The University of Calgary
Department of Electrical and Computer Engineering

ENCM 509 - Fundamentals of Biometric Systems Design

Laboratory Experiment # 6

Part I: Face recognition. Part II: 3D face modeling

1 Introduction

The purpose of this laboratory exercise is to investigate a basic face recognition approach such as "eigenfaces". Using 1:1 matching , we will design a procedure or 1:M comparison, or identification.

The second purpose of this lab is to explore biometric data modeling (synthetic biometrics) and its use. For example, modeling faces can help facial recognition.

Facial recognition algorithm used in this lab, is implemented in the script FaceRec.m located on N: drive *\ENCM\509\lab6*.

The input data are bmp or jpg images. Some sample (synthetic) images named 1.bmp .... 9.bmp are located in the same directory (lab6). Copies of images are also given in `SyntheticFaces` . An example of a database of 16 images (8 for one person, and 8 for another) is given below.

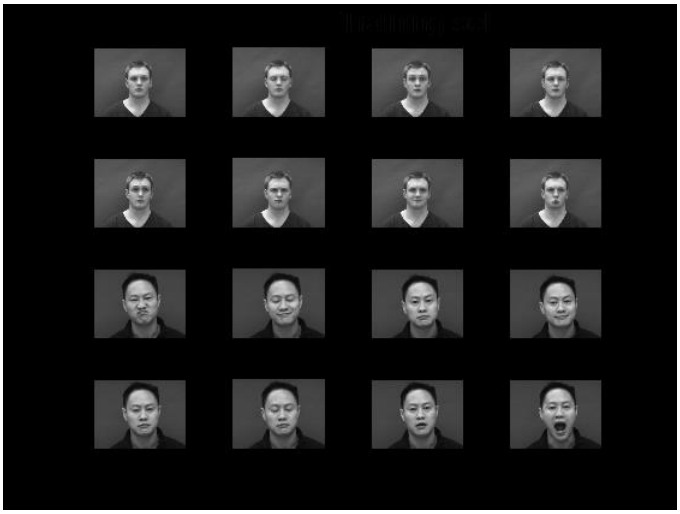

Figure 1: Example of a database.

It is important to have the images normalized: their sizes must be equal (in the number of pixels), otherwise, the program will report errors. You can use any graphical editor, for example, Paint, to do so.

The program finds the mean face, and performs calculations of differences between the mean image and every image, finding a covariance matrix and so-called "eigenvectors".

The algorithm have a "training" step in order to create eigenfaces for images in the database. After the "training", the recognition can be done. The feature vector called "eigenvector" of the probe image is calculated, and a difference (Euclidean distance) between that and the feature vectors of the database images is found. An example is shown below:

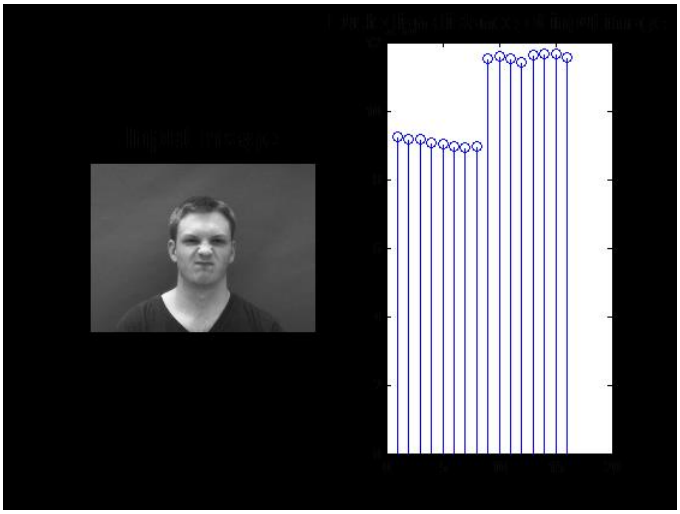

Figure 2: Face recognition (Euclidean distances of a probe face to 8 faces of the same person and 8 faces of another)

The last step is analyzing this difference and making a decision about the match (which one of the database facial image matches the probe). This is $1 : N$ comparison, or identification, by performing 1:1 matching N times.

The main criterion for matching of a probe face (known to be a face of a person in the database) to the database face or faces is that the Euclidean distance for the same person's face feature vectors must be minimal.

The second criterion of a good face recognition algorithm is that it has to distinguish between a face and a non-face object: the input face belongs to a class, if the Euclidean distance is below some established threshold. Then, the face image is considered to be of the known face. If the difference is above the found threshold, but below a second threshold, the image can be determined as an unknown face. If the distance is above these thresholds, the image is not a face.

Note that for large databases, a ranking model of matching can be applied, and the top few minimal distances are ranked as possible candidates.

2 Exercise 1: Face recognition

Find file FaceRec.m on N: drive $\backslash ENCM \backslash 509 \backslash lab6$.

You can run this demo program using sample images. The sample data are synthetic images named 1.bmp .... 9.bmp located in $\backslash ENCM \backslash 509 \backslash lab6$.

When you run the file FaceRec.m, it will ask you to input a file for comparison. When you choose a file (you can choose a sample file for demo, such as `1.bmp`), it will show you the face and the plot of Euclidean distances between this face feature vector, and the feature vectors of other images in the training set. The smallest distance indicate the closest similarity. The Matlab will also type the smallest and largest Euclidean distance. You will need those to detect the best threshold in the later exercises in this lab.

The default path of your input image is lab directory on N drive (you can change it by changing line 172 in Facerec.m file). The number of images, M, is set up in the code (it is equal to 9, you can change it to 8 and use the remaining face for testing; this number shall be large for larger databases).

It is important to have the images normalized: their sizes must be equal (in number of pixels), otherwise, the program will report errors. You can use any graphical editor, for example, Paint, to do so.

For this exercise,

- Use your own images and images of 2 or more other people in order to create a small database (for example, 8 images of each person). Remember, image must be normalized (has the same size in term of pixels) prior to processing. You will use the file FaceRec.m on N: drive $\backslash ENCM \backslash 509 \backslash lab6$ to match pairs of images of the same person known to be in the database, pairs of images of different people that are known to be in the database, and an unknown person (not in database). Use several selected photos from the database for training. For example, select 7 images for training, and keep the remaining ones a "'probes"' for testing. After the "training", the recognition can be done. A probe image must have the same size as the training images. The default path of your input image is lab directory on N drive (you can change it by changing line 172 in Facerec.m file).

  FaceRecExample.m performs calculation of Euclidean distance for 1:1 matching. Design a procedure for $1 : N$ comparison, or identification, by performing 1:1 matching N times.

- Extend the script, so that it can record the thresholds and provide the decision output.

The two thresholds described in the Introduction must be found. To find the first threshold, all the maximum and minimum Euclidean distances given the input face that belongs to the database, must be gathered. For example, given 9 faces of the same person in the database, set one aside as a probe, train the algorithm on the remaining 8 faces, and then and run FaceRec.m 8 times, comparing each of the 8 faces against the probe face; record the maximum and the minimum Euclidean distances resulting from these matches. Repeat this 9 times (it is a 9-fold training). Create a table containing min and max distances for images of the same person in your database.

Design the procedure to perform the above for each person in your database. You can calculate the mean and standard deviation of the min and max distance. Thus, the first threshold can be set at one of those means (max or mean distances, or their boundary varies (defined by the deviation). Investigate which one shall be chosen.

To find the second threshold, use unknown (someone's else) faces as the input faces and gather all the max and min Euclidean distance. Create a table containing the min and max of the images not in database. Investigate which value (mean of the min distance, or max distance, or one of their boundary values) shall be chosen as the second threshold.

- Perform facial recognition using the steps above, and find the true and false match rates and rejection rates. Use the top matching face (regular matching, also called rank-1).
- Design a ranking model of matching using the top two or three matches (rank-2 or rank-3 approach) of the closest matches. Analyze scores and
- Draw conclusions and make suggestions on the procedure improvement.

3 Face modeling

The term "face synthesis" indicates a class of problems aimed at synthesizing a model of facial appearance. The generation of faces and facial expressions is the manipulation of topological primitives with special rules. Physical modeling is based on descriptions of facial muscles, skin, and their relationships by mathematical equations. Given a muscle model and emotions, the latter can be encoded or incorporated to produce a synthetic face. It allows for creating databases of facial images to be used for testing the facial recognition algorithms and systems.

Collecting images for face recognition and testing the face recognition algorithm is a time-consuming problem: it is lengthy and privacy-issue related procedure. In addition, it is hard or impossible to imitate real-life noises or distortions such as

backgrounds, facial accessories and head rotation. In this case, so-called analysis-through-synthesis can be applied, where the testing data can be created using a model of the face.

In this lab, we will use a package for 3D facial modeling, called FaceGen from Singular Inversions. The 3D model used in the package is a combination of 3D mesh model of a human head, and a texture model of the skin. A face appearance can be morphed with respect to various parameters: expression morphing (smile closed and open, anger, fear, disgust, sad, surprise), age morphing (shape and texture modifications), race and gender morphing, asymmetry etc.Version 3.3 of this package is used in this exercise and is available on `N:\ENCM\509\Singular Inversions\FaceGen Modeller 3.3`.

One of the functions of this package, called PhotoFit, can create a 3D model of a head given one still frontal image, or, better yet, two more profile images. You can use your pictures taken prior to the lab, and stored on `N:\ENCM\509\`.

4 Exercise 2: Face modeling and recognition using synthetic faces

4.1 Creating your face model

- Open FaceGenModeller 3.3 from C: drive and run the application. The application displays some average 3D face.
- Choose the most right panel "PhotoFit" and click on the bottom key "Next".
- In the new window, choose "Load" for Frontal Image, and browse to the directory where your face is stored. Choose one neutral facial expression, and possibly front and/or rotated images.
- Proceed to the next prompted step "Start Now" (you may also check mark the option "Preserve facial hair in detailed picture" if not marked yet). It may take several minutes to create the model.
- You can change the background color (for example, to green) in "View" and "Background Color".
- To model hair, you can choose "Texture overlay", the panel under the picture, and in the new window move "Short black hair" from "Available overlays" to "Current overlays" using the button `>>>`.
- Save the image with extension `.fg` (to be able to open it again in FaceGen) using "File" then "Save", as well as with extension `.bmp` using "File" then "Save as" (to be able to use for further steps such as face recognition).

Thus, you save the primary face model (using green background) with neutral facial expression.

4.2 3D model manipulation

Explore some option in the FaceGen menu for your neutral facial expression (you can load the saved file `.fg` of your 3D face model from Exercise 1):

- Choose panel "Generate", - you can change age (20 – 60 years old), gender, race etc.
- Choose panel "View", - you can change background color and light options (Light Source) etc.
- Choose panel "Camera", - you can change zoom and distance ratio
- Choose panel "Shape", - you can change some of the facial shapes expressions
- Choose panel "Texture", - you can add some facial accessories (though the colors may not be quire realistic)
- Choose panel "Genetic", - you can generate 8 random faces similar to yours
- Choose panel "Morph", - you can change some of the facial expressions.
- In addition, you are able to rotate the 3D view using the mouse while pressing the left button and holding it on the face image.

Save the following results:

- Among these options, use some, for example, "Shape" and "Morph" to change your neutral facial expression as close as possible to your chosen expression (for example, "Mouth open"). This will be a "Synthetic expression". Save 3 different face expressions; they shall correspond to any of the 3 expressions of your face taken prior to the lab. We will compare the results of face recognition for both real and synthetic face later.
- Use the rotation and rotate face $15^0, 30^0, 45^0$ (approximately) degree and save images as `.fg` and `.bmp` files., - we will use them to verify if the algorithm is still be able to recognize the expression on the rotated face (head rotation is the real problem in the automatic face recognition).
- Use the options from "View" to change background to some pattern; save images as `.fg` and `.bmp` files.
- Use the options from "View" to change lighting, for example, light source coming from the left or right or top; save images as `.fg` and `.bmp` files.

4.3 Testing the facial recognition algorithm

In real video-recording or frame recoding, the person can arbitrary move the head, and the facial recognition algorithm must be robust enough to recognize facial features for certain degree of such a "distortion". Thus, 3D face modeling can be used to generate the test sets for testing or verification of the recognition algorithms.

Use your 3D face model, created in Exercise 2.

The sample (synthetic) images named 1.bmp .... 9.bmp are located in the lab directory, and the number of images, M, is set up in the code as 9. Choose your own number of images.

- Use FaceGen 3.3. to generate synthetic faces rotated by $15^0, 30^0, 45^0$ (approximately) and/or more different degrees. Perform face recognition and record the results.
- Repeat the same actions for the face model with morphing (to create various expression).
- Repeat the same actions for the face model with aging (now, at 30,40,..,60 years old etc.) and record the calculated differences.
- Repeat the same actions for the neutral face using 4 various backgrounds, and record the calculated differences.
- Repeat the same actions for the neutral face using 4 different lighting conditions, and record the calculated action units.
- Evaluate the true and false match and reject rates.
- Compare the algorithm performance for both real and synthetic images. Draw conclusions and make suggestions on the procedure improvement.

5 Laboratory Report

The total number of marks is 10:

- Report on Exercise 1, with illustrations where appropriate, (4 marks).
- Script of the modified code for the identification procedure, with thresholds determined (2 marks).
- Report on Exercise 2, with illustrations where appropriate (4 marks).

6 Acknowledgments

The FaceGen Software is the product of Singular Inversion Inc., that is under the educational license in the ECE department.

Svetlana Yanushkevich
February 25, 2019