Introduction

This assignment involves unsupervised texture segmentation as we discussed in class. Download the homework2.zip file from myCourses `Contents`→`Programming_assignments`→`Homework2`; this contains the instructions, starter code and image data needed for the assignment.

Requirements

You should perform this assignment using Python along with any image library of your choice, and it is due on **Saturday October 5th by 11:59pm**. You are required to submit your code in a Jupyter notebook along with a brief report containing short write-ups based on the question(s) in the assignment. Your solutions should be zipped and uploaded to myCourses via Assignments (formerly known as Dropbox) before the due date.

Your submitted zipped file for this assignment should be named **LastnameFirstname_hw2.zip** and should contain at least two files - *LastnameFirstname_hw2.pdf* and *LastnameFirstname_hw2.ipynb*. Feel free to submit any other auxiliary files required to run your code. We should be able to execute your code for the assignment from your submitted Jupyter notebbok. Include a `Readme` file if necessary (especially if using external libraries other than those used in the starter code).

You do not need to include any images with your final submission (but you should have images of your results embedded in your report). For grading, we will be testing your code on both the given images and on a different set of images you have not seen.

Problem 1. Foreground-background texture-based segmentation via clustering (Total 100 points)

Figure 1: Transferring an object from one image to another. Far left: original image; center left: segmented textured animal; center right: original background; far right: composite image - where we "grab" the animal from the first image and place it in the background image.

.

The goal of this problem is learn to segment an image that contains multiple textures into a foreground and background region, using a bank of N_{fil} filters. By convolving the image with each of the N_{fil} filters in the bank, each pixel is now transformed to an N_{fil}-dimensional vector. These vectors are then clustered using the k-means algorithm. A subset of the k segments that represents the foreground region is then transferred into a different background image. See Figure 1.

You are provided with an auxiliary file to help create the Leung-Malik (LM) bank of filters. This LM filter set is a multi-scale, multi-orientation filter bank with 48 filters. It consists of first and second derivatives of Gaussians at 6 orientations and 3 scales making a total of 36; 8 Laplacian-of-Gaussian (LoG) filters; and 4 Gaussians. We consider an over-complete bank where the filters occur at the basic scales $\{\sigma = \sqrt{2}, 2, 2\sqrt{2}, 4\}$. The first and second derivative filters occur at the first three scales with an elongation factor of 3 (i.e. $\sigma_x = \sigma$ and $\sigma_y = 3\sigma_x$). The Gaussians occur at the four basic scales while the 8 LoG filters occur at σ and 3σ. The filter bank is shown in Figure 2.

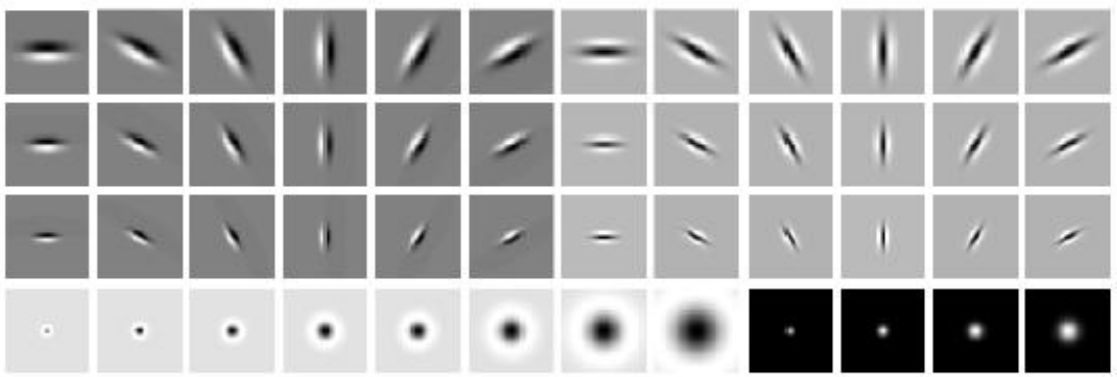

Figure 2: The LM filter bank has a mix of edge, bar and blob detectors at multiple scales and orientations. It has a total of 48 filters - 2 Gaussian derivative filters at 6 orientations and 3 scales, 8 Laplacian of Gaussian filters and 4 Gaussian filters.

You are also provided with an auxiliary functions to help you transfer pixels from a source image to the target image. Feel free to modify the auxiliary files or functions in case you want to use other external libraries such as OpenCV, but make sure to include any modified files with your submission. Also, include a `Readme` file with your submission if any major modifications are done.

Your tasks for the assignment include the following:

(a) Download the code and data from the zipped file. The data which is stored in the `\images` directory, consists of five textured animal images along with three background images, all named accordingly. You will be segmenting out each of these animals from the original images and transferring them to any background images of your choice.

(b) (15 points) Before implementing your segmentation you should write your own version of the k-means clustering algorithm although you are strongly advised to use the an implementation from an existing library such as scikit-learn for the rest of your work (The scikit-learn version has been optimized significantly). If you choose not to write your own version you will lose the points allotted for this portion of the problem. We will test your algorithm on a small set of cluster points.

(c) (50 points) You are now ready to segment. First create a function `segmentImg` and that can (#1) read in an image file (`.png` or `.jpg` file) and output a segmented version of the image. In this function, you will (#2) create a bank of filters and (#3)

convolve each filter in the bank with your input image. You are provided with a file `makeLMfilters.py` which returns a bank of 48 filters as shown in Figure 2. Then you will (#4) use the absolute value of your filter responses to construct a data matrix `X`. Finally, (#5) use k-means (either yours or the optimized library version) to cluster the 48-dimensional points in `X`. You should have a total of k clusters. Next, (#6) reshape your clustering result into the dimensions of the input image (without the color channels). (#7) Test with different values of k to see which gives you the best segmentation. The output of this should be a segmentation of the input image.

(d) (5 points) Next, use the given function `transferImg` to transfer your segments (obtained via k-means) into a new background image of your choice. The signature for `transferImg` is `[fgs, idxImg, sImg, tImg]`. The first variable `fgs` is a vector containing the indexes representing the foreground values from your segmentation, for example `fgs = [1, 3, 4]`. The second variable `idxImg` is the segmented image obtained from function `segmentImg` above. The third and fourth variables `sImg` and `tImg` are your source and target images respectively. The output of this function should be the composite image where the animal has been transferred into a new background.

(e) (10 points) Repeat Task (c and d) but this time, (#8)write a new function called `segmentImgClr` which takes the same arguments as before. Instead of using texture filters, this time, use color. (#9)Test out the new color-based segmentation and (#10) try out different colorspaces[1] (RGB, Lab, HSV) as your features. For different animals, compare your the color-based segmentation with the texture-based one.

(f) (20 points) A well-written, neat and concise report which includes a section for each of the following questions:

(i) Briefly describe your implementation, focusing on the interesting aspects. What are some artifacts and/or limitations of this technique, and what are possible reasons for them?

(ii) Indicate whether or not you implemented your own version of k-means clustering. If your report does not indicate this, we will not specifically check the code and it will be assumed that you did not; no grade will be given here.

(iii) For the dog image and two other animal images given, display the original source image and the segmented animal transferred to a new background. You can also use a plain white image as the background to showcase your segmentation. Display the texture-based segmentation alongside the color-based ones and discuss the differences.

(iv) As you would have noticed, many of the segmentations have holes and artifacts from the background in them. Discuss why this is the case and how different choices in our segmentation strategy could improve the results. Feel free to include additional input images or run additional experiments to illustrate your points.

You should turn in both your code and report in the zipped file to get full credit.

[1] I found the `a` and `b` channels of `Lab` to give the best color features for segmentation.