EL213 - Analog Circuits

Project Report

Smart Security Camera with Face Recognition using OpenCV

April 14, 2018

Group 21

1. Rutvik Shah - 201601416
2. Rutvik Kothari - 201601417
3. Ayush Jain - 201601404
4. Himil Patel - 201601409
5. Vidhin Parmar - 201601003
6. Deep Thanki - 201601068
7. Ravi Sawlani - 201601120
8. Ujjval Patel - 201601234
9. Hiren Vaghela - 201601203
10. Dhruvesh Asnani - 201601423

1 Introduction

This system is used to detect and recognize(if it is a person) whether someone is present at some specified place of the house. To do this, we use a camera along with raspberry pi and use some machine learning algorithms to detect some intruder and finally send a mail to the owner of the house. So, the owner can keep a track of people who enter his/her room and can take appropriate action.

2 Components

2.1 Hardware

- Raspberry Pi 3
- Raspberry Pi Camera module
- Power supply battery

2.2 Software

- OpenCV library

3 Block Diagram

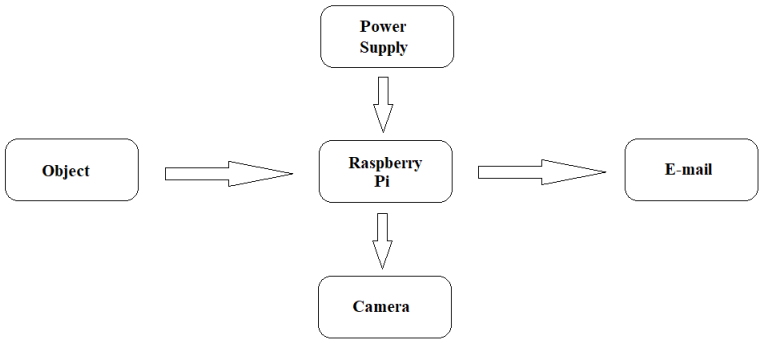

Figure 1: Block Diagram

4 Algorithm Working

4.1 Face detection

We have used OpenCV(Open Source Computer Vision Library) for face detection. OpenCV is a library of programming languages, mainly aimed at real time computer vision. OpenCV uses Haar-cascade algorithm to detect features of an object (a face as an example). It is a machine learning based approach where a cascade function is trained from a lot of positive and negative images. It is then used to detect objects in other images. Below is the description of how the algorithm actually works.

1. A feature can be thought of as a rectangle containing different arrangements of black and white areas which can be used to detect different pixels.

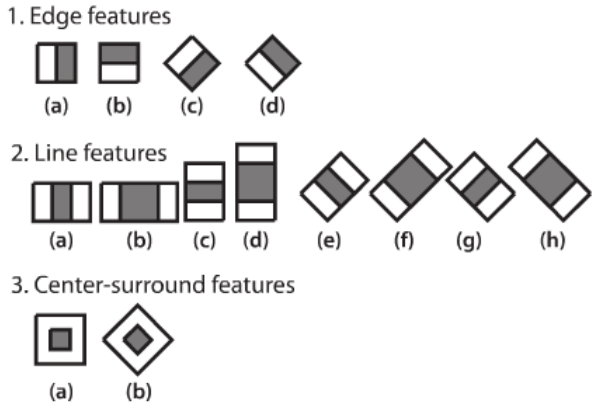

Figure 2: Features

2. But applying 16,000 features to every part of the image can be expensive. To overcome this limitation, features are classified into stages and the features of each stage are applied chronologically. Suppose when the features of stage-1 are applied then all those parts of the image which are found negative (non-facial) are removed which means features of the next stages won't be applied on these parts of the image. Because of this the algorithm is called Haar "Cascade" because of usage of a "Cascade of Classifiers".

3. In this way, different features of the image like nose, eye, lips, etc. are recognised.

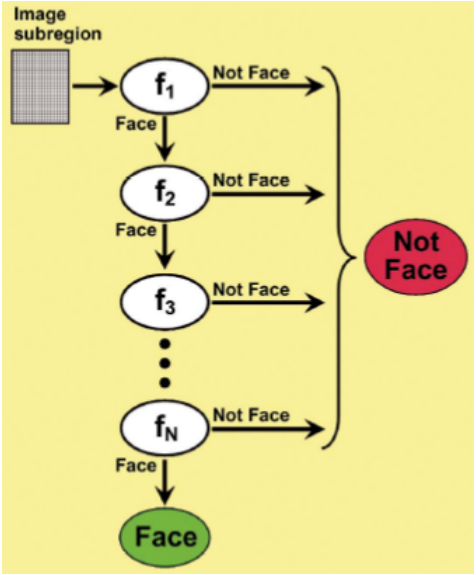

Figure 3: Cascade of Classifiers

4.2 Face Recognition

There are three different algorithms for face recognition:

1. Eigenfaces
2. Fischerfaces
3. Local Binary Patterns Histograms(LBPH)

Here, we use the LBPH algorithm. The basic idea of Local Binary Patterns is to summarize the local structure in an image by comparing each pixel with its neighborhood. Take a pixel as center and threshold its neighbours against it. If the intensity of the center pixel is greater-equal its neighbour, then denote it with 1 and 0 if not. We will end up with a binary number for each pixel, just like 11001111. So with 8 surrounding pixels we will end up with 2^8 possible combinations, called Local Binary Patterns or sometimes referred to as LBP codes. Here is an example:

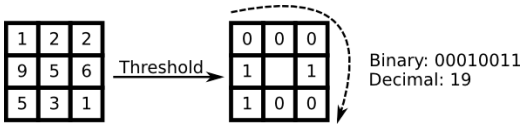

Figure 4: LBPH working

4.3 Algorithm Logic flow

For training, there is a folder called "training-data". The subfolders are named as s1, s2 and so on. i.e. "s" followed by a number. Each of these folders contain training images for different persons. At least 10 images of each person are recommended. Moreover, the images should have some variations. The respective names of the persons have to added to the subjects[] array in the main code at the index indicated by the folder number.

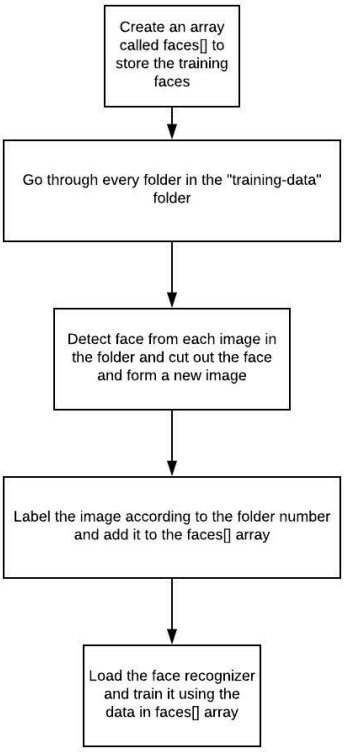

Figure 5: Logic flow for training data

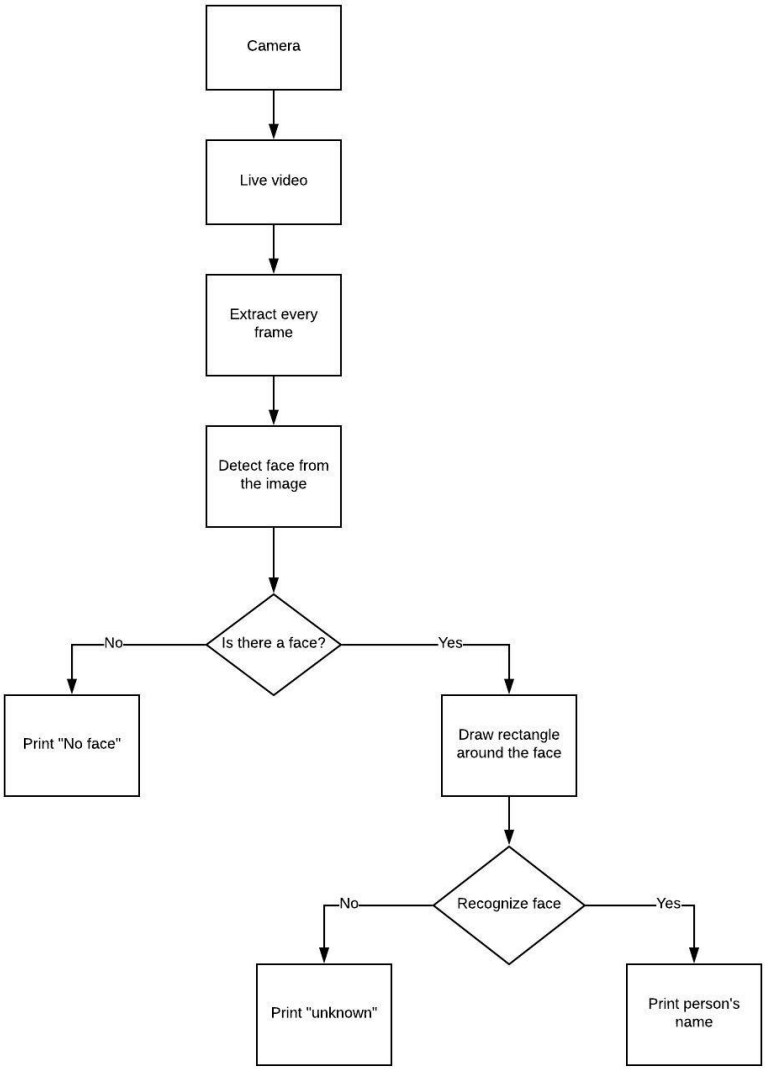

Figure 6: Logic flow of algorithm working

5 Final Expected Output

The camera is always on. Whenever someone comes in front of it and is found to be some person, the software sends a mail to a specified email id along with the image of the person.

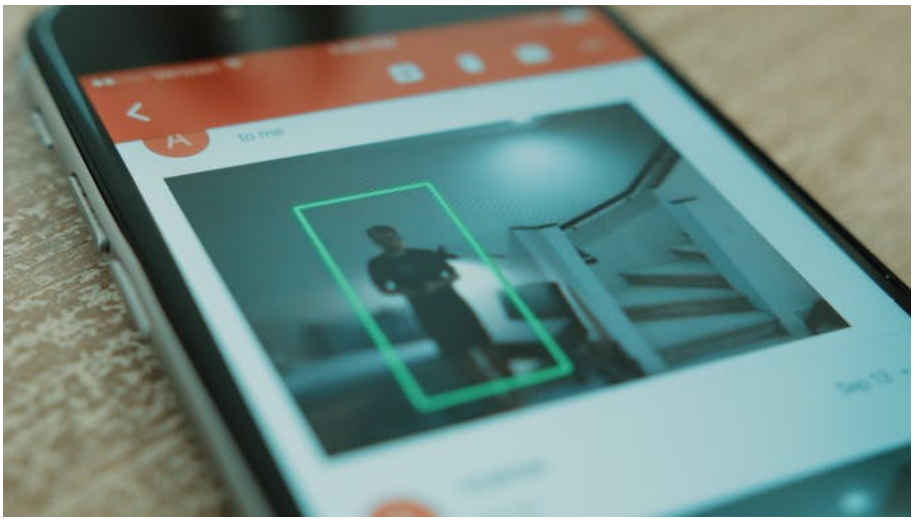

Figure 7: Email after object detection

6 Applications

- Can be used for automated attendance systems.
- For automated door locking/unlocking for home
- In security cameras
- For keeping a track of people who enter a given place.

7 References

1. https://www.hackster.io/hackerhouse/smart-security-camera-90d7bd
2. https://docs.opencv.org/3.3.0/d7/d8b/tutorial_py_face_detection.html
3. https://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html

8 Instructions for setup and use

1. Connect necessary components to setup raspberry pi. Attach the camera module to the raspberry pi.
2. After booting the raspberry pi, open up the terminal and write the following commands.
   (a) $ source -/.profile
   (b) $ workon cv
   (c) $ cd Desktop/e1
3. To add training images for a new face, add around 10 images of the person in a new folder, with name starting with letter 's' followed by the least unused number, in the 'training-data' folder. The names of the images should be natural numbers for example '1.jpg', '2.jpg', and so on in increasing order.
4. Also append the name of the person to the 'subjects[]' array in the 'main.py' file.
5. Specify the sender's and receiver's email id and sender's email id password in the 'mail.py' file.
6. Now type the following command on the terminal '$ python main.py'.
7. Now you will see several images popping up each one of which is a training image that are present in the training-data folder.
8. And then a new window will pop up which will be the live video stream as seen by the camera module of the raspberry pi.
9. Now whenever a face shows up in front of the camera a green rectangle will be drawn around the face, and if the face is recognized i.e. the person's images were there in the training images then the name of the corresponding person will be written above the rectangle and if the face is unknown then 'unknown' will be written.
10. Also that particular frame of the stream will be sent as an email to the given e-mail id.