CSE166 – Image Processing – Homework #7
Instructor: Prof. Serge Belongie
`http://www-cse.ucsd.edu/classes/fa11/cse166-a`
Due (in class) 12:00pm Wed Nov. 30, 2011.

General Homework Guidelines

- Use the Cover Sheet provided.
- Please attach all code that you use. Attach code at end of submission.
- In general try to keep you answers concise. Use as many words as you need and no more. Also work on your presentation skills. This means organize your plots and displays. Always use titles and add captions when appropriate. *Points will be awarded for clarity and presentation.*

Reading

- GW Second or Third Edition 11.4.
- GW Second and Third Edition Review Material Ch. 1, "A Brief Review of Matrices and Vectors."

Written exercises

1. GW Second Edition, Problem 11.17.
   or
   GW Third Edition, Problem 11.20
2. GW Second Edition, Problem 11.18.
   or
   GW Third Edition, Problem 11.21
3. Consider the 2×2 matrix
$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}.$$
   Show that the inverse is given by
$$A^{-1} = \frac{1}{\det(A)} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}.$$

Matlab exercises

1. Lucas-Kanade optical flow.
   (a) Implement the Lucas-Kanade algorithm for measuring optical flow, as described in class. Allow the user to specify the size of the window used in enforcing the smoothness constraint. Use the `quiver` function to display the optical flow vectors. In addition, have your program return the two eigenvalues of the windowed image second moment matrix at each pixel.
   (b) Construct two frames of a simple motion sequence as follows. Make a 16×16 white square centered on a black background of size 32×32. Blur it with a Gaussian filter with $\sigma = 1$. This image represents $I(x, y, t)$. Produce the second image, representing $I(x, y, t+1)$, by displacing the first image down one pixel and to the right one pixel. Display each frame, as well as I_t and the two components of ∇I.

(c) Compute and display the optical flow for the above sequence using a window size of 5×5. Since you know the "ground truth" displacement (i.e. $u = 1, v = 1$), comment on the accuracy of your measured optical flow at various points throughout the image. Demonstrate how, by applying a threshold on the eigenvalues, you can suppress the flow vectors at pixels that suffer from the aperture problem.

(d) Construct a new sequence consisting of the original first frame and a second frame produced by rotating the first one by 5° (use `imrotate` with the `'bil'` and `'crop'` options). Now repeat step 1c using this sequence.

Things to turn in:

- Code listing for steps 1a, 1b, 1c, and 1d.
- Program output for steps 1b, 1c, and 1d.
- Written comments for steps 1c and 1d.

2. Clustering in color space using k-means.

(a) Implement the k-means clustering algorithm as described in class. For the stopping criterion, allow the user to supply a threshold on the change in J (the sum of squared error over all clusters) on each iteration, as well as a maximum allowed number of iterations (e.g. 25). Have your program output the value of J on each iteration. Initialize the cluster centers by choosing k feature vectors at random from the data.

(b) Load in Figure 6.30(a) (the bowl of strawberries) and use `imresize` with the `'bilinear'` option and `M=0.25` to reduce its resolution by a factor of 4 in x and y. Display the resized image. Construct a matrix of 3-dimensional feature vectors for this image using the RGB values of each pixel.

(c) Use k-means to perform clustering on this image using $k = 3$ and $k = 4$. In each case, run three trials to see the effects of the random initialization. Display each of the six resulting segmentations as a pseudocolor cluster membership image, like the example shown in class. (Note: if you don't have access to a color printer, it's ok to use shades of gray.)

Things to turn in:

- Code listing for parts 2a and 2b.
- Program output and parameter settings for part 2c.

3. Principal Components Analysis.

This exercise makes use of the face dataset on `http://isomap.stanford.edu/datasets.html`, consisting of a large set of images of a single face under varying pose and lighting conditions. The total number of faces in the dataset is 698. Each face is stored as a column vector of length 4096 and can be reshaped into a 64×64 grayscale image (for example use `imshow(reshape(data(:,10),[64 64]))` to display the 10th face). For purposes of this exercise, we will only use the first 100 faces, which you can download on the class webpage.

(a) Load this dataset into Matlab and display the first 12 images in a 3×4 subplot.

(b) Compute and display the mean face.

(c) Do PCA on the set of 100 faces, using the trick based on the eigenvectors of the small covariance matrix discussed in class. Make a plot of the eigenvalues sorted in descending order. Display the first 20 eigenfaces in a subplot, and title each image with the corresponding eigenvalue.

(d) Compute the reconstruction of face no. 1 based on the first 50, 75, and 90 principal components. How much of the variance is captured in each of these cases? Make a 2×2 subplot showing the original image followed by the three reconstructions.

(e) Repeat the previous step for face no. 50.

Things to turn in:

- Printouts of program output for steps 3a, 3b, 3c, 3d, 3e.
- Written answer for part 3d.
- Code listing for steps 3b, 3c, 3d.