Internet Of Things (CS G518)
Course Project Report

Project Team members-
Piyush Patil – 2021H1030127P
Pratik Tripathi - 2021H1030130P

Smart Surveillance System using Raspberry Pi and Face Recongnition

Abstract-

Surveillance is one of the important aspects in various fields such as banking sectors, military areas, or personal security. Security systems such as CCTV have proven to be hugely popular for security purposes due to their cost efficient nature and easy maintenance. There are systems available other than CCTV such as Retina scanner, fingerprint scanner, IR lasers, RFID systems only with the drawback that they are cost inefficient with high implementation and maintenance costs. Our project covers all these drawbacks by its efficiency, portability. This surveillance system is low-cost and user-friendly too.

This project was done with "Open Source Computer Vision Library", the OpenCV. In this project, we will be using Raspberry Pi (so, Raspbian as OS) and Python.OpenCV was designed for computational efficiency and with a strong focus on real-time applications. So, it's perfect for real-time face recognition using a camera.

Background/Literature Study-

This system will serve as smart security module for monitoring. Traditional surveillance systems only records the activities based on motion, but this system serves the purpose of facial recognition so as to reduce the error caused due to motion detection .Raspberry Pi camera module is used to capture images once the motion is detected by the UltraSonic Sensor. This system will monitor when motion detected and checks for the faces in the image captured and with the help of face recognition alerts if the face detected is not stored in the database.

This system is largely based on Python programming from detecting the motion to generating an alert. Various Python libraries are used to control Ultrasonic Sensor for detecting the motion, Python is used for Pi-Camera to capture and process images. The captured image is then processed using OpenCV library that integrates with Python.

Methodology/Process/Algorithm-

Face Recognition/Facial Recognition is a category of biometric software which identifies people by their faces. Face is captured by digital camera and the system is trained and then it is capable of identifying the person. The most basic task on Face Recognition is of course, "Face Detecting". Before anything, you must "capture" a face in order to recognize it, when compared with a new face captured on future. The most common way to detect a face (or any objects), is using the "Haar Cascade classifier". Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. 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. Initially, the algorithm needs a lot of positive images (images of faces) and negative images (images without faces) to train the classifier. Then we need to extract features from it. OpenCV comes with a trainer as well as a detector. If you want to train your own classifier for any object like car, planes etc. you can use OpenCV to create one.

Setup description-

We will be using Raspberry Pi, Picam as a Hardware and OpenCV a pre-built, open-source CPU-only library (package) that is widely used for computer vision, machine learning, and image processing applications. It supports a good variety of programming languages including Python. Sequence of steps to perform real time face recognition are:

1. **Installing OpenCV library**
2. **Testing Camera** - Once you have OpenCV installed in your Raspberry Pi, let's test to confirm that your camera is working properly. I am assuming that you have a PiCam already installed on your Raspberry Pi. We must have the camera enabled.

3. **Face Detection** - The most basic task on Face Recognition is of course, "Face Detecting". Before anything, you must "capture" a face in order to recognize it, when compared with a new face captured on future.The most common way to detect a face (or any objects), is using the "Haar Cascade classifier"Object

Detection using Haar feature-based cascade classifiers is an effective object detection method.

4. **Data Gathering** - Starting from Face Detecting, we will simply create a dataset, where we will store for each id, a group of photos in gray with the portion that was used for face detecting.
5. **Trainer** - We must take all user data from our dataset and "trainer" the OpenCV Recognizer. This is done directly by a specific OpenCV function. The result will be a .yml file.
6. **Recognizer** - We will capture a fresh face on our camera and if this person had his face captured and trained before, our recognizer will make a "prediction" returning its id and an index, shown how confident the recognizer is with this match.

Results and discussion

The recognizer will take as a parameter a captured portion of the face to be analyzed and will return its probable owner, indicating its id and how much confidence the recognizer is in relation with this match. From the experiment we conducted our system is able to correctly identify and recognize the faces and is able to predict the output correclty. For best results of facerecognition, large number of images of authorized persons have been taken as datasets for comparison of images to determine authorized and unauthorized persons. About 30 to 40 images of per person with their different facial expressions have been taken into consideration while performing the tests. Proposed IOT based smart surveillance system provides energy management by turning the system ON, based on the occurrence of a particular motion. System will sense the motion and depending on the detected motion system will switch on the camera, capture the image of intruder, recognize it and send a notification.