CS 485/685 – Computer Vision

Programming Assignment 2

(Due March 5)

1. (100 pts) Most face recognition algorithms require that human faces in an image are normalized prior to recognition. In general, normalization is performed with respect to location, size, orientation, and lighting. Here, you would need to design and implement a simple algorithm based on affine transformations.

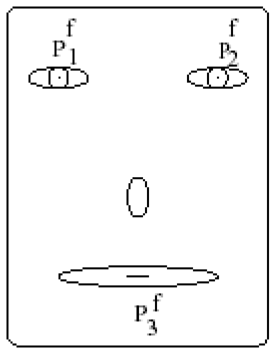

Figure 1. A typical face image showing three features of interest.

Specifically, the algorithm uses an affine transformation to map certain facial features to predetermined locations in a fixed window. Figure 1 shows an example of such facial features. Figure 2 shows examples of faces normalized with respect to location, size, and orientation.

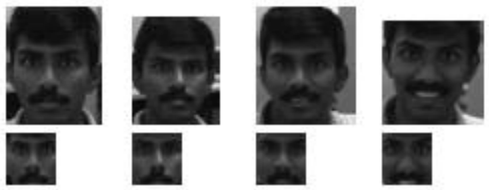

Figure 2. Examples showing face images before and after normalization.

A set of images to be used in your experiments is available from the class webpage. In this assignment, you will be using the following 4 facial features: left eye center, right eye center, nose tip and mouth center. First, you will need to extract manually the actual coordinates of these facial features from each face image provided. Use any image viewer that shows the image coordinates for the location currently under the mouse cursor. Then, you have to choose the predetermined locations of these features in a fixed size window, say 48x40 (note that the original images have size 112x92, so the aspect ratio is maintained) and compute the parameters of the affine transformation.

Every feature point needs to be mapped to its predetermined location in the fixed window, thus it needs to satisfy the affine transformation equations. The example below considers only 3 features, although you will use 4. Denote the actual eye and mouth locations with (P_1, P_2, P_3) and their predetermined (fixed) locations with (P_1^f, P_2^f, P_3^f). The affine transformation equations are given below:

$$P_1^f = AP_1 + b$$
$$P_2^f = AP_2 + b$$
$$P_3^f = AP_3 + b$$

where:

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \text{and } b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

The above equations can be rewritten as:

$$Pc_1 = p_x$$
$$Pc_2 = p_y$$

where:

$$P = \begin{bmatrix} X_1 & Y_1 & 1 \\ X_2 & Y_2 & 1 \\ X_3 & Y_3 & 1 \end{bmatrix}$$

$$p_x = \begin{bmatrix} X_1^f \\ X_2^f \\ X_3^f \end{bmatrix} \qquad p_y = \begin{bmatrix} Y_1^f \\ Y_2^f \\ Y_3^f \end{bmatrix}$$

$$c_1 = \begin{bmatrix} a_{11} \\ a_{12} \\ b_1 \end{bmatrix} \qquad c_2 = \begin{bmatrix} a_{21} \\ a_{22} \\ b_2 \end{bmatrix}$$

Each facial feature from each image contributes a pair of equations (one for the *x* coordinates and one for the *y* coordinates). By putting together all the equations for that image we obtain an over-determined set of linear equations that can be solved using Singular Value Decomposition (SVD), in order to determine the parameters of the affine transformation. The code for solving over-determined systems of equations is available on the class webpage.

Once you have solved for the parameters of the affine transformation for each image, you have to normalize each image using its computed affine transformation.

Furthermore, verify how well the computed affine transformation aligns the actual facial features with the fixed ones – compute and report the average error in pixels, for each image separately and for the entire set of images.

Submit the following:

- (hardcopy) A report describing your results; it must include the recovered parameters for the affine transformations, the normalized images, the average error in pixels, for each image separately and for the entire set of images, a discussion of results and comparisons; for all figures provide captions with a brief description.
- (Canvas) One ZIP file “PA2_pb1.zip” containing:
    - The report file “PA2_pb1_report.pdf” mentioned above.
    - The source code files in C/C++.
    - A “README.txt” file with instructions on how to compile and run the program and any parameters that need to be set.

2. (10 pts – mandatory for graduate students, optional as extra credit for undergraduate students)

   Once a face image has been normalized with respect to position, scale, and orientation, some lighting correction can be applied to account for non-uniform illumination. First, a linear (or higher order) model is fit to the intensity of the image; for example, the following model can be used:

$$f(x,y) = ax + by + cxy + d$$

Each pixel (x,y) in the input image must satisfy the above equation where f(x,y) is the intensity value of the input image at location (x,y). Using SVD, we can find the "best" coefficients a, b, c, and d (in a "least-squares" sense). For an N x M image, this yields NM equations with four unknowns (an over-determined system).
Figure 3 provides an example; the first row shows some input images while the second row shows the linear model fit to each of these images. To correct for lighting, simply subtract the linear model from the original image. The last row of Figure 3 shows the results. Your task is to apply such lighting normalization on the normalized face images obtained in the previous problem.

Submit the following:

- (hardcopy) A report describing your results; it must include the recovered parameters for the models fit, the lighting normalized images, a discussion of results and comparisons; for all figures provide captions with a brief description
- (Canvas) One ZIP file “PA2_pb2.zip” containing:
    - The report file “PA2_pb2_report.pdf” mentioned above.
    - The source code files in C/C++.
    - A “README.txt” file with instructions on how to compile and run the program and any parameters that need to be set.

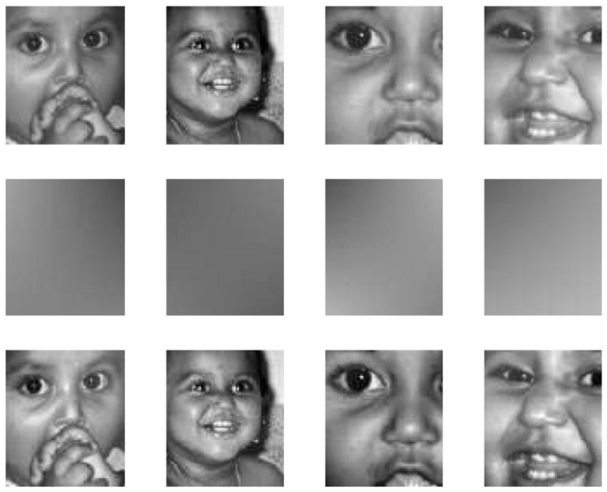

Figure 3. Original images (1st row), model fit (2nd row), lighting normalized images (3rd row).