ECE-178 Homework #2.
Due on Wednesday, October 16, 2024, 11:59 PM

Name:
TA:

A. Written Problems:

1. An imaging lens in a digital camera has a focal length of 6 cm. How far should the lens be from the camera's imaging plane (detector) to focus on an object
    (a) 12 cm in front of the lens?
    (b) 15 cm in front of the lens?

2. You are given an 8-bit grayscale image where each pixel value P ranges from 0 to 255. You are asked to adjust the brightness of the image using gamma correction.
    (a) Give the gamma correction formula for this problem
    (b) Suppose the original pixel value is $P = 100$. Apply gamma correction with $\gamma = 0.5$ and calculate the corrected pixel value P_{corr}.
    (c) Now, apply gamma correction with $\gamma = 2.2$ for pixel value $P = 100$. What is the new corrected pixel value P_{corr}?
    (d) Explain the visual effect of gamma correction with values of γ less than 1, equal to 1, and greater than 1 on an image's brightness and contrast.

B. Programming Assignment:

For this part of the assignment, you will write simple Python functions that apply operations to a given image. Each function takes one or two images (as appropriate) and then returns a new image with the required modifications. You will then chain those operations together to create a new image of your own.

Note that the images you will be working with are RGBA, which means that in addition to the standard red, green, and blue channels they also have an alpha channel that specifies the opacity of each pixel. If you read in an image without an alpha channel, assume that the alpha = 1 for all pixels.

The first set of functions to create take in only one image as input:

- Flip(A): flips image A vertically
- Flop(A): flips image A horizontally
- Inv(A): inverts image A (i.e., dark colors become light and vice-versa)

- InvAlpha(A): inverts alpha channel of image A (opaque things become transparent and vice-versa)

Once you are done, write some *compositing operators* that will allow you to composite one image with another. For these, we will follow the operators defined by Porter and Duff (see paper on GitHub resource directory). A good explanatory webpage can be found here. Implement the following functions:

- Over(A, B)
- Atop(A, B)
- Xor(A, B)

With these functions implemented, perform a series of operations on an image. For example:

I = Over(Flip(A), Flop(A))
which composites a flipped image over a flopped image of itself.

On GitHub, we are providing some images for you to experiment with. In particular, we would like you to implement the following functions and show your results:

- out = Atop(Flip(HW2_dest), Flop(HW2_source))
- out = Atop(Flop(HW2_source), Flop(HW2_dest))
- out = Xor(HW2_source, HW2_dest)
- out = Xor(HW2_dest, HW2_source)
- out = Over(HW2_Lemur, HW2_Background1)
- out = Over(Flop(HW2_Trolls), (HW2_Background2))

Finally, beyond the examples we provide, we would like you to make your own image by writing your own operation equations on images. Experiment around and try to come up with something that looks cool and submit that along with your code.

For each result, save the final images. In your submission, please provide your code (your .py files) along with your input and output images in a single .zip or .tar file. Please use the following convention to name the zip/tar files: <First name>_<Last name>_HW2.<zip/tar>.