Computer Vision
G22.2271-001
Assignment 4.

May 3, 2011

1 Introduction

This assignment asks you to implement the Normalized Cuts algorithm of Shi and Malik (2000). You should download both examples images from the course webpage and use these to show the results of your algorithm.

2 Normalized Cuts

Before attempting to implement the algorithm, download and read the Shi and Malik paper: "Normalized Cuts for Image Segmentation", from the course webpage. In the guidelines that follow, we will use the same notation as the paper.

The input to the algorithm should be a grayscale images of size M by N with intensities in the range 0 to 1. You should use `tiger.jpg` and `baseball.jpg` images on the course webpage.

The first stage involves computing the affinity matrix W between all pairs of pixels which will be a large MN by MN sparse matrix. The affinities between a pixel i and j are given by equation 11 in the paper. Note that there are two terms, one using the intensity difference between the pixels and the other using the distance between their locations. However, if the distance is greater than some threshold r, then the affinity is zero (which has the effect of keeping the matrix sparse). You should use the values $r = 5$, $\sigma_I^2 = 0.05^2$ and $\sigma_X^2 = 4^2$.

Since a non-sparse matrix of size MN by MN will not fit in memory, you should use Matlab's sparse matrix functions to represent it. Type `help sparse` to see how such a matrix can be generated from a list of row indices, column indices and sparse values. Also, using for loops to generate the matrix is likely to be slow. A better approach would be to use the `im2col` command to decompose the image into a series of patches, each of size $2r+1$ by $2r+1$ within with the two affinity terms are computed. Then you should build a list of row and column indices of the non-zero terms within each patch in terms of the original images coordinates. From this representation, it would be straightforward to generate the matrix W with the `sparse` command. To handle the edges correctly, it is probably easiest to pad the image with the `padarray` command before using the `im2col` command, and then ensure the affinities within the padded region are zero before building the sparse matrix. Some tips to debug your code: (i) use a small patch of the image to start with so that you can compute using simple methods the correct value of W; (ii) W is symmetric – check that this is the case; (iii) the diagonal should consist of ones. Please comment your code clearly so that I can figure out how you are computing W.

Once you have built the sparse matrix W, compute the diagonal matrix D consisting of the row-sum of W using the `spdiags` command.

Now, compute the matrix $A = D^{-1/2}(D - W)D^{-1/2}$ and solve for the smallest K eigenvectors using the command: $[v, d] = eigs(A, K,' SM')$;. For visualization purposes, we want to look at several of the eigenvectors, so set $K = 12$.

Plot out each eignevector, reshaping it into an image and plotting it with the commands `subplot` and `imagesc` and using `colormap(gray)` to make the images grayscale. For each of the test images, save the figure showing these eigenvectors.

Now, the second smallest eigenvector should approximate the optimal normalized cut solution. However, we want a binary segmentation of the image. Once solution is to threshold at zero, but a better one is to try various different binarization thresholds on the 2nd eigenvector and pick the one with the lowest NCUT(A,B). You should do the latter for a dozen or so different thresholds. See the top of section 2.1 in the Shi and Malik paper to see how to compute NCUT(A,B). You should print out the optimal value and plot the resulting segmentation.

Finally, for extra credit, split the image into more segments, using the code above to recursively split each of the two test images into 16 segments.

Plot the resulting image segmentation, using a different color for each segment.

You should turn in your code, along with figures showing the eigenvectors and segmentations for both test examples.