Name: _____

Score: /19

CSE 5524 Computer Vision for HCI

Homework Assignment #3

Due: See Carmen for due date

---

1) Generate a 3-level Gaussian pyramid (original image is level-0) and the corresponding Laplacian pyramid of an image (select one from the web, make it grayscale). Use the formula in the notes to first determine a viable image size (use N=3, and pick N_C and N_R), and crop the image (if needed) to test the pyramid code. Use a=0.4 for the Gaussian mask – **use separable masks!** Write/use functions for properly reducing and expanding an image. Write your own interpolation function - do not use Matlab/Python in-built interpolation functions (e.g., interp2). Lastly, perform a reconstruction of the original (cropped) image using the Laplacian pyramid. [8 pts]

2) Using the grayscale images (walk.bmp, bg000.bmp) provided on the WWW site, perform background subtraction 1 (abs diff) to extract the object. Make sure your image is of type *double*! Experiment with thresholds and discuss. [2 pts]

3) Using the grayscale images (walk.bmp, bg[000-029].bmp) provided on the WWW site, perform background subtraction 2 using statistical distances. Experiment with thresholds and discuss. [5 pts]

4) Dilate your best binary image resulting from problem 3) using: [1 pt]

```
from scipy import ndimage
d_bsIm = ndimage.binary_dilation(im, structure=np.ones((3,3)))
```

5) Next perform a connected components algorithm, and keep only the largest region in L (save/display as an image). [1 pt]

```
from skimage.measure import label, regionprops, regionprops_table
labels = label(im)
regions = regionprops(labels)
properties = regionprops_table(labels, im, properties=['area','convex_area','bbox_area'])
```

Turn in a report containing all code, printouts of images, and discussion of results. Make a script to do all tasks and call needed functions. Upload your report, code, and selected images to Carmen. [2 pts]

Python help: Loading several ordered grayscale images:

```
Im=np.array([io.imread('bg%03d.bmp' % i) for i in range(30)], dtype='float64')/255
```

The %03d gives a zero-padded number (e.g., 001), and the image storage is now a 3-D cube. To get image 3 from the cube, use myIm = Im[3,:,:] With such a cube, NumPy operations such as 'mean' can be told to work along certain dimensions of the cube.