CS/ECE 545 Digital Image Processing
Homework 3, Spring 2016 (Due Wed April 13, by 6PM)

1. Burger & Burge Exercise 9.4 (page 171) (30 points): Implement as an ImageJ plugin, called **Circular_Hough**, the Hough Transform for finding circles with varying radii. Make use of a fast algorithm for generating circles, such as described in sec 9.4, in the accumulator Array. Suitable images to test your algorithm will be posted on the class site. Test at least one image with your submission. Also explain briefly how your algorithm works in your comments in the code.

2. Using the Hough Transform from question 1, modify the code to perform 2 types of flood fill algorithms to fill in each circle found in the image (25 points). The result should be a binary image with the foreground contains only the filled-in circles and the background is 0. The plugin should be named **Circle_Flood** and should contain a Boolean variable which could be modified switch between the flood fill methods. The first method should be the naïve recursive flood fill algorithm (see Algorithm 11.1 in book). The second method you should implement is the Coherence method described in class (slide ~59).

3. Laplacian edge detection method (45 points): The goal of this assignment is to implement the Laplacian edge detection method. You should implement separable convolution routines. You will just have to call these routines with different kernels (Gaussian and 1st and 2nd derivatives of Gaussian). If you do use a helper class, be sure to turn in the source code for it as well. Your filters should all work with floating point images as input. Also, all convolutions should be separable, use wrapping on the boundary, and all filters that use Gaussians should have a (σ) parameter that you can set.
    a. (5 pts) Create a plugin called **Gradient_Magnitude** that computes the gradient magnitude of an image.
    b. (10 pts) Create a plugin called **Laplacian_Image** that computes the Laplacian of an image. Use Gaussian 2nd derivative filters.
    c. (15 pts) Write a plugin filter called **Zero_Crossings** that calculates the zero crossings in an image. It should set a pixel to 1 if it is a zero crossing and 0 otherwise.
    d. (15 pts) Put it all together! Use the three filters above to do edge detection. Now you should have a threshold parameter to apply a threshold to the gradient magnitude. Follow this by an AND operation with the result of the Laplacian zero-crossing image. Your resulting images should be binary, with 1 on the edges and 0 elsewhere. Call your plugin filter **Laplacian_Edge_Detection**.

4. Use any of the previous questions implementations (or none) to help design an algorithm that uses **Morphology** to measure the size of the pupils of the two images on the class website (50 points). *NOTE: A detailed description of this problem has been left out on purpose to force you think about how one might solve this problem.* Measurement the pupil size will be based on the radius of pupil detected. For this assignment, we will define the pupil as simply the opening of the eye for which light enters the inner part of the eye (https://en.wikipedia.org/wiki/Pupil). For simplicity (not reality) provide the measurement in terms of the number of pixels. Create a plugin called **Pupil_Measure**.

Submit all ImageJ plugins (Circular_Hough.java, Circle_Flood.java, Gradient_Magnitude.java, Laplacian_Image.java, Zero_Crossings.java and Laplacian_Edge_Detection.java, Pupil_Measure.java). Along with the plugins, a readme.txt file is required with your name, email, and a description of the files included in the zip file.