Project # 1
Image Segmentation
Assigned: Wed, April 18th, Due: Thu, May 10th

In this project you will be working in groups to experiment different image segmentation algorithms with different types of features. You are required to develop a simple GUI-based program to make it easy to conduct experiments and save results.

Segmentation Algorithms

You are required to test three different algorithms: *k-means*, *mean shift*, and *graph cut*.

Image Features

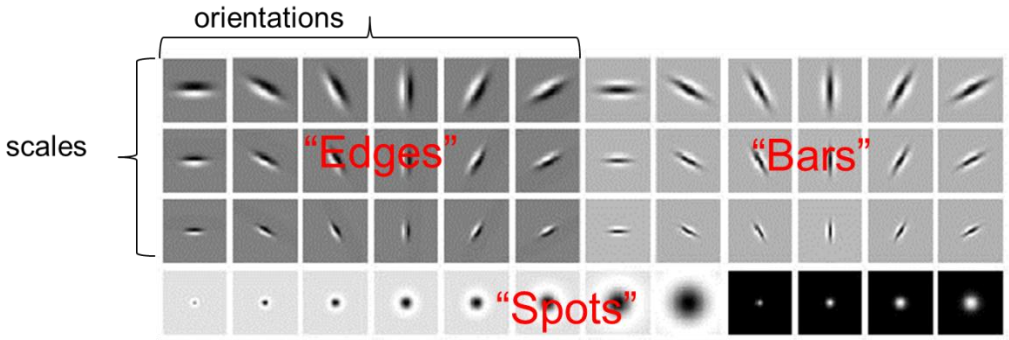

Figure 1 Leung-Malik Filter Bank

You are required to test each algorithm on different types of image features. The performance of each algorithm will depend on the nature of the input image, the features used, and the parameter setting for the algorithm. Here are the image features you will be testing your algorithms on.

- **Intensity:** This is the gray scale value for each pixel. If the input is a color image, convert to gray scale first.
- **Intensity and pixel coordinates:** Here we add the (x, y) coordinates of each pixel to its intensity value to make a 3D feature vector.
- **RGB color:** Color images are typically stored in the RGB color space. For gray scale images, use the intensity value for the three color channels.
- **YUV color:** Distance between RGB color vectors typically do not reflect human's perception of the difference between two colors. Sometimes, using other color spaces such as YUV makes algorithms behave closer to human's expectation. Convert input RGB color images to the YUV space using appropriate OpenCV functions for this experiment.
- **Leung-Malik (LM) texture filter bank[i]:** In textured areas, using color or intensity may result in too many segments. LM texture filter bank, Figure 1, describes a pixel by the response of 48 filters. The filters measure existence of bars and edges in 6 orientations and 3 scales, existence of blobs in 8 scales, and also measure the smoothed intensity value in 4 scales. Due to the high dimensionality of the feature vector, using mean shift or graph cut may not be practical. So, for this case, only experiment with k-means. Note that to apply a filter to a color image, apply it to the three channels and keep the maximum response over the three channels for each pixel[ii].

- **Invariant LM (ILM) filters:** LM filters in the previous bullet are not invariant to rotation and scaling of texture patterns. To make the filter bank rotation and scale invariant, we can store the maximum response for each filter type over all orientations and scales. This means we apply the 48 filters, but, for each pixel, we keep the maximum response over all 18 bar filters, the maximum response over all 18 edge filters, the maximum response over the 8 blob filters, and the maximum response over the 4 smoothing filters. This makes only a 4D feature vector for each pixel. Because this is a compact texture descriptor (4 dimensions only), it can be used with all three segmentation algorithms.
- **PCA of combined features:** In this last experiment, you will use a longer feature vector including YUV colors and the 48 LM filters responses. The combined 51-dimensions feature vector is then reduced to 4-dimensions using Principal Component Analysis (PCA).

Project Folder

The project folder includes:

- The project description document that you are currently reading
- Sample test images: There are two groups of images. Apply k-means and mean shift on all image. Apply graph cut on the group under the 'single object' folder only. In this group, there is a single foreground object. You should apply graph cut to segment this foreground object from the background.
- LM Filters: The text file `LeunMalikFilterBank.txt` contains the kernels for the 48 filters. Each kernel is a 49×49 matrix stored in row major order. The order of the filters in the file are as the same as the images in the folder 'LMFilters'.
- References: There are few background papers under the 'references' folder that could be helpful.

Deliverables

You are required to deliver the following:

- Your code
- Output segmentation for the provided test images for each valid algorithm-features combination. For k-means and mean shift, your output should include an image created by replacing the colors of all pixels in the same segment by a single color, which could be the median, mode, or mean of pixel colors in the segment. For graph cut, create an image with the foreground object alone on a blank background.
- Report including explanation of your code, GUI usage, and representative results on sample test images along with your interpretation of the results. For example, you should provide reasoning on why a certain algorithm/feature combination outperforms another. You might not find a good reasoning for all the results you obtain. Just try your best!

Bonus

You may acquire up to 10% bonus if your implementation/report exceeds the requirements. You may consider one of the following bonus items:

- The group showing the best segmentation results over the entire class will be rewarded a bonus of 2% of the grade. Bonus will be distributed over tied groups.

- [2%] Applying and showing better results with other combinations of features. For example, consider different filter banks for texture analysis[iii].
- [3%] Applying another segmentation algorithm, such as Normalized Cuts[iv] or Random Walks[v]
- [5%] Using a different distance measure between points in k-means and graph cut. Particularly, consider representing each pixel by a histogram of filter responses in a circular neighborhood surrounding the pixel (a circular neighborhood can be implemented using a Gaussian-line weighting function for pixels in a rectangular neighborhood). Then, you can use the χ^2 distance between histograms.
- [3%] Quantitative evaluation of the algorithms based on the ground truth segmentation available with the Berkley Segmentation dataset

Policies

- Group size is 5 members at most. No extra credit for smaller groups.
- Late submission is accepted with 20% penalty for submission within one week after the deadline, and 50% penalty afterwards.

Useful resources

- The Berkley Segmentation dataset: http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/
- Python library for graph algorithms including max-flow: http://code.google.com/p/python-graph/
- Python library for mean shift segmentation that is compatible with OpenCV: http://code.google.com/p/pymeanshift/
- For k-means and PCA, you can use the implementations available in OpenCV

[i] http://www.cs.berkeley.edu/~malik/papers/LM-3dtexton.pdf
[ii] This is one of the ways to apply a filter to a color image.
[iii] Consider ones with Matlab (Octave-compatible) code in http://www.robots.ox.ac.uk/~vgg/research/texclass/filters.html
[iv] http://www.cs.berkeley.edu/~malik/papers/SM-ncut.pdf
[v] http://cns.bu.edu/~lgrady/grady2006random.pdf