Attendance and Security Automation System

by
Anindya Kumar Verma

CONTENTS

Abstract

It's NOT a simple Attendance Automation System but actually a Complete Security System which can be used as an Attendance Automation System.

- ➡ We won't just be telling whether the student is present but also whether he or she attend the whole class or not and where they went after or before every class and for how long.
- ➡ A complete security system that can be used to find the movement and whereabouts of people in real time without having to look through the whole security feed.
- ➡ Image Processing is a fast growing field with many applications and face recognition is the very first and basic application.
- ➡ We are using face recognition to find the identity of all the students present in any particular area at that time.
- ➡ We will also provide RFID Tags which will be added to students ID cards and they will help the system to know who is present in the area and tell camera who to detect and who not to.

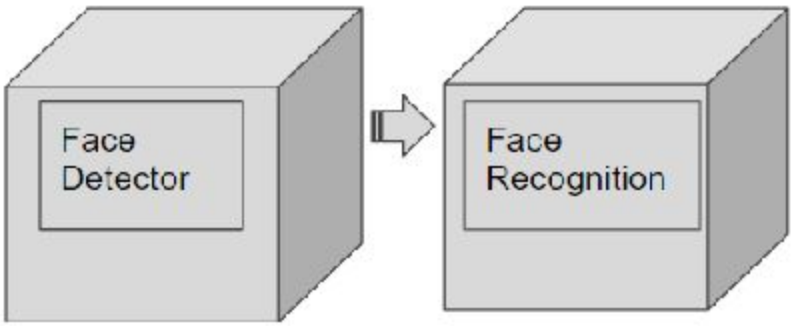
1. Introduction

We will use cameras to get the feed of the class and the collage which we will use to extract faces at regular interval and perform recognition and hence taking the attendance of the class .

- ➔ By getting continuous feed and checking it at regular intervals we will get accurate attendance of the class for the whole period.
- ➔ RFID Tags which will help the system to know who is present in the area and tell camera who to detect and who not to and hence we can find who has their ID card also and no one will be able to use proxy because the camera is using face recognition as well.
- ➔ If a student leaves a class in the middle without telling the teacher we will be able to detect it and report it to the teacher immediately and automatically .
- ➔ We can use the collected data to find out the behaviour of students and their activities in the class and perform analysis on the data achieved automatically.
- ➔ No human needed for the whole process.

2. Objective

To detect multiple faces and implement face detection and recognition which will associate a name to each face. RFID Tags which will help the system to know who is present in the area and tell camera who to detect and who not to .



What we are basicly doing

To do so we will need a database ,cameras and a processing unit.

Create a database of all the students data which will have:

- ➔ Students basic information(name ,address ,roll no ,department etc)
- ➔ Students Photo (front and side view)
- ➔ Students RFID

Once the database is created we will perform face detection and recognition inside the college campus in various areas (Class , Labs etc)

When a class will start the camera will start recording and will take a photo at intervals of 3 minutes and perform face recognition and find out who is in class and who is not ,Store that data and at the end of the class update the attendance register.

For tracking purpose once we know a person has entered the college we will start following the student through cameras by telling each camera who that camera has to expect or who will cross that cameras location(we will get this information through their ID cards) , by repeating this process we will be able to form a path or log of activities of each student.

We will need to put a ID card checking system at the college door which will check whether the person entering has a valid ID or not so that our system works properly this will also help us stop any unauthorised entry in the college and hence make the campus more secure.

3. Technology

Various Technologies used are :

➔ **Python**

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together.

Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

➔ **OpenCV**

OpenCV (Open Source Computer Vision) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage and is now maintained by Itseez.The library is cross-platform and free for use under the open-source BSD license.
OpenCV supports the Deep Learning frameworks TensorFlow, Torch/PyTorch and Caffe.

➔ **Pandas**

Pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language.

Pandas provide fast, flexible, and expressive data structures designed to make working with structured (tabular, multidimensional, potentially heterogeneous) and time series data both easy and intuitive.

It aims to be the fundamental high-level building block for doing practical,

real world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis / manipulation tool available in any language.

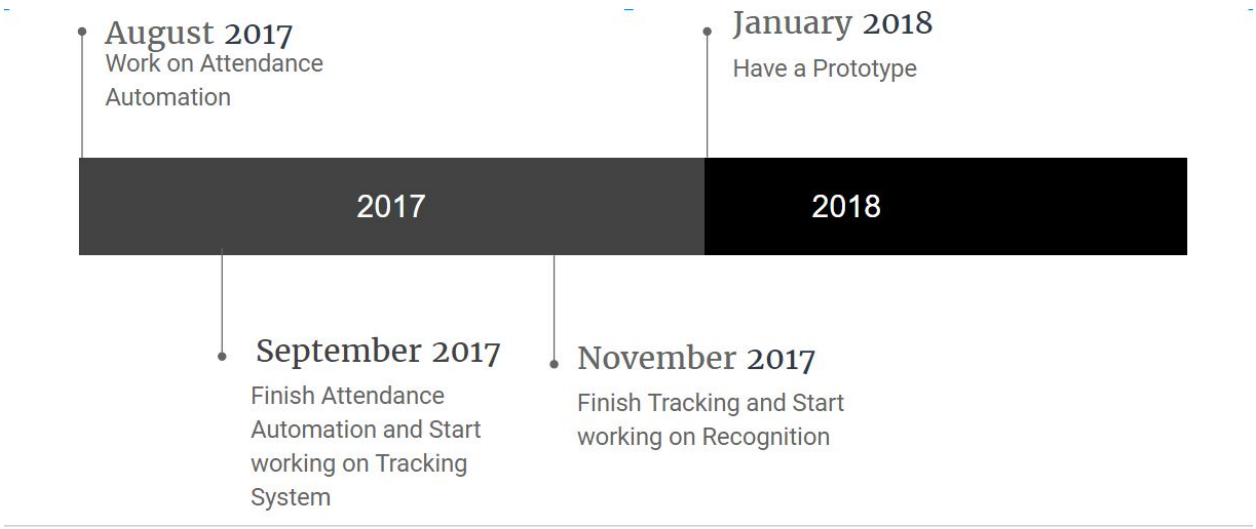
➔ **Tkinter**

Tkinter is a Python binding to the Tk GUI toolkit. It is the standard Python interface to the Tk GUI toolkit, and is Python's de facto standard GUI.Tkinter is included with the standard Microsoft Windows and Mac OS X install of Python. The name Tkinter comes from Tk interface.

Tkinter was written by Fredrik Lundh.As with most other modern Tk bindings, Tkinter is implemented as a Python wrapper around a complete Tcl interpreter embedded in the Python interpreter. Tkinter calls are translated into Tcl commands which are fed to this embedded interpreter, thus making it possible to mix Python and Tcl in a single application.

Python 2.7 and Python 3.1 incorporate the "themed Tk" ("ttk") functionality of Tk 8.5. This allows Tk widgets to be easily themed to look like the native desktop environment in which the application is running, thereby addressing a long-standing criticism of Tk (and hence of Tkinter).

4. Planning of Work


August 2017
Work on Attendance Automation
January 2018
Have a Prototype
2017
2018
September 2017
Finish Attendance Automation and Start working on Tracking System
November 2017
Finish Tracking and Start working on Recognition

5. References

1. Rafael C. Gonzalez and Richard E. Woods, Digital Image Processing, Third Edition, Prentice Hall, 2007. An excellent textbook on algorithms for image processing for upper-level undergraduate students.
2. My Top 9 Favorite Python Libraries for Building Image Search Engines, Adrian Rosenbrock, a nice comparison of popular Python image processing libraries
3. scikit-image Web site, the Web site for a popular Python image processing library
4. Mahotas documentation, another popular Python image processing library. It usually is faster than scikit-image, since more of it is written in C++, but mahotas has less functionality than scikit-image.
5. Rapid object detection using a boosted cascade of simple features
6. Viola, Jones: Robust Real-time Object DeteFace Detection and Tracking using the KLT algorithmction, IJCV 2001
7. L. Sirovich; M. Kirby (1987). "Low-dimensional procedure for the characterization of human faces". Journal of the Optical Society of America A.
8. M. Kirby; L. Sirovich (1990). "Application of the Karhunen-Loeve procedure for the characterization of human faces". IEEE Transactions on Pattern analysis and Machine Intelligence
9. M. Turk; A. Pentland (1991). "Face recognition using eigenfaces" (PDF). Proc. IEEE Conference on Computer Vision and Pattern Recognition. pp. 586–591.
10. M. Turk; A. Pentland (1991). "Eigenfaces for recognition" (PDF). Journal of Cognitive Neuroscience.
11. A. Pentland, B. Moghaddam, T. Starner, O. Oliyide, and M. Turk. (1993). "View-based and modular Eigenspaces for face recognition". Technical Report 245, M.I.T Media Lab.
12. P. Belhumeur; J. Hespanha; D. Kriegman (July 1997). "Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection". IEEE Transactions on pattern analysis and machine intelligence.
13. M. H. Yang (2000). "Face recognition using kernel eigenfaces". Proceedings International Conference on Image Processing. 1. pp. 37–40.
    R. Cendrillon; B. Lovell (2000). "Real-time face recognition using eigenfaces" . Visual Communications and Image Processing. pp. 269–276.
14. DC. He and L. Wang (1990), "Texture Unit, Texture Spectrum, And Texture

Analysis", Geoscience and Remote Sensing, IEEE Transactions on, vol. 28, pp. 509 - 512.
15. L. Wang and DC. He (1990), "Texture Classification Using Texture Spectrum", Pattern Recognition, Vol. 23, No. 8, pp. 905 - 910.
16. T. Ojala, M. Pietikäinen, and D. Harwood (1994), "Performance evaluation of texture measures with classification based on Kullback discrimination of distributions", Proceedings of the 12th IAPR International Conference on Pattern Recognition (ICPR 1994), vol. 1, pp. 582 - 585.
17. T. Ojala, M. Pietikäinen, and D. Harwood (1996), "A Comparative Study of Texture Measures with Classification Based on Feature Distributions", Pattern Recognition, vol. 29, pp. 51-59.