clearly typed up!!!**

Put everything into ONE zip file named *yourfirstname_yourlastname_hw4.zip* using Turnin.
DON'T EMAIL ME YOUR HOMEWORK. ALSO, TEST YOUR CODE BEFORE SUBMITTING, COMPILE ERRORS WILL BE MARKED DOWN!!