Assignment 4: CS 763, Computer Vision

Due 2nd April before 11:55 pm

Remember the honor code while submitting this (and every other) assignment. You may discuss broad ideas with other student groups or ask me for any difficulties, but the code you implement and the answers you write must be from members of the group. We will adopt a zero-tolerance policy against any violation.

Submission instructions: You should ideally type out all the answers in Word (with the equation editor) or using Latex. In either case, prepare a pdf file. Put the pdf file and the code for the programming parts all in one zip file. The pdf file should contain instructions for running your code. Name the zip file as follows: A4-IdNumberOfFirstStudent-IdNumberOfSecondStudent.zip. If you are doing the assignment alone, the name of the zip file should be A4-IdNumber.zip. Late assignments will be assessed a penalty of 25% per day late. Please preserve a copy of all your work until the end of the semester.

1. Answer the following questions pertaining to the essential/fundamental matrix in a binocular stereo system:

   (a) Consider two cameras with parallel optical axes, with the optical center of the second camera at a location $(a, 0, 0)$ as measured in the first camera's coordinate frame. What is the essential matrix of this stereo system?

   (b) Suppose I gave you the fundamental matrix of a stereo system, how will you infer the left and right epipoles?

   (c) Prove that any essential matrix will have one singular value which is zero, and that its other two singular values are identical. Derive a relationship between these singular values and the extrinsic parameters of the stereo system (*i.e.*, the rotation matrix $\mathbf{R}$ and/or the translation vector $\mathbf{t}$ between the coordinate frames of the two cameras). [Hint: Show that if $\mathbf{E}$ is the essential matrix, then we can write $\mathbf{E}^T\mathbf{E} = \alpha(\mathbf{I}_{3\times 3} - \mathbf{t_u}\mathbf{t_u}^T)$ where α is some scalar which you should express in terms of $\mathbf{R}$ and/or $\mathbf{t}$, $\mathbf{I}_{3\times 3}$ is the identity matrix with 3 rows and 3 columns, and $\mathbf{t_u}$ is a vector of unit magnitude in the direction of $\mathbf{t}$].

   (d) In the noiseless case, what is the minimum number of corresponding pairs of points you must know in order to estimate the essential matrix? Or in other words, how many degrees of freedom does an essential matrix have? Justify your answer. (Think carefully).

   (e) We have studied the eight-point algorithm in class for estimating the essential/fundamental matrix. There exist algorithms that require only 7 pairs of corresponding points. In robust estimation, what main advantage will a 7-point algorithm have over the 8-point version?

   [1 + 1 + 5 + 1 + 2 = 10 points]

2. In class, we have seen (or will soon see) a method of normalizing matching points from the two given images, before computing the fundamental matrix or the homography transformation (you would have seen this normalization in the code for homography that was handed out in assignment 2). This normalization involves zero-centering each individual point-set followed by a rescaling. Express this normalization in the form of a matrix, and write out the expression for the final fundamental matrix. Explain why this normalization is useful. [5 points].

3. In this exercise, you will implement the Adaboost method for creating a strong binary classifier from a series of weaker classifiers. You will work with some synthetic datasets and also with a dataset containing images of digits.
Consider a training set consisting of N input vectors $\{\mathbf{x}_j\}_{j=1}^N$ in a d-dimensional space, and their respective labels $\{y_j\}_{j=1}^N$ where $\forall j, y_j \in \{-1, +1\}$. You will assign a scalar weight to each input vector. Before the first iteration of Adaboost, these weights will be set to be equal in value. In each round t, you will pick the best classifier from the following family of weak classifiers: $h_t(\mathbf{x}; i, p, \theta) = \text{sign}(p(x_i - \theta))$ where x_i is the i^{th} element of d-dimensional input vector $\mathbf{x}$, the parameter $p \in \{-1, +1\}$ and θ is a real-valued threshold parameter. Basically, this classifier assigns input vector $\mathbf{x}$ the label '+1' if either (1) $x_i > \theta$ and $p = +1$, or (2) $x_i \leq \theta$ and $p = -1$. Otherwise it assigns $\mathbf{x}$ the label '-1'. The best classifier refers to the classifier producing the least weighted error on the training set, i.e. least value of $\epsilon = \sum_{j=1}^N w_j I(h_t(\mathbf{x}_j) \neq y_j)$ where $I(.)$ is an indicator function that returns 1 if the predicate passed as a parameter is true, and returns 0 otherwise. Note that the search for the best classifiers involves picking the tuple (i, p, θ). After picking the best classifier, you will update the weights following the method in the standard Adaboost algorithm. This entire process is repeated for T rounds. You need to specify the value of T but $T = 30$ to 40 is sufficient. The final classifier after T rounds will have the form $H(\mathbf{x}) = \text{sign}(\sum_{t=1}^T \alpha_t h_t(\mathbf{x}))$.
You will work with the following datasets.

   (a) A dataset containing 2000 points in 2D drawn from a [0,1] bounded uniform random distribution. Label all the points lying on or inside a rectangle bounded by the lines $x = 0.3$, $x = 0.7$, $y = 0.3$, $y = 0.7$ as $+1$ and the rest as -1. Randomly divide this dataset into disjoint sets of 1000 training points and 1000 test points (called dataset1).

   (b) A dataset containing 2000 points in 2D drawn from a [0,1] bounded uniform random distribution. Label all the points satisfying any of the following conditions as '+1' and the rest as '-1': (1) lying on or inside a rectangle bounded by the lines $x = 0.3$, $x = 0.7$, $y = 0.3$, $y = 0.7$ as $+1$, (2) with x-coordinate between 0.15 and 0.25 or between 0.75 and 0.85, (3) with y-coordinate between 0.15 and 0.25 or between 0.75 and 0.85. Randomly divide this dataset into disjoint sets of 1000 training points and 1000 test points (called dataset2).

   (c) A dataset containing 2000 points in 2D drawn from a zero-mean Gaussian distribution of standard deviation 2. Label all the points whose distance from the origin is less than 2 as $+1$ and the rest as -1. Randomly divide this dataset into disjoint sets of 1000 training points and 1000 test points (called dataset3).

   (d) A dataset containing 2000 points in 2D drawn from a zero-mean Gaussian distribution of standard deviation 2. Label all the points whose distance from the origin is either less than 2, or between 2.5 and 3, as $+1$ and the rest as -1. Randomly divide this dataset into disjoint sets of 1000 training points and 1000 test points (called dataset4).

   (e) A database containing images of digits from 0 to 9 can be downloaded from `http://www.cse.iitb.ac.in/~ajitvr/CS763_Spring2015/HW4/` (in the form of 4 text files with self-explanatory names). The text files can be read into MATLAB using 'dlmread'. Each image has size 13×13, and there are 1000 images each in the training set as well as the test set. Label the images belonging to any one digit (say '2' or '3') as '+1' and all the others as '-1'. Thus for this database, your job is to determine whether a given image contains the selected digit or not.

   For each of the five datasets, do the following after each round of Adaboost: (1) estimate and print the training error of the strong classifier created thus far, (2) estimate and print the error of the strong classifier created thus far on the test set. For the first four datasets, also plot the test points with their associated labels using the MATLAB function called 'scatter' (in each round of Adaboost). Sample scatter plots for the first four datasets can be found at `http://www.cse.iitb.ac.in/~ajitvr/CS763_Spring2015/HW4/` (named as dataset1.jpg and so on). Finally, for all four datasets, plot a graph of the test set error versus the number of rounds of Adaboost. Do you notice something peculiar with the fourth dataset? Explain what you would you do to remedy that situation (there is no need to implement).

[15 points]