Color Project

This project requires that you implement four programs to manipulate color images.

OpenCV support

Color conversion among the color spaces that were discussed in class can be achieved using OpenCV. The OpenCV representation of these color spaces is different from what was discussed in class in the following way: all values are mapped into the $[0-255]$ range and rounded to the nearest integer. Here is the OpenCv conversion routine:

```
output_image = cvtColor(input_image, cv2.FLAG)
```

Several FLAG values that are relevant to what was discussed in class are:

COLOR_XYZ2BGR, COLOR_BGR2XYZ,
COLOR_Lab2BGR, COLOR_BGR2Lab,
COLOR_Luv2BGR, COLOR_BGR2Luv.

The project also requires OpenCV code for histogram equalization. Here is how it is used:

```
output_gray_image = cv2.equalizeHist(input_gray_image)
```

Comments related to all programs

We will test your programs using an automatic script. if your programs fail the script because of issues mentioned below you will lose points. Here are the requirements:

- Each program should be fully contained in a single file, named as explained below. (You are not allowed to share files between programs.)
- Do not use Python / OpenCV unusual libraries unless you first clear it up with the TA.

Programs

First program: linear stretching

Please name this program **ls.py**. It gets as input the names of the input and the output images. The output is computed from the input by linearly stretching the L values in the Luv color space to their fullest range. It should be executed as follows:

```
python3 ls.py inputimage outputimage
```

You may want to use the provided template program to handle input and output of images.

Second program: histogram equalization

Please name this program **he.py**. It gets as input the names of the input and the output images. The output is computed from the input by histogram equalization applied only to the L values in the Luv color space. It should be executed as follows:

```
python3 he.py inputimage outputimage
```

You may want to use the provided template program to handle input and output of images.

Third program: adaptive linear stretching

Please name this program **als.py**. It gets as input a window parameter w and names of the input and the output images. The output is computed from the input by changing only the L values. The output $L(i, j)$ is computed by applying linear stretching to the window of size $(2w+1 \times 2w+1)$ centered at i, j. The L value at the center of the window is the output value $L(i, j)$. It should be executed as follows:

```
python3 als.py w inputimage outputimage
```

You may want to use the provided template program to handle input and output of images.

Fourth program: adaptive histogram equalization

Please name this program **ahe.py**. It gets as input a window parameter w and names of the input and the output images. The output is computed from the input by changing only the L values. The output $L(i, j)$ is computed by applying histogram equalization to the window of size $(2w+1 \times 2w+1)$ centered at i, j. The L value at the center of the window is the output value $L(i, j)$. It should be executed as follows:

```
python3 ahe.py w inputimage outputimage
```

You may want to use the provided template program to handle input and output of images.

Experiments

Program 1: You are asked to submit two images and describe the results in a short paragraph. One image shows an example where histogram stretching significantly improves the image. Another image shows an example where histogram stretching does not improve the image. The images should be named p11.extension, p12.extension.

Program 2: Same as Program 1. The images should be named p21.extension, p22.extension.

Program 3: You are asked to submit two images and describe the results in a short paragraph. One image shows an example where the image is better for a small value of w and becomes worse for larger w values. The other image shows an example where the image is better for a large value of w and becomes worse for smaller w values. The images should be named p31.extension, p32.extension.

Program 4: Same as Program 3. The images should be named p41.extension, p42.extension.

Uniqueness of images

The images you submit to describe your experiments must be unique. You will lose points if someone else in class submits the same image. One way of making sure your image is unique is to post it on elearning with a short description.

What you need to submit

- Python code of all programs.
- 8 images.
- Documentation describing your selection of the images.

These items should be put in a zip file named with your netid and submitted on elearning. For example, if your netid is xyz1234 you should submit a zip file named xyz1234.zip.

Due Date: TBA