Programming Assignment

Due at 11:59:59pm, Friday, Dec. 4th

All the codes should be written in C or C++ for Linux and commented appropriately for major steps/functions

The codes should be submitted through department dropbox

Without special permission, late submission is not accepted.

1. Background and Introduction

In this project, you will be asked to implement a system to recognize human facial action units [1] from facial images. Facial behavior is the most powerful and natural means of human communication. A combination of facial actions will describe facial behaviors. If the computer can recognize facial actions from images automatically, it can identify our emotion and intention. There are many emerging application areas such as interactive games, intelligent transportation, and deception detection.

Here is a list of facial actions you need to recognize:

Figure 1. Exemplary images for three target facial action units (pictures adapted from [2])

What they are used for? We use them to recognize some basic expressions.

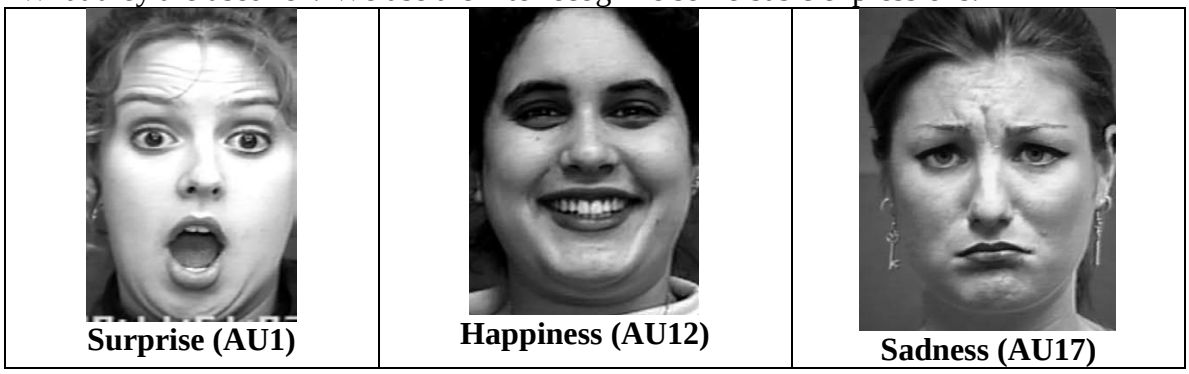

Figure 2. Exemplary images for three facial expressions involving the three target facial action units (pictures adapted from [3])

2. Methodology for Recognizing AUs

We will recognize the AUs using a K-nearest neighbor-based method as illustrated below:

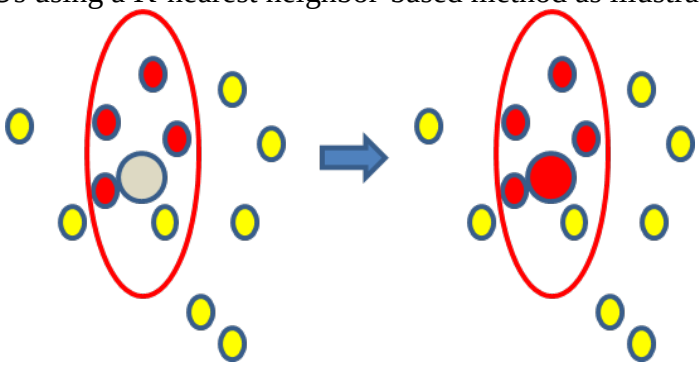

Figure 3. An illustration of K-nearest neighbor classification

The color of the big gray circle is determined by the voting results of its 5 nearest neighbors. Since 4 neighbors have red colors, it will be painted by red color as well.

In this project, each image is represented by a predefined feature set (a vector containing 5,632 elements) and denoted by $\vec{\mathbf{x}}$. Given a template image dataset $\mathbf{Y} = \{\vec{y}_1, \cdots, \vec{y}_j, \vec{y}_{N-1}, \vec{y}_N\}$ (containing N=138 images in this project), you need to compare the similarity between $\vec{\mathbf{x}}$ and each of the template images $\vec{y}_j$ and find the 10-nearest neighbors of $\vec{\mathbf{x}}$. The similarity function is defined as follows

$$s_j = \frac{\vec{\mathbf{x}}^T \vec{\mathbf{y}}_j}{\|\vec{\mathbf{x}}\| \|\vec{\mathbf{y}}_j\|} = \frac{\sum_{k=1}^{5,632} x_k y_{j,k}}{\sqrt{\sum_{k=1}^{5,632} x_k x_k} * \sqrt{\sum_{k=1}^{5,632} y_{j,k} y_{j,k}}}$$

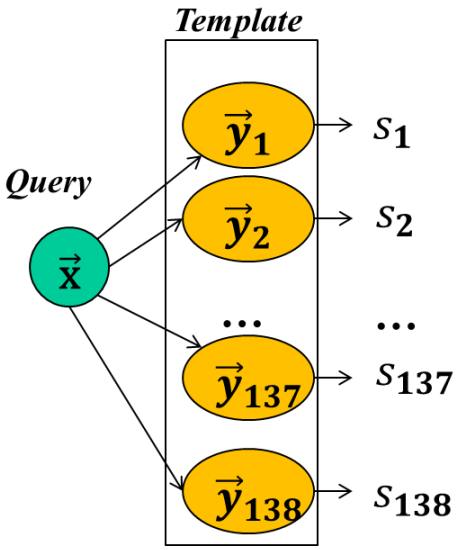

Figure 4. An illustration of computing similarity values between the query image and the template image.

By using this similarity function, the most similar template image (the closest neighbor of $\vec{\mathbf{x}}$) has the maximum similarity value.

3. Data File Provided (adapted from [3])

- **Query data**: You will be given 141 query data files named following the format "###_AU**_query.dat", where "###" represents the subject ID from 001 to 047; and "**" represents the AU label "01, 12, and 17", respectively. Each data file is an ASCII file containing a query $\vec{\mathbf{x}}$.

- **Template data**: For each subject, you will be given one data file named following the format "###_template.dat", where "###" represents the subject ID from 001 to 047. Each data file is an ASCII file each row of which contains a template image $\vec{y}_j$.

4. Project Task

You need to write codes in C or C++ to implement a software to recognize AU1, AU12, and AU17 using the provided data.

- For each query data, you need to find its **10-nearest neighbors** from the 138 template data with the same subject ID. For example, for "001_AU01_query.dat", you should look for its 10-nearest neighbors in "001_template.dat".
- For each query data, you need to output the row indices of the 10 nearest-neighbors in an order from the closest one (the one with highest similarity value) to the 10^{th} closest one, shown in both your program output and your written report.
- You need to report the execution time shown in both your program output and your written report.

NOTES:

- You are encouraged to form a team. Each team can have at most 2 members. You only need to submit one combined report for the team, but you need to clearly state the contributions of each member in the report.
- You are encouraged to make all possible improvements to achieve an efficient implementation and ensure the correctness of the solution.
- You can use the similarity function provided or define your own. But if you choose to use your own similarity function, the results (the nearest neighbors returned) should be exactly the same as the one resulted from the original similarity function; otherwise, we will consider your solution is NOT correct.

5. Requirements

You should submit a single zipped file through departmental electronic dropbox including

- A written report including
  - An introduction of the problem
  - The methodology you use to solve the problem including the strategy you propose to improve efficiency
  - a pseudo code for your algorithm
  - experimental results: the solution of the problem and execution time
  - conclusion
- Codes written in **C or C++ in Linux** (correct, clearly commented, complete, etc)

- A script file or readme file should be submitted together with the codes to explain how to compile your code. It is your responsibility to ensure your code can be compiled successfully
- If you are using some specific toolboxes, make sure you include them

Without the special permission from the instructor, **NO late** submission will be accepted

6. Performance Evaluation and Competition

The performance is evaluated based on

- the correctness of the solution and
- the time efficiency

We will have a competition on the time efficiency:

- Qualification: the solution should be correct
- 30% bonus credits (30%*11% of your final grade) will be given to the teams who rank the top 15% among all teams in terms of time efficiency (using the least time)
- 15% bonus credits (15%*11% of your final grade) will be given to the teams who rank the top 16%-50% among all teams in terms of time efficiency

Reference

[1] P. Ekman and W.V. Friesen, Facial Action Coding System: A Technique for the Measurement of Facial Movement. Consulting Psychologists Press, 1978.
[2] Y. Zhang and Q. Ji, "Active and Dynamic Information Fusion for Facial Expression Understanding from Image Sequences," IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 27, no. 5, pp. 699-714, May 2005.
[3] T. Kanade, J.F. Cohn, and Y. Tian, "Comprehensive Database for Facial Expression Analysis," Proc. Fourth IEEE Int'l Conf. Automatic Face and Gesture Recognition, pp. 46-53, 2000.