

CS 216: Image Understanding
Fall 2008
Homework 3

Review: your class notes and Forsyth and Ponce Chapter 9

1. Color quantization: Although k-means clustering on color alone doesn't provide very satisfying segmentations, it can be used to reduce the color palette of an image. Write a script which takes a color image and a value for k and returns a new version of the image which uses only k distinct colors. Your code should cluster the pixel values using k-means and then replace each pixel with the mean color of the cluster to which it belongs. Demonstrate your code on a (colorful) image for different values of $k = 2, 5, 10$. What happens if you scale one of the feature coordinates, say the R value by a factor of 10?
2. Filterbank: In order to analyze texture, we would like to describe the image in terms of distribution of filter outputs. Implement a filter bank function which takes an image as input and filters the image with Gaussian derivatives in the horizontal and vertical directions at 3 different scales $\sigma = 1, 2, 4$. Also create an additional center surround filter by taking the different of two isotropic Gaussian functions at two different scales, i.e. $G_2(x, y) - G_1(x, y)$ and $G_4(x, y) - G_2(x, y)$. Feel free to use the various tricks we've discussed for making this process fast (e.g. separability). Your filterbank function should take as input one grayscale image and return 8 filter response images. Submit your code and an image which shows your 8 filter kernels.
3. Filter Distributions: Now convince yourself that the distribution of filter outputs really captures something about the texture. Use the image:
http://www.ics.uci.edu/~fowlkes/class/cs216/hwk/zebra_small.jpg
Select a image patch (say 40x40 pixels) over the zebra's neck and compute the mean absolute response of each of the 8 filter responses in that region ¹. Compute the mean absolute responses of the filterbank

¹If you don't take an absolute value then the positive and negative components will tend to cancel out and give you mean 0

channels for similar regions in the tree leaves above the zebra's back and also on the grass in front of the zebra. Print out the three mean response vectors for these regions and explain the differences you see in terms of your filters.

4. Textons: A nice way to represent the joint distribution of filter responses is to quantize them. Use your same k-means procedure from the first problem to quantize your filter bank output (now your feature vectors will be of dimension 8 instead of 3) using $k = 20$. Rather than output the cluster centers, create an image where each pixel is assigned its cluster label $1 \dots 20$. Submit the resulting texton image (use `colormap jet`).
5. Segmentation: Now that we have a good way to describe texture, lets try to segment the zebra image. For each pixel in the image, compute a histogram of the texton labels which appear in a 5×5 neighborhood around the patch. This will give a feature vector of length 20 associated with each pixel since there are 20 different types of textons. Now run k-means on these resulting feature vectors and display the resulting segmentation. We expect that pixels which all lie on the zebra's neck will have similar texton histograms and hence will end up in the same cluster. Compare the results of your clustering to simply doing k-means on the color values (as in problem 1). Submit your MATLAB code and your favorite segmentation result.
6. MRF: In this problem you will build an MRF model for segmenting an image into foreground/background and use an st-mincut solver to find the MAP assignment of pixels to foreground and background. You can download the code for the mincut solver from: http://www.ics.uci.edu/~dramanan/teaching/cs216_fall09/hw/mincut.zip. The file `demo.m` includes a demo application that uses the mincut solver to segment a noisy binary image.

Your code should display the test images (`segtest1.jpg` and `segtest2.jpg`) and prompt the user to click on two seed points (use `ginput` to get the mouse clicks). You should use these seed points to do two things. First, use the color of the pixels in the vicinity of the two points to initialize the foreground and background color models. Second, set the st connections so that these seed points are constrained to be in the foreground and background of the final solution (i.e. give them large weights)

- (a) Build a model in which the foreground and background are assumed to be constant color. Your st edges should have weights which are proportional to the distance in RGB between the pixel and your foreground or background color. Connections between neighbors should be proportional to the difference in color between neighboring pixels. There is a free parameter (λ in eqn 1 of Boykov & Jolly, or w_p in the online class notes) which controls the relative importance of the st edges and the neighbor edges in the graph. You will want to experiment to find a setting of this parameter that gives you a good segmentation result for the test image.
- (b) Modify your model to use your texture/color feature code developed above in order to build foreground and background models based on texture histograms. For each pixel, you can define the st weights by comparing the histogram in a window around the pixel to your foreground and background models. Keep the neighbor edge weights based on color differences as before.

You should submit your code for both parts along with example results of the segmentations produced from the test images <http://www.ics.uci.edu/~fowlkes/class/cs216/hwk/segtest1.jpg> and <http://www.ics.uci.edu/~fowlkes/class/cs216/hwk/segtest2.jpg> in the website hwk directory as well as any others of interest. Make sure and use the `plot` command to plot 'x's on the original image indicating where your seed points were for each example. Comment on the successes and failures of this approach.