



VIDEO ATTENDANCE MODULE

CS 101

AUTUMN 2014

Software Requirement Specification

Keshav Srinivasan
Kshitij Bajaj
Kumar Ayush
Reebhu Bhattacharya

November 24, 2014

Contents

1	Introduction	3
2	Objective	3
3	System Overview	3
4	System Attributes and Requirements	4
4.1	Hardware Specifications	4
4.2	Software Specifications	5
4.3	Supplementary Requirements	5
4.3.1	Usability	5
4.3.2	Reliability	5
4.3.3	Performance	6
4.3.4	Supportability	6
5	Included Functions and Header files	6
6	Constraints and Dependencies	6
7	References	7
8	Possible Beta Extensions	8

1 Introduction

We live in a world, where minimizing of human effort is the core area of research in most fields; Machines were after all made for this purpose.

The first semi-automated facial recognition software was made for the FBI by Woody Bledsoe, Helen Chan Wolf and Charles Bisson in the 1960s which required the to identify the facial features such as Nose, eyes, etc. Since then, these softwares have been continuously devopling and improving themselves to create a streamlined and near-perfect facial recognition software.

In 1991, Turk and Pentland discovered the Eigenface technique which improved detection to a healthy 90 percent accuracy. By 2002, the advent of the Face Recognition Vendor Tests (FRVT) reduced the error to a mere 1 percent.

2 Objective

We aim to make an integrated video attendance package which involves the use Facial Recognition software. Its main purpose is to reduce human effort and to streamline the process of attendance by minimizing error and also eliminating the chance of a proxy attendance. This program hopes to revolutionize the system of attendance by creating a self-sustained facial recognition platform. The program is also extremely user friendly and can be used practically anyone due to its simple GUI(Graphical User Interface)

3 System Overview

As stated earlier,the main purpose of out program is facial recognition. The program includes functions to separate faces from a crowd to give rise to multiple-face recognition in a single frame. This proves to be a powerful feature for large number of students in a single classroom.

The complete system is built on OpenCV (Open Source Computer Vision) for basic image-processing functions which is integrated with a C++ platform (Microsoft Visual Studio)

The program has a central network connectivity to a database for storage and comparison.

We also have a GUI (Graphical User Interface) for smooth functioning of the program and also make the program more friendly with buttons and various options to interact with the system. We also aim to create a smart-system which which continuously updates itself to account for evolution of features

and variation of faces with time.

This software is made mainly for an Intel powered 32 bit system, but this being an overview, further hardware and software specifications are described in detail further in the document.

Lastly, we have thought of various Beta Extensions which might be implemented in further releases: Blink-trigger Attendance and Sound Processing

4 System Attributes and Requirements

The whole program relies on complete integration of the hardware and software involved. Hence it is paramount to have appropriate hardware interface for the optimum running of the program.

4.1 Hardware Specifications

The Hardware Specifications are listed below:

- Obviously, the program requires the use of an HD (High Definition) camera. The better the quality of image, the more accurate the face recognition, but this also creates problem in data storage of such a large number of image. Hence the quality of image is a compromise between accuracy and feasibility.
- A minimum of 2GHz Processor and 2GB RAM (Random Access Memory). But considering the standard of computers that are manufactured in the present day, this should not be a hurdle.
- A very large database with near-infinite memory storage is required to accommodate images of all test data and other required images.
- Communication interface, like High-Speed LAN (Local Area Network) must be in place so that the terminal can effectively communicate with database and vice-versa.
- Since the program being described requires high performance processors, a Multi-core Processor is ideally, with NVIDIA on board. The program will then use the GPU (Graphics Processing Unit) instead of CPU (Central Processing Unit)

4.2 Software Specifications

The software requirements for our program is not too much, but it still includes some essential components:

- Most importantly, we need OpenCV libraries and DLL (Dynamic Linked Libraries) to be present on the host system for the function of the program
- Requires Windows x64 (7 or above)
- Windows .SDK and .NET Frameworks (Version 4.5 and above)
- The System must also have YAML to access the .yaml files
- The System is based on C++/CLI system and hence requires basic Windows Libraries for MDF and system .dll files.
- The program will use an additional software known as CUDA by NVIDIA to effectively use parallel processing, hence the Processor is more effectively used and reduces program processing time.

4.3 Supplementary Requirements

The system has certain requirements that are not functional requirements but still lend to the quality of the product.

4.3.1 Usability

The system should have a well-defined interface to interact with the user. The user's primary interaction is with the Software. The client software on the system is responsible for taking photos and communicating to the server on the host PC. The GUI is easy to use. It has options such as the ability to change number of photos and ability to communicate with the host using a standard protocol.

4.3.2 Reliability

The client and server software must be reliable. The server software, especially, must be reliable enough to be always-on with maximum time between failures. The communication between the client and the server should follow a standard reliable protocol such as TCP/IP. The hardware (Wi-Fi radio) must also be reliable and secure. The Wi-Fi protocol must follow WEP (Wired Equivalent Privacy) so that the transmission is not compromised.

The reliability of the results of photo identification is wholly dependent on the reliability of the whole system.

4.3.3 Performance

A system transaction involves capturing of a photo of the subject, transmission of the photo over a wireless protocol to a server, the identification of the photo by the server using a database, and the transmission of the results to the client. This transaction must be fast, in the order of a few seconds. This requires a fast terminal, and a fast host (server) system. The algorithm used for identification is already developed and its optimization is beyond the scope of this project.

4.3.4 Supportability

The system must be supportable. Using off-the-shelf standard products and standard development environments reduce the risk of incompatibilities. This makes the product more supportable. A clean interface between the user, the client software and the server makes it easy to use, and supportable across multiple use cases.

5 Included Functions and Header files

Our program has an extensive list of functions and header files which all be described in the Project documentation. But a brief overview includes:

- FisherFaces in OpenCV Libraries (opencv2/core/highgui.h)
- Haar Cascade Classifiers for Frontal faces (haar-cascade.xml)
- Windows Form application (System::Windows::Forms)

These will all be documented in detail along with various other functions.

6 Constraints and Dependencies

Obviously, This program is in its alpha stage and needs to be improved multiple times before it can implemented on a large scale. But, we must also recognize certain shortcomings of the program which can be possibly addressed and corrected in the future.

- If Low-Resolution images are fed into the system, the system might turn unreliable.
- There is theoretical limit on the Face-recognition algorithm which is inevitable on any system of this sort.
- The lighting of the environment greatly affects the image quality and hence must be ideally lit with not much variation in diurnal light intensity, else images may seem different in the day and night, due to different lighting of the same face.
- Many induced errors like haircuts or shaves can possibly affect the image as appearance of a person can change drastically sometimes.
- Background Disturbances.
- Clearly, the program has various hardware and software requirements and completely relies on them. Hence due to the dependencies and due to the complex nature of the architecture of this program, it is very difficult to convert the given program to embedded system.

7 References

- <http://docs.opencv.org/doc/tutorials/tutorials.html>
- <http://opencv-srf.blogspot.in/>
- http://docs.opencv.org/trunk/doc/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html
- http://en.wikipedia.org/wiki/Haar-like_features.
- Enhanced Computer Vision with Microsoft Kinect Sensor: A Review, by J.Han,L.Shao,D.Xu and J.Shotton

8 Possible Beta Extensions

Our Program has a lot to offer, and some of the possible applications are mind boggling. Firstly, we are trying to make our system 'Un-Supervised' , which means, the program will run without any human interference and also collect its own sample data. Another extension of the program is that we can use a 'Blink-Trigger' system where the attendance system is triggered by the blink of an eye. This removes the of Proxy attendance via means of a poster, etcetera. Yet another module that we discussed was a voice-recognition system, which coupled with our face recognition system, can eliminate error further.