Rensselaer Polytechnic Institute
Department of Electrical, Computer, and Systems Engineering
ECSE 4540: Introduction to Image Processing, Spring 2018

Homework #6: due Monday, Mar. 26^{th}, at the beginning of class.
Show all work for full credit!

For this homework, you'll need the images in the file `http://www.ecse.rpi.edu/~rjradke/4540/hw6.zip`.

1. (15 points.) Write a Matlab function that calls the `superpixels` command with a number of superpixels K to show (1) a figure with the K superpixel boundaries superimposed on the original image, and (2) a figure where each of the K superpixels is replaced with the average color inside the region, creating a "stained glass" effect. Hint: `doc superpixels` describes the necessary process. Now apply your function to the `glass.png` image. Include with your homework plots and corresponding values of K that result in "too many" superpixels (i.e., oversegmentation), "too few" superpixels (i.e., undersegmentation) and "just right".

2. (15 points.) Use the Matlab Image Segmenter to investigate the performance of active contour based segmentation on the images `elk.png` and `fish.png`. For each image, first draw a freehand initial contour that roughly segments the object of interest (without trying to get too close to the actual boundary). Then apply both the edge-based segmentation algorithm with 600 iterations (click on the "Method" area at upper left to change it) and the region-based segmentation algorithm with 300 iterations (the default). For each of the images, include a picture of your initial contour, the edge-based segmentation, and the region-based segmentation. Critically discuss the behavior of each technique for each image. What was the overall speed? Where did each segmentation algorithm work well, and where did it fail (and why)?

3. (25 points.) Load the image `tomatoes.png`. The goal in this problem is to try to get a good estimate of the number of tomatoes using image processing. You can see that almost every tomato has a single glint of white on its surface from the reflection of light. Write an image processing algorithm to detect and count these white spots. Hint: by using the Data Cursor in the Matlab figure window (or a Paint program), you can look at the color values of the white pixels (generally high values of R, G, and B) as well as the color values of the tomatoes (generally very high values of R, middle values of G, low values of B). You can create a binary image by thresholding the color channels at different values to isolate just the white pixels. Adjust your threshold so you visually get a white spot on each tomato. On the other hand, you probably will need to use `regionprops` to find and remove spurious detections that only contain a small number of pixels. Finally, use `regionprops` and `plot` to find the centroids of the remaining blobs and plot these locations as big green dots on top of the original image. How many tomatoes did you find with your algorithm? For full credit, be sure to include full details about your algorithmic process, including the values of any thresholds you used.

More fun on the next page ⟶

4. (25 points.) Load the image `buttons.png`. The goal in this problem is to use operations on binary images to extract the colored buttons on the cards. Here's the rough process you should use:

   - Find the white cards by creating a binary image by finding pixels whose R, G, and B values are all above a high threshold. This step will also pick up the white shelves in between the racks of buttons.
   - Use the `bwfill` command to fill in the holes in the image (i.e., the locations of the buttons).
   - Determine the logical operation on the previous two images that will result in an image with just the button locations (and small areas of extra junk).
   - Use `regionprops` to isolate only those binary regions that are (1) sufficiently large and (2) sufficiently circular (e.g., using the `Eccentricity` or `MajorAxisLength/MinorAxisLength` options).

   Provide an image and some discussion for each intermediate step of your process, including the values of any thresholds you used. Which buttons did your algorithm miss at each step and why?

5. (20 points.)

   (a) Use `strel` to create a structuring element `d3`, a disk of radius 3.

   (b) Apply `d3` to erode the binary image in `drink.png` using `imerode`. Describe what happened and why.

   (c) Apply `d3` to dilate the result of part (b) using `imdilate`. Describe what happened and why. Note that the result is what you would have gotten by applying `imopen` to the original image with structuring element `d3`.

   (d) Now suppose we were to reverse the order of erosion and dilation (i.e., morphological closing instead of morphological opening). Explain what you see, and discuss why the two results are different.