Smart Surveillance System with Crowd Behaviour Analysis

A PROJECT REPORT

Submitted by

Yashasvi Asthana (15BCE1161)

Ritwik Kala (15BCE1114)

Suryansh Bhardwaj (15BCE1047)

in the partial fulfilment for the award

of

Bachelor of Technology

in

Computer Science and Engineering

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

Vandalur - Kelambakkam Road, Chennai - 600 127

April - 2019

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Computing Science and Engineering

DECLARATION

We hereby declare that the project entitled "Smart Surveillance System with Crowd Behaviour Analysis" submitted by us to the School of Computing Science and Engineering, Vellore Institute of Technology, Chennai Campus, Chennai 600127 in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering is a record of bonafide work carried out by us under the supervision of Prof Harini S. We further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma of this institute or of any other institute or university.

Place: Chennai
Date:

Signature of Candidate (s)

Yashasvi Asthana (15BCE1161)
Ritwik Kala (15BCE1114)
Suryansh Bhardwaj (15BCE1047)

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Computing Science and Engineering

CERTIFICATE

This is to certify that the report entitled "Smart Surveillance System with Crowd Behaviour Analysis" is prepared and submitted by Yashasvi Asthana (15BCE1161), Ritwik Kala (15BCE1114) and Suryansh Bhardwaj (15BCE1047) to VIT Chennai, in partial fulfilment of the requirement for the award of the degree of B.Tech CSE programme is a bona-fide record carried out under my guidance. The project fulfils the requirements as per the regulations of this University and in my opinion meets the necessary standards for submission. The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma and the same is certified.

Guide

Name: Prof. Harini S

Date:

Program Chair

Name: Dr. B Rajesh Kanna

Date:

External Examiner

Signature:
Name:
Date:

Internal Examiner

Signature:
Name:
Date:

Internal Examiner

Signature:
Name:
Date:

ACKNOWLEDGEMENT

We wish to express our sincere thanks to a number of people without whom we could not have completed the thesis successfully.

We would like to place on record our deep sense of gratitude and thanks to our guide Prof. Harini S, School of Computer Science and Engineering (SCSE), VIT, Chennai campus whose esteemed support and immense guidance encouraged us to complete the project successfully.

We would like to thank our Program Chair Dr. B Rajesh Kanna & Co-Chair, Dr. C Sweetlin Hemalatha, B.Tech. Computer Science and Engineering and Project Co-ordinator Dr. B V A N S S Prabhakar Rao, VIT Chennai campus, for their valuable support and encouragement to take up and complete the thesis.

Special mention to our Dean Dr. Vaidehi Vijayakumar, Associate Deans Dr. Vijayakumar V and Dr. Subbulakshmi T, School of Computing Science and Engineering (SCSE), VIT Chennai campus, for spending their valuable time and efforts in sharing their knowledge and for helping us in every minute aspect of software engineering.

We thank our management of VIT, Chennai campus for permitting us to use the library resources. We also thank all the faculty members for giving us the courage and the strength that we needed to complete our goal. This acknowledgment would be incomplete without expressing the whole hearted thanks to our family and friends who motivated us during the course of our work.

Yashasvi Asthana (15BCE1161)

Ritwik Kala (15BCE1114)

Suryansh Bhardwaj (15BCE1047)

Table of Contents

List of Tables

List of Figures

List of Abbreviations

Sr. No.	Abbreviation	Full Form
1	AI	Artificial Intelligence
2	ANN	Artificial Neural Network
3	API	Application Programming Interface
4	CCTV	Closed-Circuit Television
5	CNN	Convolutional Neural Network
6	CSI	Camera Serial Interface
7	DPM	Discrete Phase Model
8	FPS	Frames Per Second
9	GPU	Graphics Processing Unit
10	GSM	Global System for Mobile communications
11	HOG	Histogram of Oriented Gradients
12	HTTPs	Hypertext Transfer Protocol Secure
13	IDE	Integrated Development Environment
14	IoT	Internet of Things
15	IR	Infrared
16	LBP	Local Binary Patterns
17	LFA	Linear Function Approximation
18	ML	Machine Learning
19	PCA	Principal Component Analysis
20	PIR	Passive Infrared Sensor
21	RFID	Radio-Frequency Identification
22	Rpi	Raspberry Pi
23	SDLC	Software Development Life Cycle
24	SSD	Single Shot Detector
25	SSH	Secure Shell
26	SVM	Support Vector Machine
27	UI	User Interface
28	USB	Universal Standard Bus
29	UV4L	User Video for All
30	VM	Virtual Machine
31	VPN	Virtual Private Network
32	WBS	Work Breakdown Structure
33	WLAN	Wireless Local Area Network

EXECUTIVE SUMMARY

The Smart Surveillance System is an active and real time security and monitoring system that aims to supersede the current passive CCTV based surveillance systems as they limit themselves by only recording the live footage through security cameras and require continuous human monitoring. This system can cater to the needs of individuals for home security and even large organizations. It provides a face recognition based authentication module that tries to identify every person in real time be it the owner of the house and his family or the employees of an organization. Similarly, the concerned authority could be notified and warned if an unwanted personnel or a criminal is identified immediately. The system also tries to analyze human crowd behaviour in order to find abnormal behaviour and actions like crowd diverging away or converging to a point. The live feed can be provided by well-placed cameras or even use the previously installed ones. Various algorithms for Human and Face Detection, Face Recognition, Object Detection have been efficiently implemented for the system to work in real time. The user can update the list of known or unwanted personnel for the system to adapt and work accordingly. Notifications can be in the form of alarms, messages, photo snippets, with the live feed available for the user on his phone or laptop anytime.

ABSTRACT

The advent of technology has led to innovative advancements in the field of science and medicine, however its impact on security and surveillance still is not that prevalent. While organizations and government do install cameras at important locations the footage is stored away for future use. The huge amount of cameras also make it humanly impossible to be monitored continuously by security personnel as a few people are left to monitor tens of cameras. Hence a more active approach needs to be undertaken in the form of analyzing the live feed thereby decreasing the load on human monitoring while also providing insights that could have been missed otherwise.

The current CCTV surveillance systems are very basic and passive; they capture video and transmit it to a local server, where it is either stored or monitored by a person live. The system itself doesn't attempt to analyse or process the live feed but just transmits the same. In this project, the idea is to build a smart and active surveillance system which alerts the users, based on factors like identity and behaviour of people from live video cameras. Raspberry pi and pi camera would be used to capture the live video and transmit it to the server, where the video is then analysed. Pre-installed cameras can also be used for the same. The system would first detect any human, then check if face recognition is possible. If the person is unknown, an alert to the user would be sent stating that someone unknown is visiting the premise and whether that person could be a possible threat based on his past actions or concern by the user. On the other hand, if the person is known, a message with any information of that person would be sent to the user along with alerts regarding the same. Hence, the system actively processes the live feed as it tries to identify the people while also trying to analyse their behaviour and alert the concerned authority, taking into consideration previous crime record or warning by the user.

1. INTRODUCTION

1.1 Objective

Many cities still face crimes and antisocial behaviour problems like fights, breaking and entering shops, vandalism, etc. These places tend to have video cameras already installed but the issues is that the usage is limited to watching the incident at a later time when it has already happened. The surveillance systems are too passive and hence are not able to automatically analyse the live video data. The advantage of such analysis is that it could detect unusual and suspicious events like people converging, trying to hide themselves or running erratically and immediately alert the concerned authority. The system could also be used to identify people and notify unwanted or unexpected presence.

Since the amount of video data collected everyday by such surveillance cameras increase, so does the need for automated smart systems to detect and identify suspicious activities performed by people or objects. This is especially true because it becomes humanly impossible for a few people to manage the huge amount of live data across hundreds of cameras. For a single operator it is very challenging to monitor multiple screens while looking for abnormal activities and hence increases the chance of missing important events. Hence a shift towards a more active and smart system is the need of the hour.

1.2 Background

Surveillance and authentication are one of the most important security aspects in various fields such as banking sectors, military, and even personal security. Due to exponential rise in burglary and theft activities, surveillance systems are proving to be a great source of security. With the advancements in technology, people are relying on the same for their safety. Security systems such as CCTV have proven to be hugely popular for security purposes due to their cost efficient nature and easy maintenance.

Surveillance systems are deemed to be highly important by law enforcement officers for the investigation and even prevention of criminal activities and also for identifying and monitoring threats. Also, surveillance systems have always played a vital role in dealing with theft and burglary related crimes. These CCTV systems are not well optimized and might result in high power consumption and memory wastage. Moreover, they do not give real-time alert on any suspicious activities detected. There are systems available other than CCTV such as Retina scanner, fingerprint scanner, IR lasers, and RFID systems, only with the drawback that they are cost inefficient as they tend to have high implementation and maintenance costs. Hence, such systems are not a preferred way for security purposes for both small scale and large scale applications. The proposed system covers many of these drawbacks efficiently as this surveillance and authentication system is low-cost and user- friendly with real-time updates.

The rise in CCTV coverage can also be observed anecdotally. There's a steady uptick in CCTV clips circulating on social platforms like WhatsApp and Facebook, capturing crimes or funny events that would otherwise have gone undocumented. Many of the sensational crimes recently, including multiple incidents of murder in

Tamil Nadu, were captured on CCTV cameras, distilling the pure horror of those moments on the mobile screens, and also offering valuable proof to nail the culprits.

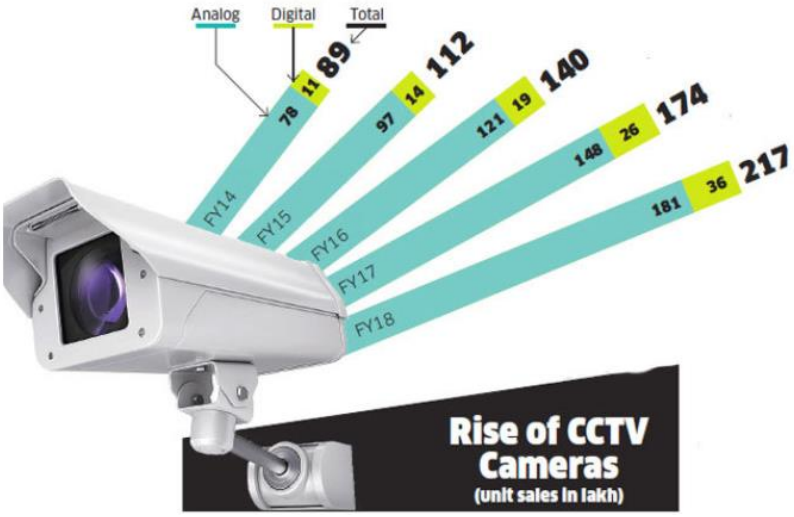

Figure 1: CCTV Camera Unit Sales in India

1.3 Motivation

It is nearly impossible to guarantee safety in our homes or any organization that we work in. Even CCTV systems that are currently deployed to improve our safety are passive and do not work the way they should work. In CCTV and other forms of security surveillance, usually there may be one or two guards to monitor tens of cameras. They can, and they many times do, miss some unexpected entries or events. We can surely catch the suspect afterwards by seeing the CCTV recordings, but we were still unable to prevent the crime. If it is a theft the consequence might not be huge as there still is a possibility of retrieving the lost belongings. But if the crime is something very serious and harmful leading to loss of human life, it can't be undone. This is where our current systems fail. We are very well equipped to catch the suspect later but by then it might be too late as the deed would be done.

The increase in the number of CCTV cameras for surveillance increases the number of screens to be monitored by the security personnel. This increase in workload and monitoring multiple screens simultaneously increases stress by a huge amount. As the number of screens to be monitored by the human operator increases, the concentration of human observation on each screen simultaneously decreases with time. For a single operator it is very challenging to monitor multiple screens while looking for abnormal activities and hence increases the chance of missing important events. Therefore, there is an increasing demand for active and automated surveillance systems and algorithms with the main aim to alert the concerned authority in case of any abnormal or dangerous situation. The advent of Machine Learning and Artificial Intelligence has made it possible to incorporate algorithms for object detection and human recognition. Advancements in the field of Computer Vision and Image Processing also warrants the need for such automated and smart surveillance systems.

2. PROJECT DESCRIPTION AND GOALS

2.1 Project Description

The Smart Surveillance System is an active and real time security and monitoring system that aims to supersede the current passive CCTV based surveillance systems as they limit themselves by only recording the live footage through security cameras and require continuous human monitoring. The project has a Movement Detection module implemented using Image Processing techniques. The authentication module has been implemented using Face Detection and Face Recognition algorithms. Human and Object Detection for identifying and understanding crowd behaviour and patterns have also been included for the project. Finally an Alert system has been designed in order to notify the concerned authority. The hardware requirements are very flexible thereby making it possible to implement the system by using pre-installed cameras.

2.2 Goals

The goals of the project include –

- To design and implement a smart an automated surveillance system.
- The device that would alert the concerned authority if any unknown personnel enters restricted premises.
- The device can also act as a face recognition based authentication system.
- Provide insights from analysing crowd abnormalities like erratic running, converging or diverging, etc.
- Delve into face detection/recognition, object and human detection algorithms while analysing ways to lower overall computation and boost up performance.

3. LITERATURE SURVEY

Security systems such as CCTV have proven to be hugely popular for surveillance purposes due to their cost efficient nature and easy maintenance. The Raspberry Pi satisfies this criteria in that it is a cost efficient yet powerful computer which can be interfaced and incorporated with other modules to finally deliver systems with immense functionality. A lot can be done on it ranging from controlling motor speed, automatic lighting and alarms, VPN server, security system etc. [1], the latter is especially of great interest in this project. Raspberry pi enables integration with various software thereby allowing message and voice alerts [2] as well as integration with custom designed mobile applications [3]. Raspberry Pi being a self-sufficient device in itself fairs better than PIR Sensors, GSM modules, etc. and overcomes many of the problems faced [4].

Face recognition is yet another recognized and sought-after technologies in the field of machine learning and Face Detection is the first and most basic step for carrying out face recognition. The face recognizing systems available so far usually maintain a database with every pre-processed human face with its corresponding unique features as determined from each face. These features are what the computer understands instead of features like nose size, ear length, etc. and are stored together with the respective individual's face. Therefore, when a query is raised, the unique features are extracted from the face and then compared with the features in the database and then looked for the closest, if not a perfect match [5].

For still images, algorithms like Haar, LBP and HOG show immense potential with high accuracies even over many datasets [6]. In case of video stream having varied orientations and background noise, traditional image based algorithms perform

subpar [7]. Neural Network based algorithms can be trained to become robust to such noise while focusing on real-time face recognition [8]. Principal Component Analysis (PCA) can also be used to provide fast response and reduce dimensionality of training dataset and Euclidean distance between projected vector and known face vector is another suitable measure for classification purpose [9].

Sr. No.	Approach	Representative Work
1	Holistic Methods	
	Eigenfaces	Direct application of PCA [Craw and Cameron 1996; Kirby and Sirovich 1990; Turk and Pentland 1991
	SVM	Two-class problem based on SVM [Phillips 1998]
2	Feature Based Methods	
	Pure geometry methods	[Kanade 1973; Kelly 1970], [Cox et al. 1996; Manjunath et al. 1992]
3	Hybrid Methods	
	Component Based	Face region and components [Huang et al. 2003]
	Hybrid LFA	Local feature method [Penev and Atick 1996]

Table 1: Categorization of common Face recognition Techniques

A lot of ideas of automatic CCTV image and video analysis and prediction of harmful situations have been proposed and analyzed in various recent studies. The main challenge that occurs, is the availability of the perfect dataset that would yield the best

results after applying machine learning and deep learning algorithms. There are various approaches and models for object (firearm) detection which could be used. To simplify the work required, authors of [11] decided to work only on pistols and revolvers. Authors collected the dataset in a particular environment because most of the algorithms were sensitive to light conditions. In 2017, Nakib used Image Net dataset containing datasets of knives, machine guns, revolvers, shotguns etc. [10]. Later, necessary amendments were made such as using bounding box parameters to crop each and every raw image. In [12], Roberto Olmos not only used the pre-trained model on Image Net database but also self-created a database, divided into 4 smaller database for working on two different approaches – the sliding window and the region proposal approach. The accuracy was quite less (84.21%) because of large number of false positives after testing.

Human detection is particularly difficult due to intra-class variability. The Author in [13] used PASCAL, INRIA person dataset and DPM model that represents any object through mixture of multiscale part models. Local appearance properties were stored using the models. Authors in [14] used IMFDb for acquiring firearms images used or showed in movies or series. Authors also used Scrapy library to develop a web crawler for training more data. Only images containing small arms were considered. 96.26% of accuracy was obtained that stood out for its speed of detection, capable of working in real time environment.

Sr. No.	Title	Dataset Used	Training Data	Testing Data	Classes	Accuracy
1	Crime Scene Prediction by detecting Threatening Objects Using Convolutional Neural Network [10]	Image Net Dataset	2,406	2,400	Blood, knife, machine guns, Revolver, Shotguns, Pistols	90.2%
2	Automated Detection of Firearms and Knives in a CCTV Image [11]	Self-created dataset	8.5 min video of 12,000 frames	8.5 min video of 12,000 frames	Pistols and revolver	96.69%
3	Automatic Handgun Detection Alarm in Videos Using Deep Learning [12]	Image Net Database & Self-created	28,688	608	Pistols and other images	84.21%
4	Automatic Weapon Detection in Videos [13]	PASCAL & INRIA Person Dataset + Self-created for gun dataset	479 for gun detection	270 for gun detection	Guns, Humans	68%
5	Firearm Detection Using Convolutional Neural Networks	IMFDb	5,916	669	Sub machine guns, carbines, rifles ,shotguns, light machine guns	96.26%

Table 2: Summarization of related Object Detection papers

Many object detection algorithms and models were proposed in the past few years but most of them work on large objects such as humans, number plates etc. Despite the number of projects in the field, very few algorithms work efficiently and with decent accuracy.

3.1 Neutral Networks

Artificial Neural Networks are composed of layers of computational units called neurons. Each layer of neurons is connected to the next subsequent layer of neurons. The work of each neuron is to multiply an initial value by some weight, sum results with other values coming from the all the neurons of the previous layer, adjust the resulting number by the neuron's bias, and normalize the output with an activation function (for example sigmoid function).

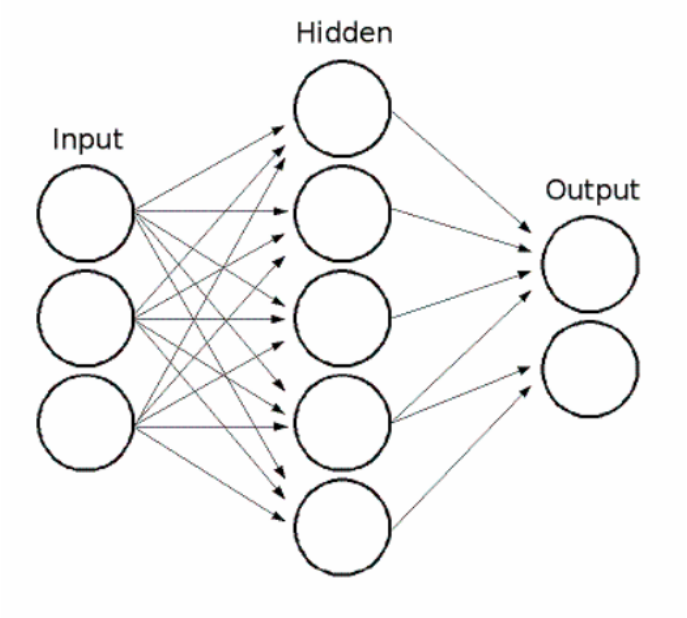

Figure 2: Basic Neural Network Design

The weight of every neuron is adjusted according to the loss in the next neuron. This way the error is back propagated, and weights are adjusted accordingly to minimize the loss. Each iteration decreases the error bit by bit. After this learning phase the model is ready to predict classes, given the attributes.

3.2 Convolutional Neutral Networks

Convolutional neural networks are deep artificial neural networks that are used primarily to classify images, cluster them by similarity and perform object recognition within scenes. The image features can be used as attributes to train a model to classify images. The convolution layer is the main building block of a convolutional neural network. It comprises of a set of independent filters. Each filter is independently convolved with the image. All these filters are initialized randomly and become the parameters which will be learned by the network subsequently.

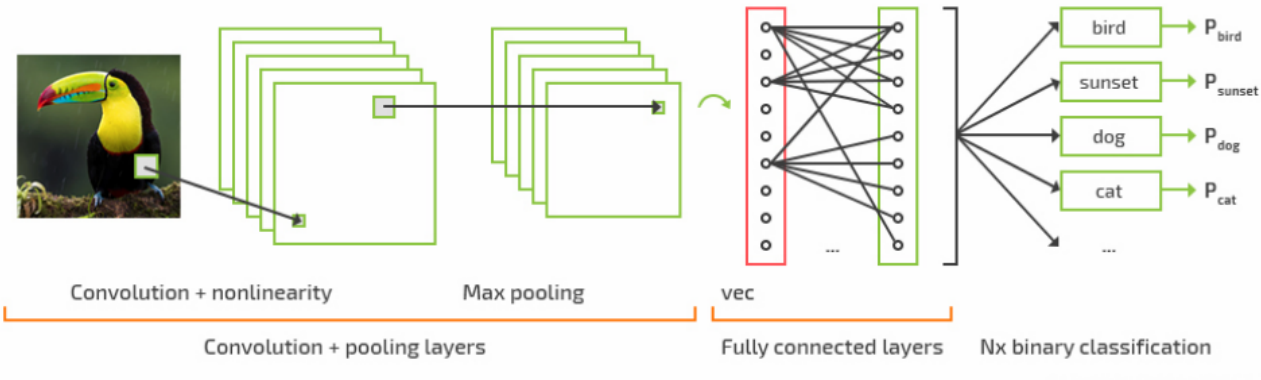

Figure 3: Architecture of a CNN

Convolutional layers apply a convolution operation to the input, passing the result to the next layer. A pooling layer is another building block of a CNN. Its function is to progressively reduce the spatial size of the representation to reduce the amount of parameters and computation in the network. Pooling layer operates on each feature map independently. The most common approach used in pooling is max pooling. Max pooling uses the maximum value from each of a cluster of neurons at the prior layer.

A CNN is able to accurately capture both the Spatial and Temporal dependencies in an image by applying relevant and flexible filters. The architecture performs a better learning of the image dataset because of the reduction in the total number of parameters involved and also through the reusability of weights. In other words, the network can be trained to understand the sophistication of the images better while maintaining accuracy.

4. TECHNICAL SPECIFICATION

4.1 Hardware Specification

4.1.1 Raspberry Pi 3

The Raspberry Pi 3 supersedes its antecedents as it has an even faster processor on the motherboard to further increase its speed. It also includes Bluetooth and Wireless-Fidelity with lower energy capacities to improve the usefulness and enhance the ability to control more capable gadgets via the USB ports.

Salient features:

- Quad Core 1.2GHz 64bit CPU
- 1GB RAM
- On board Bluetooth and Wi-Fi
- CSI camera port in order to connect the Pi camera module
- Many add-ons are already available for RPi Continuous improvement and ability to load normal Linux

Figure 4: Raspberry pi 3

The Wi-Fi functionality is especially useful as we can connect the Raspberry Pi to a laptop under a common network to get live feed through the camera module of Raspberry Pi which is a major sub-module of the project.

4.1.2 Raspberry Pi 5 MP Camera Module

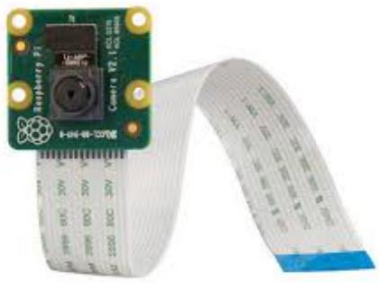

Figure 5: Raspberry Pi Camera Module

The original camera came with a 5MP OmniVision OV5647 sensor with a resolution of 2592 x 1944 pixels. In terms of video, 1080p is possible from the camera, alongside slow-motion modes, however the resolution becomes low. This 5mp camera module is capable of 1080p video recording and still images clicking and connects directly to the Raspberry Pi very easily. We just need to connect the ribbon cable that is included to the Camera Serial Interface port on the Raspberry Pi making the camera good to go.

4.1.3 Nvidia Geforce GTX 1060 GPU

The Nvidia GeForce GTX 1060 with the Max-Q design is a mobile high-end GPU from the Pascal series. The GPU has a base speed of 1,063 - 1,265 MHz and can even reach the high speeds of 1,340 to 1,480 MHz. It has the parallel computing platform and programming model used to harness the power of the GPU (1,280) memory clock speed and type (8 Gbps, GDDR5), 192-bit bandwidth and VRAM (up to 6GB) with additional support of OpenGL, OpenVL and Vulkan.

Figure 6: Nvidia GTX-1060 GPU

4.2 Software Specification

Sr. No	Software	Version	Use Case
1.	Python	3.5	The primary scripting language used for backend and frontend
2.	OpenCV	4.0	Helps in processing the live video stream
3.	Dlib	19.16.0	Contains relevant machine learning algorithms
4.	PyCharm IDE	2019.1	Python Development tools
5.	Qt Creator	4.8.2	Integrated GUI layout and designer
6.	Pushbullet API	1.5.7	Used as an alert system that can notify the user as per the threat
7.	Anaconda	4.4.0	Python virtual environment manager
8.	Jupyter Notebook	5.7.6	IDE for python development
9.	MobaXterm	11.0	SSH client for windows used to connect to raspberry pi and program it
10.	NumPy	1.16.1	Scientific computing with python
11.	Labelimg library	1.8.1	Used for labelling dataset
12.	UV4L	1.0.0	Streaming live video from raspberry pi to server
13.	Tensorflow gpu	1.13.1	Used for faster numerical computation while training and running the object detection models

Table 3: Software Specification

5. DESIGN PLAN

5.1 Engineering Design

The engineering design process is a series of ordered steps that engineers tend to follow to come up with a solution to a problem. The steps taken into account for the engineering design of the project are as follows –

1. Problem Definition – The aim of the project is to build a smart surveillance system that can supersede the current passive CCTV based surveillance systems as they limit themselves by only recording the live footage through security cameras and require continuous human monitoring. This system can cater to the needs of individuals for home security and even large organizations.

2. Background Research – Studying different methods, trends in technology and various libraries that can be useful for the implementation of the project such as HOG (Histogram of Oriented Gradients) for face detection. Various emerging technologies such as OpenCV, UV4L and other libraries were used for the implementation.

3. Requirement Specification – A low cost and efficient smart surveillance system that focuses on the safety of people in homes or any organization and predicts any wrong deed before it is done.

4. Solutions/Proposed Solutions – Exploring options of using various machine learning algorithms such as face detection, action recognition, human behavioral analysis, motion tracking systems for detecting threats and providing warning to user.

5. Selected Solution – The selected solution is the use of face and human detection models, providing firearms detection and abnormal activity detection where there is crowd.

6. Procedure/Development Work – Development of individual modules simultaneously and then integrating them.

7. Prototype – By integrating the modules, the final system gets controlled via a Graphical User Interface (GUI) for making a prototype

8. Testing – Testing the efficiency of different modules using real time video from camera and integration testing of the system

9. Make Observations – For increasing the efficiency, more training is required and other ideas such as adding known threats feature can be added to the project.

10. Redesign – The improvements and the ideas after observing the project were introduced making the system better and more efficient.

The following was the design ideology for the project -

1. Raspberry pi uses pi camera module to capture video.
2. Raspberry pi streams the video to a server.
3. The server analyses the video:

a. Detects and identifies human

b. Checks if facial recognition is possible

c. If the person is unknown or facial recognition is not possible, analyse whether the person is a possible threat, and alert the user accordingly.

d. If the person is known, then send a message with information of that person to the user.

e. Analyse crowd behaviour and look for anomalies in case of which notify the user.

5.2 Design Approach

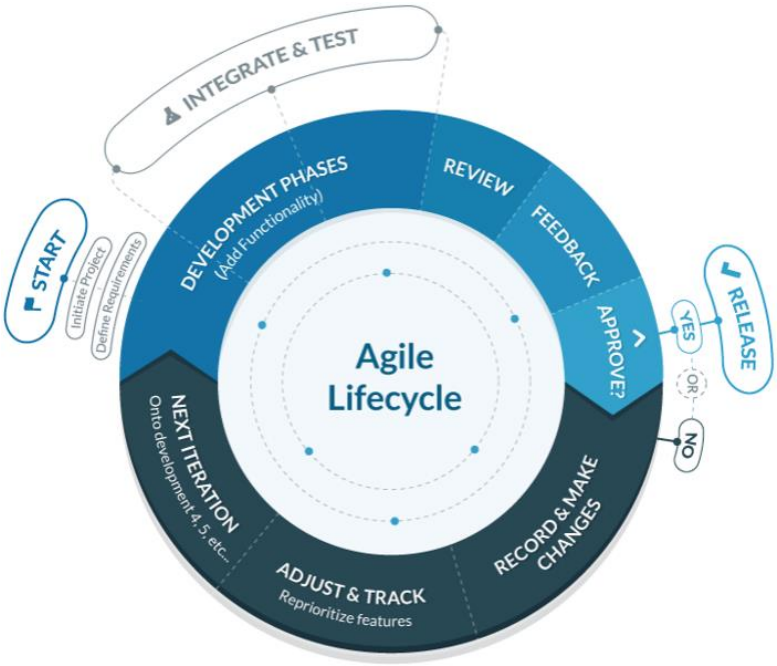

Figure 7: Agile Software Development process

Agile is an iterative approach for software development that believes in building software incrementally from the start, and not just finish the project in one go or trying to deliver everything at once but in a time bound manner. Agile methods or agile processes generally promote a disciplined project management process that supports continuous inspection, management and adaptation also maintaining a leadership philosophy that encourages teamwork, self-reliance and accountability to allow for rapid delivery of high-quality software, while also considering customer needs and company goals.

By choosing Agile as the SDLC process, allows one to curate the needs of the rapidly changing environment by embracing the idea of incremental and continuous development and develop the actual final product as it is almost impossible to gather an unchangeable yet complete set of software requirements before the start of the project. The work was constantly updated based on valuable feedback given by the mentors and through mutual discussions thereby suiting the flexible Agile approach.

Advantages of Agile approach for the project –

- Supports the concept of immediate feedback which can then be used to improve the software in the next increment.
- Suggestions by guide and panel members could be incorporated even at later stages of the project.
- The project has numerous modules and agile development eased the integration of them all.
- The prototypes were developed rapidly and on time for testing and verification.

5.3 Use Case Diagram

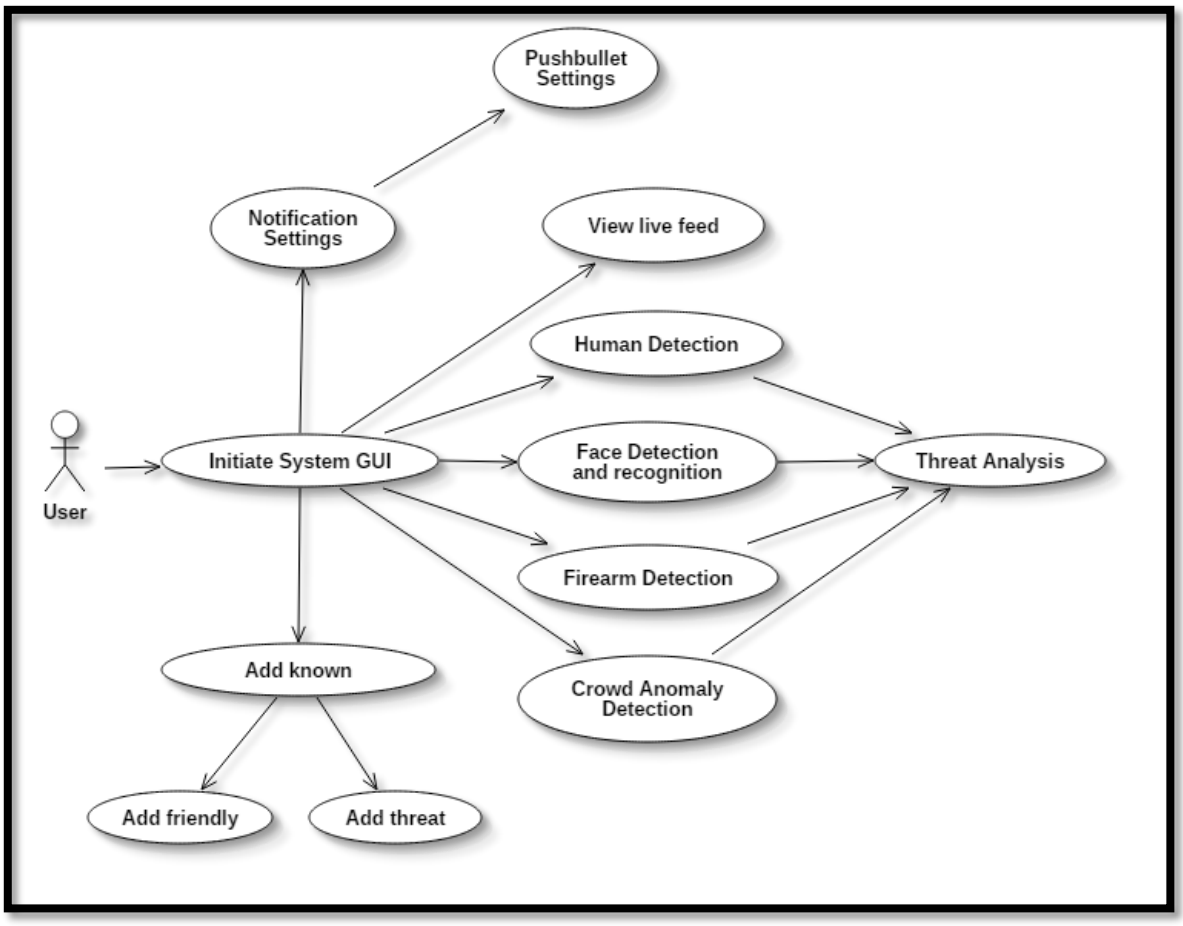

Figure 8: Use Case Diagram

The proposed system for the project is depicted by the use case diagram in the figure mentioned above. The actor (user) has the authorization to initiate the system GUI (Graphical User Interface) and can perform different functionalities. The various functionalities available for a user are:

- User can view the live feed anytime on web browser.
- User can turn on or off the various features such as human detection, face recognition, firearm detection and crowd anomaly detection. This would result in multiple checks for unknown threat and notify the user if any threat is predicted by the system

- User can add known threats or friendly “faces” to detect threat from known personnel or to avoid unnecessary notifications.
- User can make changes in notification settings and can also further change PushBullet settings for sending messages and snapshots.

6. PROJECT MODULES

6.1 Movement Detection

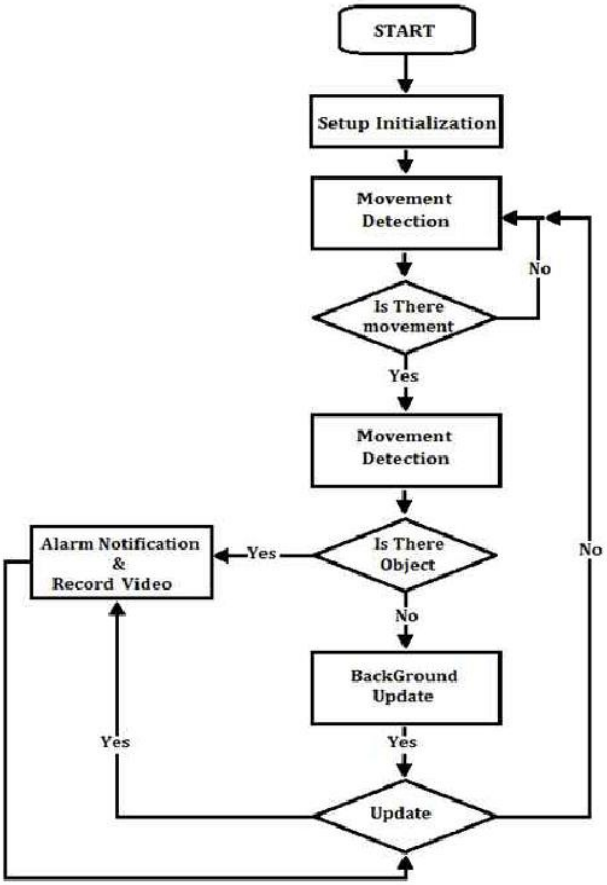

Figure 9: Movement Detection Flow Chart

Initially, a minimum area, which is the minimum size in pixels for a region of an image to be actually considered as “motion” is defined. There is definitely a minute chance that some smaller regions of an image may have changed unexpectedly because of noise or changes in the overall lighting conditions. However, these minute regions aren’t exactly actual motion thereby warranting the need to consider a minimum size for a region so that such cases of false-positives can be tackled.

Now, in order to process a frame for preparing it later for movement analysis, first the frame must be resized down, to a width of 400 pixels as this eliminates the need to process the large and unnecessary, raw images straight from the video stream.

The image is also converted to gray-scale thereby removing the colour since it has no influence on the algorithm for detecting motion and movement. Finally, Gaussian blurring is applied to smooth the images. What is important to analyse is that even consecutive frames of a video stream might not be identical which is due to minimal variations in the sensors of the camera thereby making it very difficult for two frames to be exactly the same as some number of pixels will most certainly have different values of intensity. That said, this must be accounted for and Gaussian smoothing is applied that in return helps in smoothening out of the high frequency noise that might result in throwing the motion detection algorithm off. Computing the difference between two frames is a simple image subtraction and the absolute value of their corresponding pixel intensity differences is then calculated.

delta = |background_model – current_frame|

The frameDelta is then thresholded to reveal regions of the image that only have significant changes in pixel intensity values. Given this threshold image, contour detection is then applied to find the outlines of these white regions. The small, irrelevant contours are filtered out. If the contour area is larger than the defined min-area, the bounding box surrounding the foreground and motion is drawn while also updating the text status string to indicate that the room is “Occupied”.

6.2 Face Detection and Recognition

Face recognition is one of the most sought-after technologies in the field of machine learning. In recent times, the use cases for this technology have broadened from specific surveillance applications in government security systems to wider applications across multiple industries for user identification and authentication, consumer experience, health, and advertising. Face Detection is the first and essential step for face recognition, and it is used to detect faces in the images. It is a part of object detection and can use in many areas such as security, bio-metrics, law enforcement, entertainment, personal safety, etc. For the project, various face detection and recognition algorithms like Haar Cascade, Histogram of Gradients and Neural Network were implemented while noting the computation, efficiency and performance on the hardware.

For Face Detection, the image is converted into black and white and for every pixel, its surrounding pixel is looked at. The idea is to figure out how dark the pixel is with respect to its surrounding pixels and then draw an arrow showing the direction in which the image gets darker. To analyse the basic flow of lightness and darkness, the image is broken up into small squares of 16x16 pixels each and the squares are replaced with the arrow directions. This results in the original image to now become a very simple representation that captures the basic structure of a face.

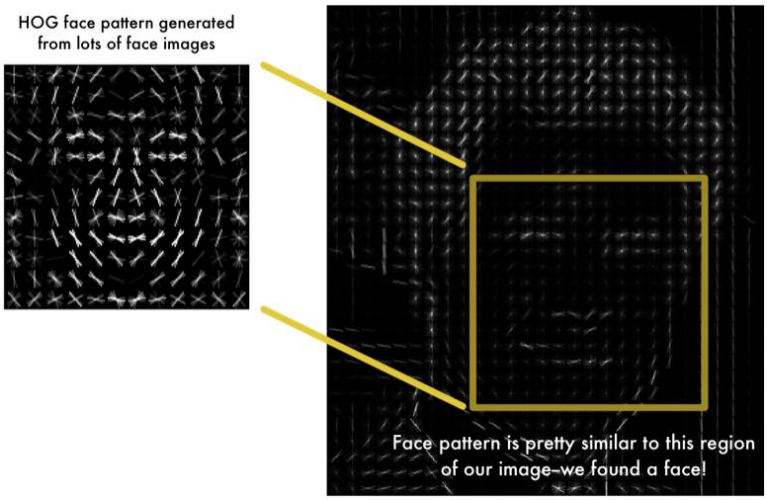

Figure 10: HOG based Face pattern

The easiest and most common approach for face recognition is to directly compare two images however this is not at all viable as looping through millions of photos while comparing with each one of them would take way too long and would also be computationally expensive. Hence the idea is to extract a few basic measurements from each face. However measurements like ear size, nose length, eye colour, etc. that our obvious for humans doesn't really make sense to a computer looking at individual pixels in an image.

Hence, for Face Recognition, a deep convolutional neural network is trained but instead of training the network to recognize pictures or objects, it is trained in order to generate 128 measurements for a face. Two images of the same person and 1 of a different person is looked at and the algorithms is tweaked so that the measurements of the same person's picture are closer and further away from the other person's. By repeating the same, the neural network learns to reliably generate 128 measurements for each person. Finally, even a simple Linear SVM classifier could be used to check for the closest measurement with the database of known faces. However to decrease the computation for the algorithm to work in real time, the Euclidean distance between the 128 encoding vector is used as the means to find the closest matching face. By

setting a threshold distance separation, we can also state that the face being tested might not match any of the ‘known faces’ thereby indicating that the test face is an unknown one.

6.3 Object Detection

Object Detection is one of the basic steps towards AI based vision. It is required to analyze the environment and create a relationship between multiple objects that are co-existing. Recognizing people is the first step towards recognizing and interpreting their actions. Object detection also helps to generate alerts based on certain objects than possess threat, for example guns. According to the National Crime Victimization Survey, 467,321 US citizens were victims of a crime committed with a firearm in 2011. In the same year, data collected by the FBI shows that firearms were used in 68% of murders, 41% of robbery offenses and 21% of aggravated assaults nationwide. Therefore, preventing crimes involves detecting firearms.

Tensorflow object detection library is used to train models to recognize people and firearms. Tensorflow provides various object detection classifiers with specific neural network design. For example SSD-MobileNet model is one of the fastest running model. Similarly, there are multiple classifiers based on number of number of layers and type of networks. Different models are suitable for different tasks; there’s a tradeoff between detection speed and accuracy, higher the speed lower the accuracy and vice versa. These models can be used for out-of-the-box inference, or initializing new models when training on custom datasets for custom classes (as done in this project).

The first step to train a model that can recognize particular objects in images is to create an image dataset with bounding boxes in corresponding xml files. The image dataset should contain around 300-500 images with people in them. “ImageNet” provides image datasets with bounding boxes in xml format but that might not be sufficient always. Other way is to manually collect images and create bounding boxes using open source libraries, like “labelImg” (as done in this project). After creating this database, the next step is to collect the data from all the xml files and create train and test csv files which contain image name and 4 co-ordinates of the bounding box for every image. Then using these csv files TFRecords for train and test data is generated. Then the training record is used to train the model and the test record is used to evaluate the trained model.

6.3.1 Human Detection

Multiple images of people were collected to create a database of around 300 images. These images were labelled and the configuration used to train the model was ‘ssd_mobilenet_v1_coco’, as this is faster and hence more suitable for a project based on live feed. The time taken to train the model with 20,000 steps on GTX 1060 was around 3 hours.

6.3.2 Firearm Detection

Images of firearms with bounding boxes are available on ‘ImageNet’, and around 400 images were collected; 10% of that batch was used for model evaluation purpose. The configuration used to train the model was ‘ssd_mobilenet_v1_coco’, with various changes in batch size and number of steps. The time taken to train the model with 25.000 steps on GTX 1060 was around 4 hours.

6.4 Graphical User Interface (GUI)

The graphical user interface (GUI) is a kind of user interface that helps users to interact with electronic devices through graphical icons and widgets instead of text based interaction such as command labels. Using different machine learning algorithms by running them separately, one by one would create disarray for the user. The efficient and the simplest solution is to make a GUI that would have multiple options (push buttons) to turn on or off any feature such as Human Detection, Face Detection etc. as mentioned in the above modules.

The GUI is made using QT Creator application which allows the developer to build GUI easily using widgets. Then the file is saved in ".ui" extension. This file is converted into python file using command

mainwindow.ui –o UI_MainWindow.py

A new main python file imports this file for starting the user interface. For writing the backend code for the UI, PyCharm is used. PyCharm helps in writing the backend code and the basic functioning of the GUI.

The GUI consists of 6 checkboxes which are:

- Face Detection
- Human Detection
- Firearm Detection
- Crowd Anomaly Detection
- Add known
- Notification Settings

On checking each of these checkboxes, the algorithms which were running individually before will run together. So, the user would be able to run different algorithms according to his/her needs. This would enhance the flexibility of the project and would be a simple and efficient way to manage all the disarray.

Benefits of Implementation:

1. User can choose what all algorithms to run according to his/her own needs. For example, if someone wants to run only face recognition and human detection then he/she can check the box in front of those two options and the output would vary accordingly. If the system is used at a public place with obscure vision, face recognition might not work as desired etc. so the system can be only set to detect crowd anomaly detection in case of any unexpected events.
2. Ease of use in updating and storing data of known friendly people and known threats.
3. Users can change the notification settings easily and turn off the notifications if they are in a meeting or don't want to be disturbed. Users don't have to open up the code and make amendments.

6.5 Crowd Behaviour Analysis and Anomaly Detection

To automatically understand human behavior is increasingly becoming an important domain in computer vision, due to its wide applications for human action recognition, crowd behavior analysis and ultimately, in smart and intelligent surveillance. Despite the recent advancements in this direction, analyzing motion information is still considered a challenging topic in the computer vision especially due to the typical issues such as occlusions, background clutter, low quality of video and unpredictable camera motion. The fact that the algorithm must run every frame or few frames itself

makes the computation cost too high thereby being an additional burden. This is even more challenging in crowd scenarios, where people movements are governed by complex social interactions, group and individual dynamics, size and context of the crowd scenario.

In case of multiple entities, such as many people in the frame, detecting objects and tracking their trajectories to understand relationships becomes very difficult. In such a case, the video can be analyzed by using Social Force Model [28] approach. This is a holistic approach to capture the dynamics of the crowd behaviour. Social force model describes the behaviour of the crowd as the result of interaction of individuals. Therefore, the abnormal crowd behavior is essentially an eccentric state of the crowd interactions.

This model works by calculating the force of each pedestrian as he/she changes his/her velocity.

$$F = m_i \, (dv_i/dt)$$

, where pedestrian i with mass m_i, changes his/her velocity v_i

Social force model considers the effect of panic where herding behaviors appear in events like escaping from hazardous incident. Velocities are calculated using Particle Advection method. Particles move with the average velocity of their neighborhood; this resembles the collective velocity of a group of people in the crowd. Sudden changes in velocities of everyone creates a high social force, which is the key to analyzing abnormal behaviors.

6.6 Alert System

The main aim of the project is to notify the user if any threat gets predicted by the system. Proposed solutions included the use of Desktop Notifier in python or notify-run package that can be installed easily on the device. OneSignal tool was also proposed that provides a simple interface to push notifications. They also provide a Rest API, which is used to send notifications. The selected solution for making a reliable notification system was Push Bullet. The project uses Push bullet app for notifying users of any possible threats. Push bullet connects every device using APIs and can be configured in different languages such as Python.

The user selects the modules which should be running at the time such as Face recognition, firearm detection etc. Whenever the system detects any unknown person after facial recognition and predicts the person as a threat, a notification is sent to the user's device instantly. It also sends the frame snapshot to the user in which the unknown person is detected. This allows the user to also save the snapshot to google drive and drop box if required further.

```
with open("frame_captured.jpg", "rb") as pic:
        file = pb.upload_file(pic, "picture.jpg")

push = pb.push_file(**file)
```

Benefits for using Pushbullet for the project:

1. Send and receive text as well as visual messages on any device configured.
2. Receive and dismiss notifications on the devices from the app
3. Sync the clipboard across all the devices in a family
4. End to end encryption to ensure privacy. This can be activated by specifying the encryption key.

```
from pushbullet import Pushbullet
pb = Pushbullet(api_key, "Encryption password")
```

5. Group Texting is also possible so that each member of a group is aware of the threats around there home
6. Reliably sync outgoing text messages with the device, to ensure that the messages are sent as early as possible (even if they can’t be sent instantly)
7. Timely updates so that the app is reliable and efficient

7. CODES AND STANDARDS

7.1 Wi-Fi (IEEE 802.11)

Wi-Fi is a common and well known WLAN technology for the networking of devices and is based upon the IEEE 802.11 standards. This standard defines an over-the-air interface between say, a wireless device and a base station or even between more than once wireless devices. There are a number of specifications in the 802.11 family (a/b/g), the major difference being in the frequency band, channel bandwidth and transmission rates.

Feature	Wifi (802.11a)	Wifi (802.11b)	Wifi (802.11g)
Frequency Band	5 GHz	2.4 GHz	2.4 GHz
Transmission Rate	50 Mbps	10 Mbps	20Mbps
Primary Application	WLAN	WLAN	WLAN

Table 4: IEEE 802.11 family standards

7.2 USB

USB stands for Universal Standard Bus, and is an industry standard for establishing specifications not only for cables, connector and protocols, but also for establishing connection, smooth communication and even maintaining power supply between devices like computers, laptops, etc. and their peripheral devices. USB has kept pace with technology and the standard has been updated seeing USB 1, USB 2, USB 3 and

the latest USB 3.1 with each successive standard improving and refining the performance. USB still maintains its position in the market and its use has become very widespread since its introduction.

7.3 HTTPS

HTTPS is an extension of HTTP used for the purpose of secure communication over a computer network, and is immensely used on the internet, especially for online transactions that must be carried out in a secure manner. The protocol for communication purpose is also encrypted using Transport Layer Security. The principal motivation for HTTPS is authentication of the accessed website and also the protection of privacy while maintaining the integrity of the data being exchanged. The live video stream is accessible to the user on his browser through the https protocol.

7.4 SSH

Secure Shell is a cryptographic network protocol for operating network services securely over an unsecured network. The security is ensured using public-key cryptography which authenticates the user. There are several ways to carry out SSH: one is to generate the public-private keys manually and the other one is to generate them automatically then use a password. It runs on a client-server paradigm and is used to display on a remote client while maintaining connection with the machine on which the actual session runs.

8. CONSTRAINTS AND ALTERNATIVES

8.1 Design Constraints

The project was designed while taking into consideration the following constraints -

- The live feed should be made available to the user in a secure manner to maintain confidentiality.
- Since multiple algorithms would be running simultaneously it is necessary to optimize them and implement over the GPU for faster computation.
- Processing every frame of the live video of around 30 frames per second is compute intensive and transmitting the same live video over Wi-Fi is another challenge.
- All devices along with camera must be connected physically through cables as in the classic Closed Circuit TV system or over Wi-Fi at all times.

8.2Component Constraints

- The Raspberry Pi camera used is a 5 MP camera with 1080p resolution but works on low fps. Although the above is used for demonstration, it can be replaced by any other camera with better resolution and features or even pre-installed ones in home or offices.
- Since the project works in real time and has complex algorithms the more powerful the GPU, the better overall time and delays. The current GPU being used is a single 6 GB Nvidia GTX 1060 which is average. However using more than one Nvidia Titan V Graphic card would be a huge boost, but the price of only one such card is 4 Lakh INR.

- Similar is the case with the CPU being used which is Intel Core i5 7th Generation. However the favoured Intel Core i9 X-series cost around 1 Lakh INR which exceeds the budget as the base ideology is for the system to be accessible to individuals for their home security as well and not just large organizations.

9. SCHEDULE, TASKS AND MILESTONES

9.1 Timeline

Figure 11: Task Schedules

9.2 Work Breakdown Structure (WBS)

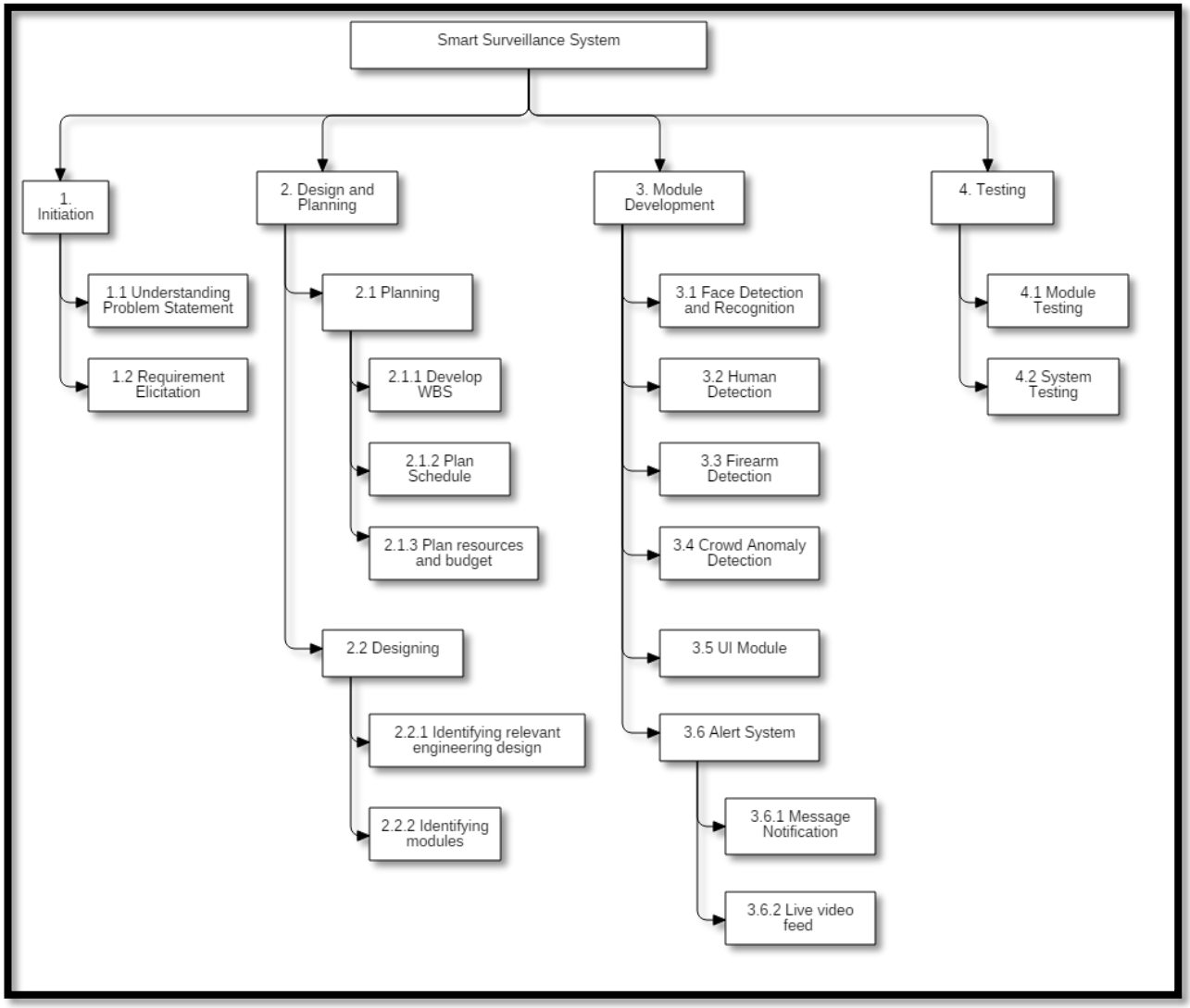

Figure 12: Work Breakdown Structure (WBS)

10. PROJECT DEMONSTRATION

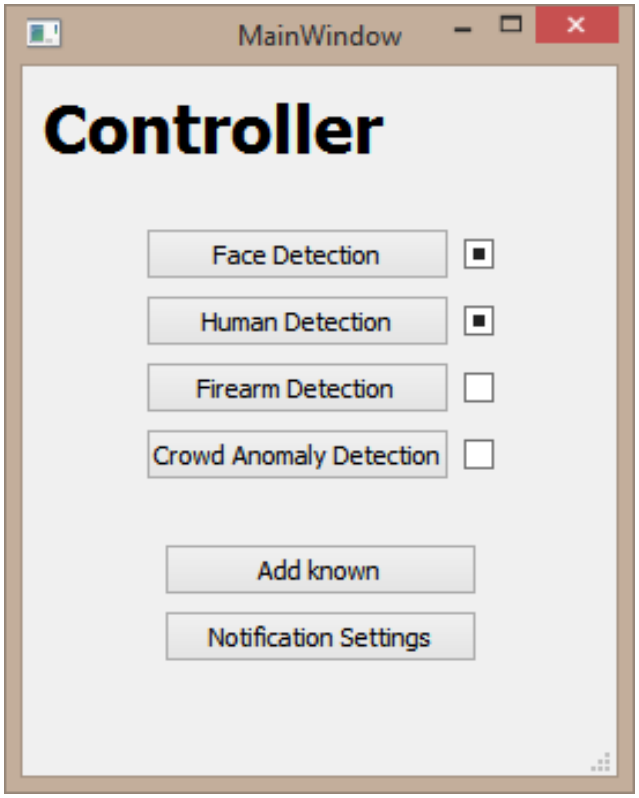

Figure 13: User Interface

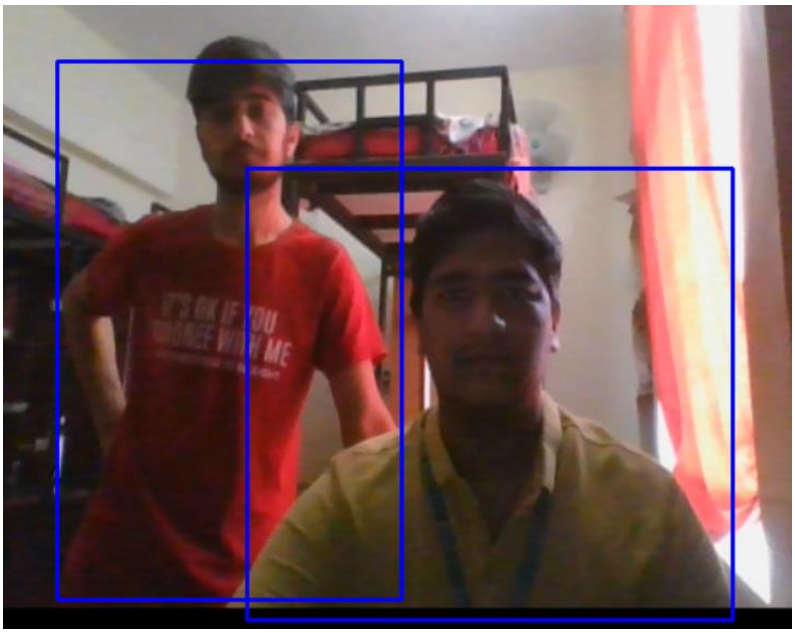

Figure 14: Human Detection

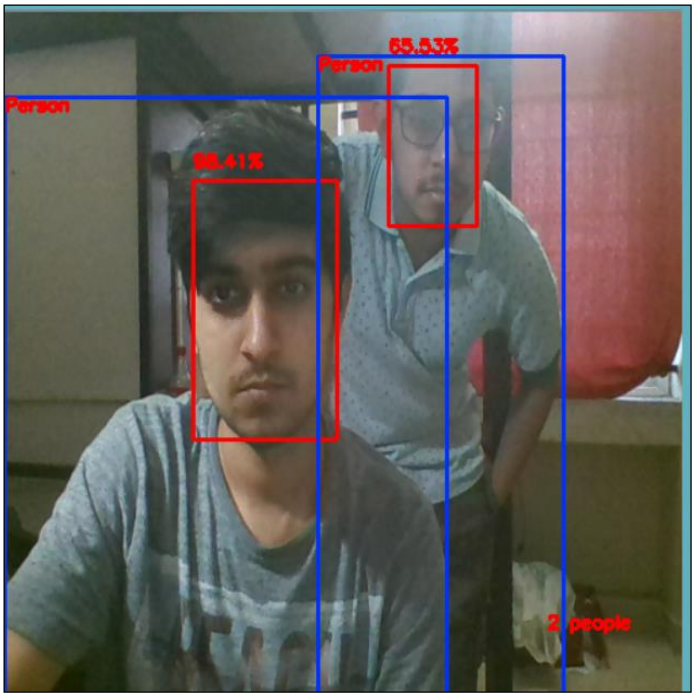

Figure 15: Face and Human Detection

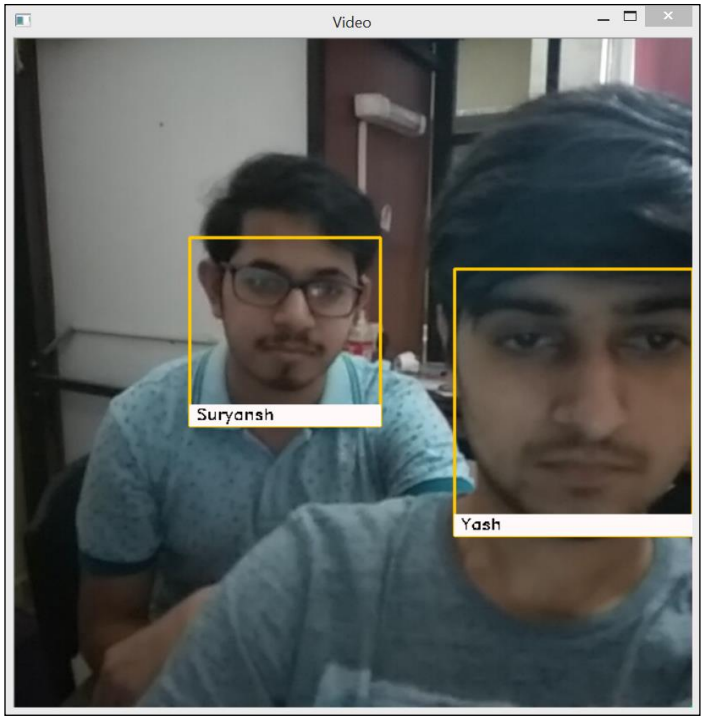

Figure 16: Face Recognition on Live video feed

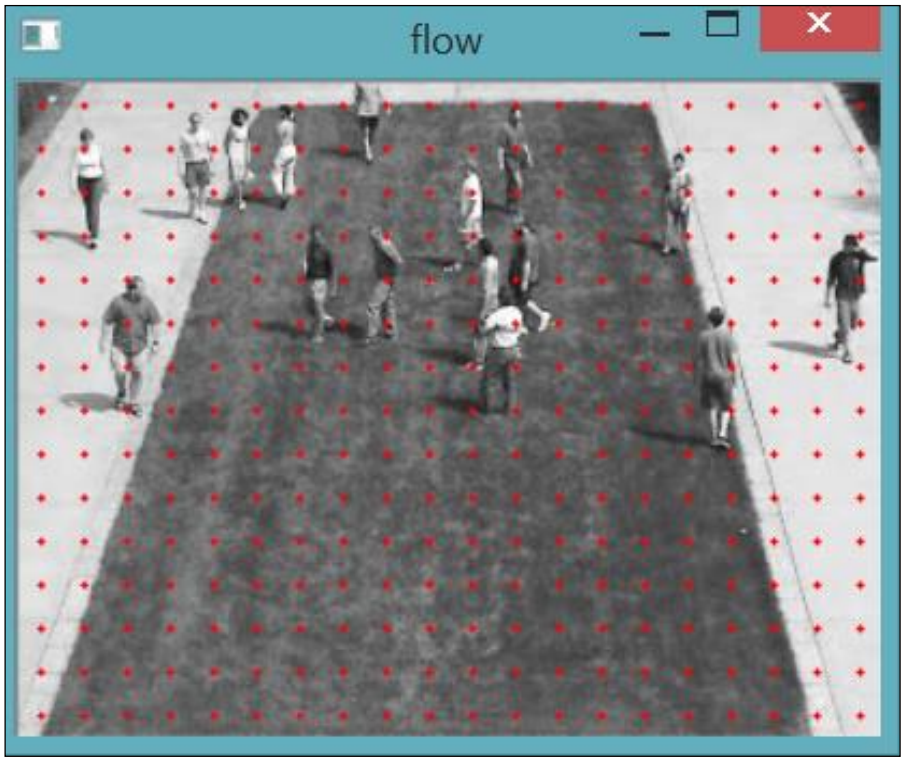

Figure 17: Normal Crowd Activity (1)

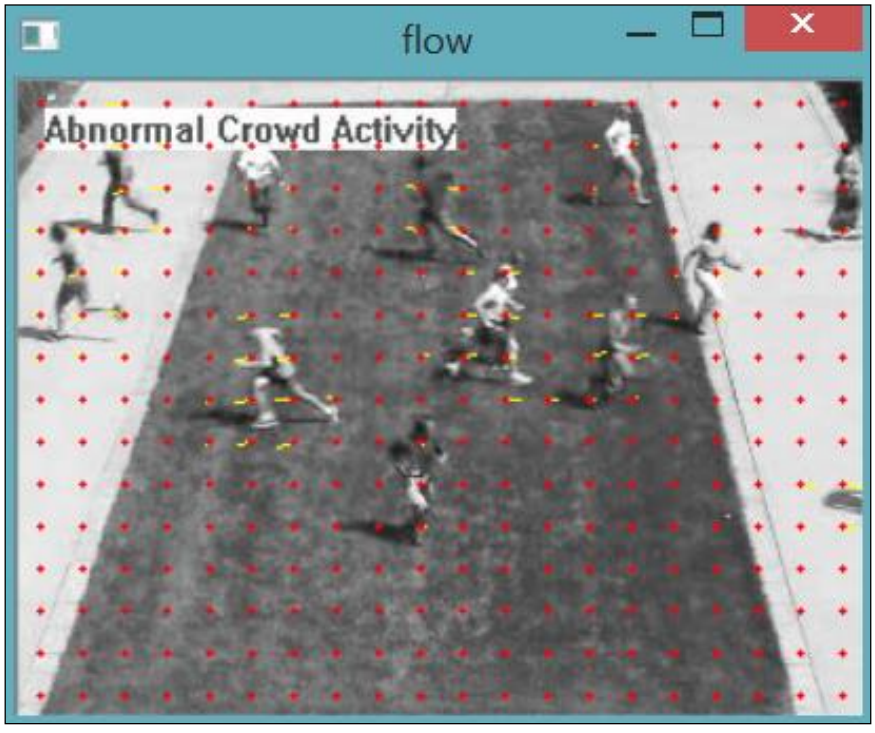

Figure 18: Abnormal Crowd Activity (1)

Figure 19: Normal Crowd Activity (2)

Figure 8: Abnormal Crowd Activity (2)

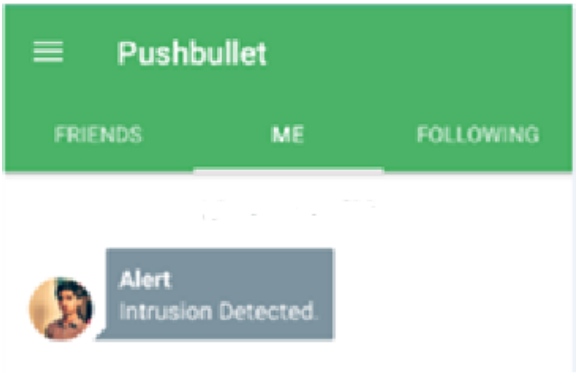

Figure 21: Pushbullet Notification

Figure 22: Firearm Detection (1)

Figure 23: Firearm Detection (2)

11. COST AND MARKET ANALYSIS

11.1 Market Analysis

Surveillance and authentication are one of the most important security aspects in various fields such as banking sectors, military, and even personal security. Due to exponential rise in burglary and theft activities, surveillance systems are proving to be a great source of security. Security systems such as CCTV have proven to be hugely popular for security purposes due to their cost efficient nature and easy maintenance. Fig 1 [20] clearly illustrates the CCTV camera unit sales in India. From 89 Lakh units in 2014, the number has jumped to 217 Lakh units in 2018 with around 181 Lakh of those being Analog cameras and the rest 36 Lakh units being digital ones. The jump from 2014 to 2018 is around 2.5 times and is a good example of how serious the matter of security is being taken in India especially due to cheap prices of such cameras. The only issue with the above is the fact that while the cameras are being installed as units, no system encapsulating everything together, as such is not available. Hence these units remain passive and are limited to providing live feed or storing the video data.

The project designed is a full-fledged surveillance system with total cost of the system rivaling such individual camera units. The fact that the system provides the flexibility to use the already installed cameras would make it more accepting to not only newer customers but also to those having just the camera units alone. By ensuring excellent quality as well as justified and flexible cost with easy installation, the project aims to supersede the existing passive systems while ensuring market acceptability.

11.2 Cost Analysis

Sr. No.	Component	Necessity	Cost
1	Raspberry Pi 3	Needed	₹ 2900
2	5 MP Camera Module	Needed	₹ 590
3	Raspberry Pi Case	Optional	₹ 300
4	Camera Module Case	Optional	₹ 200

Table 5: Cost Analysis

Total Hardware Cost per Unit = ₹ 3990

Raspberry Pi 3 and the respective camera module being used is portable, small and compact and hence ideal for demonstration purpose. It is also being used foe the sole purpose of smooth video transmission with no computation as such on the Raspberry Pi. In real life scenario, this can be replaced with more specialized CCTV cameras the price of which ranges from ₹ 2000 to ₹ 20,000 based on needs and features like IR based night vision, zoom, display resolution, weather proof, etc. Hence based on user needs, the hardware cost can be variable depending largely on the type and number of cameras chosen for surveillance purpose.

12. CONCLUSION

12.1 How the project aims to supersede the current CCTV systems

Many cities still face crimes and antisocial behaviour problems like fights, breaking and entering shops, vandalism, etc. These places tend to have video cameras already installed but the issues is that the usage is limited to watching the incident at a later time when it has already happened. The surveillance systems are too passive and hence are not able to automatically analyse the live video data. This is where our current systems fail. We are very well equipped to catch the suspect later but by then it might be too late as the deed would be done. The designed Smart Surveillance System quenches the increasing demand for active and automated surveillance systems with the main aim to alert the concerned authority in case of any abnormal or dangerous situation. By implementing algorithms efficiently and further using GPU, the system works as intended in real time with instantaneous alerts. The Face Recognition based authentication module along with crowd behaviour and analysis module are a cherry on top.

Since the amount of video data collected everyday by such surveillance cameras increase, so does the need for automated smart systems to detect and identify suspicious activities performed by people or objects. The advantage of such analysis is that it could detect unusual and suspicious events like people converging or running erratically and immediately alert the concerned authority. The system could also be used to identify people and notify unwanted or unexpected presence.

12.2 Target Market

The CCTV camera unit sales in India, from 89 Lakh units in 2014 has soared to 217 Lakh units in 2018 which is around 2.5 times the former and is a good indication of how serious the matter of security is in India. The fact that these CCTV cameras have a cheap price and are easy to maintain further increases the popularity. By taking the above into consideration, the project's total hardware cost rivals those single units of CCTV cameras. Having such low cost and ease of installation allows any person concerned for his/her family's security to be a viable user. The fact that the system allows flexibility with respect to cameras allows organizations to easily switch to our system without reinstalling cameras, i.e., using the already installed pre-placed cameras. Finally, the government could also benefit from the system as real time analysis in important places like heritage sites and monuments could give important alerts thereby helping in preventing untoward incidents that may cause harm.

12.3 Future Work

With the rapid growth in the field of Computer Vision, Machine Learning and IoT and advancements in the field of security and surveillance the project could definitely incorporate many functionalities and modify algorithms for further efficiency and decrease computational time. Unlike currently where the user adds known and threat faces, by scaling the project, a common database for say criminals could be developed. By updating this global database, each user could be alerted accordingly if a past criminal is spotted on the respective surveillance camera. For example, security personnel outside localities or shopping malls could be alerted about a person previously found guilty of car theft. Having this information would give an edge to the security personnel and prepare him in case the person tries to steal a car from the

parking lot. This feature could even help the police in catching criminals on the run if at all they come in sight of government cameras installed in public places like parks, near monuments, etc. The police authority could also immediately update the database for missing people with a relevant photograph of the missing person. This could be made available to the public and modified equivalently in the global database such that the whereabouts of the missing person could be notified immediately if at all identified anywhere by any camera that is part of the system.

13. REFERENCES

1. Membrey, Peter, and David Hows. Learn Raspberry Pi with Linux. Apress, 2013.
2. Kumbhar, Deepak S., Shubhangi M. Taur, and Shubhangi S. Bhatambrekar. "IoT Based Home Security System Using Raspberry Pi-3." (2018).
3. Pragati Ukey, Anita Shinde,Sneha Kasrung, Satish Kamble, Jidnyesh Kadu, “Development of Smart Home security system using Raspberry Pi”
4. Mane, Sneha S., and Girish R. Talmale. "Raspberry Pi Based Security System on IoT Platform."
5. Lin, Kwan-Ho, et al. "An efficient human face indexing scheme using eigenfaces." International Conference on Neural Networks and Signal Processing, 2003. Proceedings of the 2003. Vol. 2. IEEE, 2003.
6. Ahmad, Faizan, Aaima Najam, and Zeeshan Ahmed. "Image-based face detection and recognition:" state of the art"." arXiv preprint arXiv:1302.6379 (2013).
7. Bhavani K, Dhanaraj V, Siddesh N V, Ragav Vijayadev, Uma Rani S, “Real time Face Detection and Recognition in Video Surveillance”
8. Van Wyk, Desmond E., and James Connan. "Development of a real-time face recognition system for access control."
9. Pande, Varun, Khaled M. Elleithy, and Laiali Almazaydeh. "Parallel processing for multi face detection and recognition." (2012).

10. Nakib, Mohammad, et al. "Crime Scene Prediction by Detecting Threatening Objects Using Convolutional Neural Network." 2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2). IEEE, 2018.
11. Grega, Michał, et al. "Automated detection of firearms and knives in a CCTV image." Sensors 16.1 (2016): 47.
12. Olmos, Roberto, Siham Tabik, and Francisco Herrera. "Automatic handgun detection alarm in videos using deep learning." Neurocomputing 275 (2018): 66-72.
13. Olmos, Roberto, Siham Tabik, and Francisco Herrera. "Automatic handgun detection alarm in videos using deep learning." Neurocomputing 275 (2018): 66-72.
14. Rodrigo Fumihiro, “Firearm Detection Using Convolutional Neural Networks”
15. T. Mita, T. Kaneko, O. Hori, Joint Haar-like Features for Face Detection, “Proceedings of the Tenth IEEE International Conference on Computer Vision”, 15505499/05 ©2005 IEEE.
16. T. Ahonen, A. Hadid, M. Peitikainen, Face recognition with local binary patterns. “In Proc. of European Conference of Computer Vision”, 2004
17. Swathi, H. Y., G. Shivakumar, and H. S. Mohana. "Crowd behavior analysis: a survey." 2017 International Conference on Recent Advances in Electronics and Communication Technology (ICRAECT). IEEE, 2017.

18. Joshi, Kinjal Y., and Safvan A. Vohra. "Crowd Behavior Analysis." International Journal of Science and Research, ISSN: 2319-7064.
19. Zitouni, M. Sami, et al. "Advances and trends in visual crowd analysis: A systematic survey and evaluation of crowd modelling techniques." Neurocomputing 186 (2016): 139-159.
20. Rahul Sachitanand, “Sales of surveillance cameras are soaring, raising questions about privacy.” The Economic Times. Accessed 30 October 2018.
21. Taha, Ahmed, et al. "Exploring behavior analysis in video surveillance applications." International Journal of Computer Applications 93.14 (2014).
22. Chaaraoui, Alexandros, et al. "A vision-based system for intelligent monitoring: human behaviour analysis and privacy by context." Sensors 14.5 (2014): 8895-8925.
23. Xu, Honghua, et al. "Movement Human Actions Recognition Based on Machine Learning." International Journal of Online Engineering (iJOE) 14.04 (2018): 193-210.
24. Charara, Nour, et al. "Adabev: Automatic detection of abnormal behavior in video-surveillance." International Conference on Image, Signal and Vision Computing, Oslo, Norway. 2012.
25. Forczmański, Paweł, and Marcin Seweryn. "Surveillance video stream analysis using adaptive background model and object recognition."

International Conference on Computer Vision and Graphics. Springer, Berlin, Heidelberg, 2010.

26. Bobick, Aaron F., and James W. Davis. "The recognition of human movement using temporal templates." IEEE Transactions on Pattern Analysis & Machine Intelligence 3 (2001): 257-267.
27. Karpathy, Andrej, et al. "Large-scale video classification with convolutional neural networks." Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. 2014.
28. Mehran, Ramin, Alexis Oyama, and Mubarak Shah. "Abnormal crowd behavior detection using social force model." 2009 IEEE Conference on Computer Vision and Pattern Recognition. IEEE, 2009.

APPENDIX 1: CODE SNIPPETS

```python
import numpy as np
import tensorflow as tf
import cv2
import time
from imutils.object_detection import non_max_suppression


class DetectorAPI:
    def __init__(self, path_to_ckpt):
        self.path_to_ckpt = path_to_ckpt

        self.detection_graph = tf.Graph()
        with self.detection_graph.as_default():
            od_graph_def = tf.GraphDef()
            with tf.gfile.GFile(self.path_to_ckpt, 'rb') as fid:
                serialized_graph = fid.read()
                od_graph_def.ParseFromString(serialized_graph)
                tf.import_graph_def(od_graph_def, name='')

        self.default_graph = self.detection_graph.as_default()
        self.sess = tf.Session(graph=self.detection_graph)

        # Definite input and output Tensors for detection_graph
        self.image_tensor = self.detection_graph.get_tensor_by_name('image_tensor:0')
        # Each box represents a part of the image where a particular object was detected.
        self.detection_boxes = self.detection_graph.get_tensor_by_name('detection_boxes:0')
        # Each score represent how level of confidence for each of the objects.
        # Score is shown on the result image, together with the class label.
        self.detection_scores = self.detection_graph.get_tensor_by_name('detection_scores:0')
        self.detection_classes = self.detection_graph.get_tensor_by_name('detection_classes:0')
        self.num_detections = self.detection_graph.get_tensor_by_name('num_detections:0')

    def processFrame(self, image):
        # Expand dimensions since the trained_model expects images to have shape: [1, None, None, 3]
        image_np_expanded = np.expand_dims(image, axis=0)
        # Actual detection.
        start_time = time.time()
        (boxes, scores, classes, num) = self.sess.run(
            [self.detection_boxes, self.detection_scores, self.detection_classes, self.num_detections],
            feed_dict={self.image_tensor: image_np_expanded})
        end_time = time.time()
```

Figure 24: Face and Human detection (1)

```
42
43          print("Elapsed Time:", end_time-start_time)
44
45          im_height, im_width,_ = image.shape
46          boxes_list = [None for i in range(boxes.shape[1])]
47          for i in range(boxes.shape[1]):
48              boxes_list[i] = (int(boxes[0,i,0] * im_height),
49                          int(boxes[0,i,1]*im_width),
50                          int(boxes[0,i,2] * im_height),
51                          int(boxes[0,i,3]*im_width))
52
53          return boxes_list, scores[0].tolist(), [int(x) for x in classes[0].tolist()], int(num[0])
54
55      def close(self):
56          self.sess.close()
57          self.default_graph.close()
58
59  if __name__ == "__main__":
60      #path of the human detection model
61      model_path = "C:/Users/Yashasvi/Final project/ssd_mobilenet/frozen_inference_graph.pb"
62
63      #path of the face detection model
64      net = cv2.dnn.readNetFromCaffe("deploy.prototxt.txt","res10_300x300_ssd_iter_140000.caffemodel")
65      odapi = DetectorAPI(path_to_ckpt=model_path)
66      threshold = 0.6
67
68      #start video capturing
69      cap = cv2.VideoCapture(0)
70      time.sleep(2.0)
71
72      while True:
73          r, img = cap.read()
74          img = cv2.resize(img, (500, 500))
75
76          boxes, scores, classes, num = odapi.processFrame(img)
77
```

Figure 25: Face and Human detection (2)

```
        #facedetection-----------------------------------------------

        (h, w) = img.shape[:2]
        blob = cv2.dnn.blobFromImage(cv2.resize(img, (300, 300)), 1.0,
            (300, 300), (104.0, 177.0, 123.0))
        net.setInput(blob)
        detections = net.forward()
        for i in range(0, detections.shape[2]):
            # extract the confidence (i.e., probability) associated with the
            # prediction
            confidence = detections[0, 0, i, 2]

            # filter out weak detections by ensuring the `confidence` is
            # greater than the minimum confidence
            if confidence < 0.3:
                continue

            # compute the (x, y)-coordinates of the bounding box for the
            # object
            box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
            (startX, startY, endX, endY) = box.astype("int")

            # draw the bounding box of the face along with the associated
            # probability
            text = "{:.2f}%".format(confidence * 100)
            y = startY - 10 if startY - 10 > 10 else startY + 10
            cv2.rectangle(img, (startX, startY), (endX, endY),
                (0, 0, 255), 2)
            cv2.putText(img, text, (startX, y),
                cv2.FONT_HERSHEY_SIMPLEX, 0.45, (0, 0, 255), 2)

        #endfacedetection-------------------------------------------
```

Figure 25: Face and Human detection (3)

```
        # Visualization of the results of a detection.
        count_person=0
        for i in range(len(boxes)):
            # Class 1 represents human
            if classes[i] == 1 and scores[i] > threshold:
                count_person=count_person+1
                box = boxes[i]
                cv2.rectangle(img,(box[1],box[0]),(box[3],box[2]),(255,50,0),2)
                cv2.putText(img, "Person", (box[1],box[0]+10),cv2.FONT_HERSHEY_SIMPLEX, 0.45, (0, 0, 255), 2)

        #show the number of people in the frame by counting the number of boxes of class 1
        cv2.putText(img, str(count_person) + " people",(400,450) ,cv2.FONT_HERSHEY_SIMPLEX, 0.45, (0, 0, 255), 2)
        cv2.imshow("Frame", img)

        key = cv2.waitKey(1) & 0xFF
        if key == ord('q'):
            break
    cap.release()
    cv2.destroyAllWindows()
```

Figure 28: Face and Human detection (4)

```python
import face_recognition
import dlib
import itertools
import sys
import cv2
import os

known_names = []
known_encodings = []
for root,dirs,files in os.walk("KnownFaces"):
    for filename in files:
        #print (filename)
        img = face_recognition.load_image_file("KnownFaces/"+filename)
        encodings = face_recognition.face_encodings(img)
        basename = os.path.splitext(os.path.basename(filename))[0]
        known_names.append(basename)
        known_encodings.append(encodings[0])


#video_capture = cv2.VideoCapture(0)
video_capture=cv2.VideoCapture("http://192.168.43.22:9000/stream/video.mjpeg")
face_locations = []
face_encodings = []
face_names = []
process_this_frame = True

while True:
    # Grab a single frame of video
    ret, frame = video_capture.read()

    # Resize frame
    small_frame = cv2.resize(frame, (0, 0), fx=.2, fy=.2)

    # Convert the image from BGR color (which OpenCV uses) to RGB color (which face_recognition uses)
    rgb_small_frame = small_frame[:, :, ::-1]
```

Figure 27: Face Recognition (1)

```
    # Only process every other frame of video to save time
    if process_this_frame:
        # Find all the faces and face encodings in the current frame of video
        face_locations = face_recognition.face_locations(rgb_small_frame)
        face_encodings = face_recognition.face_encodings(rgb_small_frame, face_locations)

        face_names = []
        for face_encoding in face_encodings:
            # See if the face is a match for the known face(s) SVM
            matches = face_recognition.compare_faces(known_encodings, face_encoding,tolerance=.6)

            name = "Unknown"

            # If a match was found in known_face_encodings, just use the first one.
            if True in matches:
                first_match_index = matches.index(True)
                name = known_names[first_match_index]

            face_names.append(name)

    process_this_frame = not process_this_frame


    # Display the results
    for (top, right, bottom, left), name in zip(face_locations, face_names):
        # Scale back up face locations since the frame we detected in was scaled to 1/4 size
        top *= 5
        right *= 5
        bottom *= 5
        left *= 5

        # Draw a box around the face
        cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255), 2)

        # Draw a label with a name below the face
        cv2.rectangle(frame, (left, bottom - 35), (right, bottom), (0, 0, 255), cv2.FILLED)
        font = cv2.FONT_HERSHEY_DUPLEX
        cv2.putText(frame, name, (left + 6, bottom - 6), font, 1.0, (255, 255, 255), 1)
```

Figure 29: Face Recognition (2)

```
    # Display the resulting image
    cv2.imshow('Video', frame)

    # Hit 'q' on the keyboard to quit!
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# Release handle to the webcam
video_capture.release()
cv2.destroyAllWindows()
```

Figure 30: Face Recognition (3)