CHAPTER 1
INTRODUCTION

1.1 Overview

A human intrusion detection system is designed to detect an unauthorized entry into a building, shops or a protected area and deny such unauthorized access to protect personnel and property from damage or harm. Security systems are mainly used in inhabited, commercial, manufacturing, and army properties for protection against burglary or property damage, as well as private safety against intruders. This system provides proper detection of intruder and provides security. By using this system, we can reduce robbery by detecting the intruder. So we can respond quickly such that no harm takes place in our home. In the designed system, the camera is kept outside the room and the continuous video is captured by the camera. We designed our system in such a way that as soon as intruder enters the room, the processing takes place at the client. All of which could be enabled through the software or dedicated apps, and capture the intruder image.

1.2 Problem Statement

Projects which are developed in teams sometimes fail because of lack of communication between team members and being unaware of the changes made in the software code which leads to confusion and more than one version available for same project. Therefore, there is a need of an eloquent way of communication in order to develop a project successfully. This is not achieved on a single platform so far. There are various modes of communications now-a-days but they do not provide communication for this specific purpose.

For this problem to be solved, an android application is to be developed to help project leader assign task to his fellow team members and track the updates made in program codes by each developer from team. Creation and assignment of projects for various project teams must be done. If there is a new team member he must be added to the existing on-going project. Project details should be private to particular project team.

1.3 Objective

It got nice **GUI** and every button is supported by using beautiful icon.

- **Monitor** – allow to monitor the motion of intruder and taking images saving it to system of administrator
- **In Out** – Finds who entered and gone from room.
- **Record** – Recording of workstation environment.
- **Exit Module** – It will help administrator to exit the application

1.4 Features

- Monitoring the motion
- Taking look on movements of objects
- 24x7 Recording of workstation environment
- Motion Identification

CHAPTER 2
LITERATURE SURVEY

Intruder Detection system, is a system which focus on the aspects related to secured environment of workstation by ensuring the detection of the objects and saving it into the system of project taken by a software administrator. It helps the administrator to manage the workstation security. This system plays an important role in the making of secured workstation and its management. A Hybrid Intrusion Detection System by leveraging the benefits of Machine Learning techniques to build a system which detects the intrusion and alerts the respective network administrator. This can be extended from Intrusion to breach detection as well. The developed system analyses and predicts the behavior of users which in turn classifies as an anomaly or a normal behavior There are many existing intruder detection systems for surveillance and security systems. They either need human intervention to detect intruder or need a long work for the installation and there is also a possibility of false alarms. In Surveillance system, the intruder is detected using the video recordable cameras which are already installed and stored in an external storage disk. But in this system needs a huge investment for installing, storing and monitoring the activities. Thought the occurrence of activity is less, the footage is to be deleted, after examining by the owner. This leads to waste of manhour as he should watch the complete video and this may lead to missing small details during analysis. We couldn't monitor the streaming video directly. The Automatic robbery/theft detection using smart surveillance in banks Presumably Presumably, the most priceless asset of a city is its inhabitants and their things. An intruder detection system is the contemporary metropolitan idea which is totally essential for inhabitants of a framework to have a quality life. This intruder detection system is used to detect an intruder and generate the alert to the authorized person. Based upon this, the incident responder can investigate the issue and take the necessary action at the instant. Here in the system the PIR sensor senses motion of human beings and it is captured by using the Pi camera. The extracted face is recognized and it is then sent to the Raspberry Pi through HDMI cable. This system is well executed in Open CV- Python. The captured image is compared with the saved image of the authorized person in the database. The system can distinguish an authorized and unauthorized person by comparing. If it is found to be an unauthorized person, then system sends the recognized image to the owner whose authorized number we feed in the system, through email using Wi-Fi technology.

CHAPTER 3

IMPLEMENTATION

3.1 Technical Design

Language Used:

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. Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective. We have used **Python language** as it is very new and also comes with so many features like we can do Machine Learning, Computer Vision and Also make GUI application with ease.

Reasons for Selecting this language:

1 – Short and Concise Language.
2 – Easy to Learn and use.
3 – Good Technical support over Internet
4 – Many Packages for different tasks.
5 – Run on Any Platform.
6 – Modern and OOP language

Some specific features of Python are as follows:

- an interpreted (as opposed to compiled) language. Contrary to e.g., C or Fortran, one does not compile Python code before executing it. In addition, Python can be used interactively: many Python interpreters are available, from which commands and scripts can be executed.
- a free software released under an **open-source** license: Python can be used and distributed free of charge, even for building commercial software.
- multi-platform: Python is available for all major operating systems, Windows, Linux/Unix, MacOS X, most likely your mobile phone OS, etc.

- a very readable language with clear non-verbose syntax
- a language for which a large variety of high-quality packages are available for various applications, from web frameworks to scientific computing.
- a language very easy to interface with other languages, in particular C and C++.
- Some other features of the language are illustrated just below. For example, Python is an object-oriented language, with dynamic typing (the same variable can contain objects of different types during the course of a program).

3.2. Modules

It got nice **GUI** and every button is supported by using beautiful icon.

- **Monitor** – allow to monitor the motion of intruder and taking images saving it to system of administrator
- **In Out** – Finds who entered and gone from room.
- **Record** – Recording of workstation environment.
- **Exit Module** – It will help administrator to exit the application

3.3. Implementation

- **Face Detection**, a widely popular subject with a huge range of applications. Modern day Smartphones and Laptops come with in-built face detection software's, which can authenticate the identity of the user. There are numerous apps that can capture, detect and process a face in real time, can identify the age and the gender of the user, and also can apply some really cool filters. The list is not limited to these mobile apps, as Face Detection also has a wide range of applications in Surveillance, Security and Biometrics as well. But the origin of its Success stories dates back to *2001,* when *Viola and Jones* proposed the first ever Object Detection Framework for Real Time Face Detection in Video Footage. This article is about taking a gentle look on the **Viola-Jones Face Detection Technique**, popularly known as **Haar Cascades**, and exploring some of the interesting concepts proposed by them. This piece of work was done long before the Deep Learning Era had even started. But it's an excellent work in comparison to the powerful models that can be built with the modern day Deep Learning Techniques. The algorithm is still found to be used almost everywhere. It has fully trained models available on GitHub. It's fast. It's pretty accurate (at least when I try it).

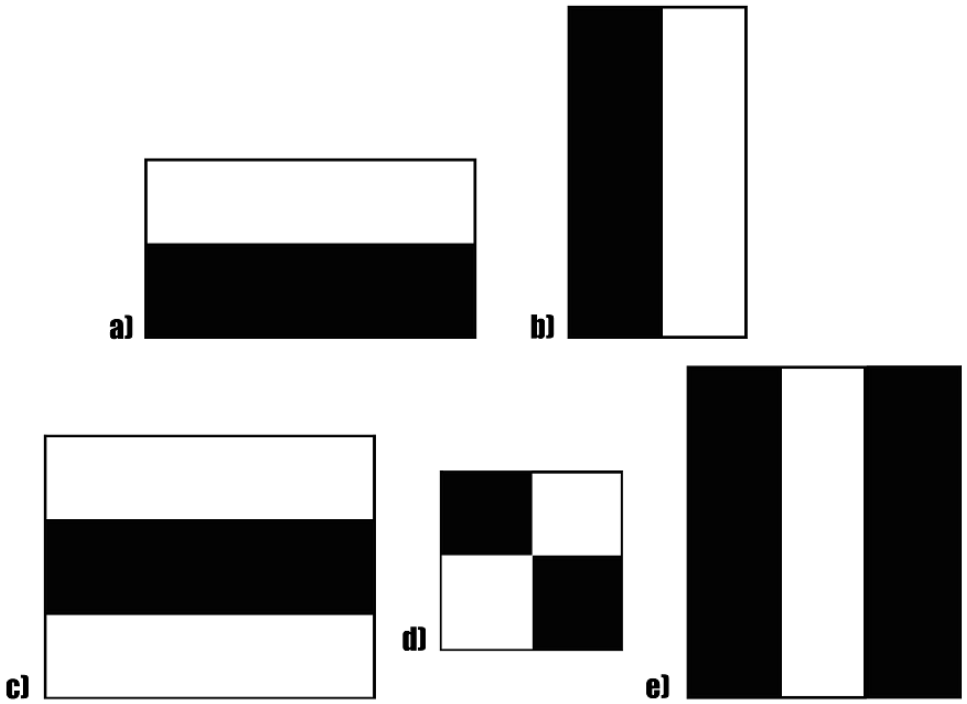
- According to Wikipedia… **Woody Bledsoe, Helen Chan Wolf, and Charles Bisson were the first ones to do the first ever Face Detection on a Computer back in the 1960s**. A person had to manually pinpoint the coordinates of facial features such as the pupil centres, the inside and outside corner of eyes, and the widows peak in the hairline. The coordinates were used to calculate 20 distances, including the width of the mouth and of the eyes. A human could process about 40 pictures an hour in this manner and so build a database of the computed distances. A computer would then automatically compare the distances for each photograph, calculate the difference between the distances and return the closed records as a possible match.

- So what is Haar Cascade? **It is an Object Detection Algorithm used to identify faces in an image or a real time video**. The algorithm uses edge or line detection features proposed by Viola and Jones in their research paper "Rapid Object Detection using a Boosted Cascade of

Simple Features" published in 2001. The algorithm is given a lot of positive images consisting of faces, and a lot of negative images not consisting of any face to train on them. The model created from this training is available at the OpenCV GitHub repository https://github.com/opencv/opencv/tree/master/data/haarcascades.

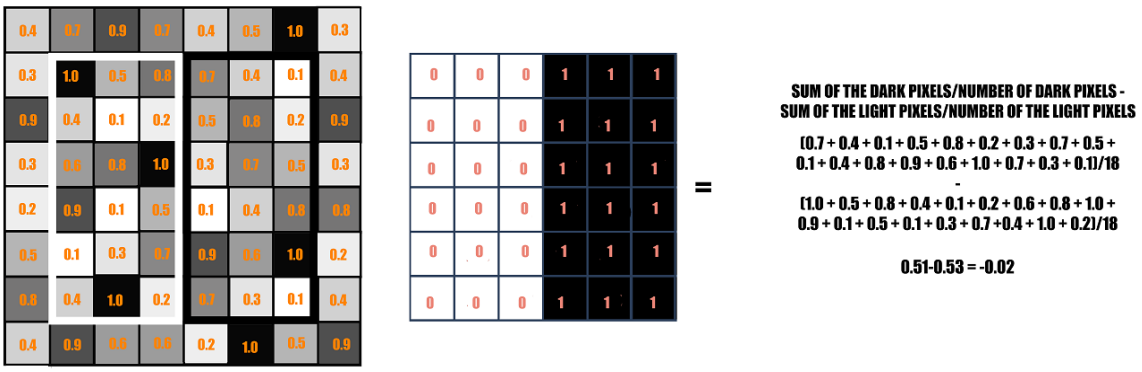
- The repository has the models stored in XML files, and can be read with the OpenCV methods. These include models for face detection, eye detection, upper body and lower body detection, license plate detection etc. Below we see some of the concepts proposed by Viola and Jones in their research.

- **Features**



- Fig. A sample of Haar features used in the Original Research Paper published by Viola and Jones.

- The first contribution to the research was the introduction of the haar features shown above. These features on the image makes it easy to find out the edges or the lines in the image, or to pick areas where there is a sudden change in the intensities of the pixels.



- Fig. The rectangle on the left is a sample representation of an image with pixel values 0.0 to 1.0. The rectangle at the center is a haar kernel which has all the light pixels on the left and all the dark pixels on the right. The haar calculation is done by finding out the difference of the average of the pixel values at the darker region and the average of the pixel values at the lighter region. If the difference is close to 1, then there is an edge detected by the haar feature.

- A sample calculation of Haar value from a rectangular image section has been shown here. The darker areas in the haar feature are pixels with values 1, and the lighter areas are pixels with values 0. Each of these is responsible for finding out one particular feature in the image. Such as an edge, a line or any structure in the image where there is a sudden change of intensities. For ex. in the image above, the haar feature can detect a vertical edge with darker pixels at its right and lighter pixels at its left.

- The objective here is to find out **the sum of all the image pixels lying in the darker area of the haar feature** and **the sum of all the image pixels lying in the lighter area of the haar feature**. And then find out their difference. Now if the image has an edge separating dark pixels on the right and light pixels on the left, then the haar value will be closer to 1. That means, we say that there is an edge detected if the haar value is closer to 1. In the example above, there is no edge as the haar value is far from 1.

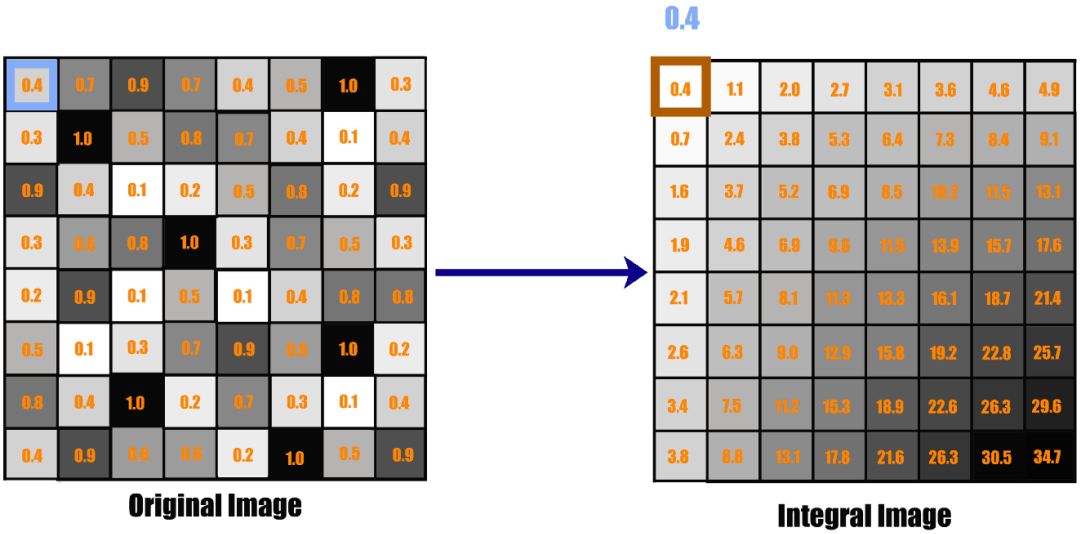
- This is just one representation of a particular haar feature separating a vertical edge. Now there are other haar features as well, which will detect edges in other directions and any other image structures. To detect an edge anywhere in the image, the haar feature needs to traverse the whole image.

- The haar feature continuously traverses from the top left of the image to the bottom right to search for the particular feature. This is just a representation of the whole concept of the haar feature traversal. In its actual work, the haar feature would traverse pixel by pixel in the image. Also all possible sizes of the haar features will be applied.

- Depending on the feature each one is looking for, these are broadly classified into three categories. The first set of **two rectangle features** are responsible for finding out the edges in a

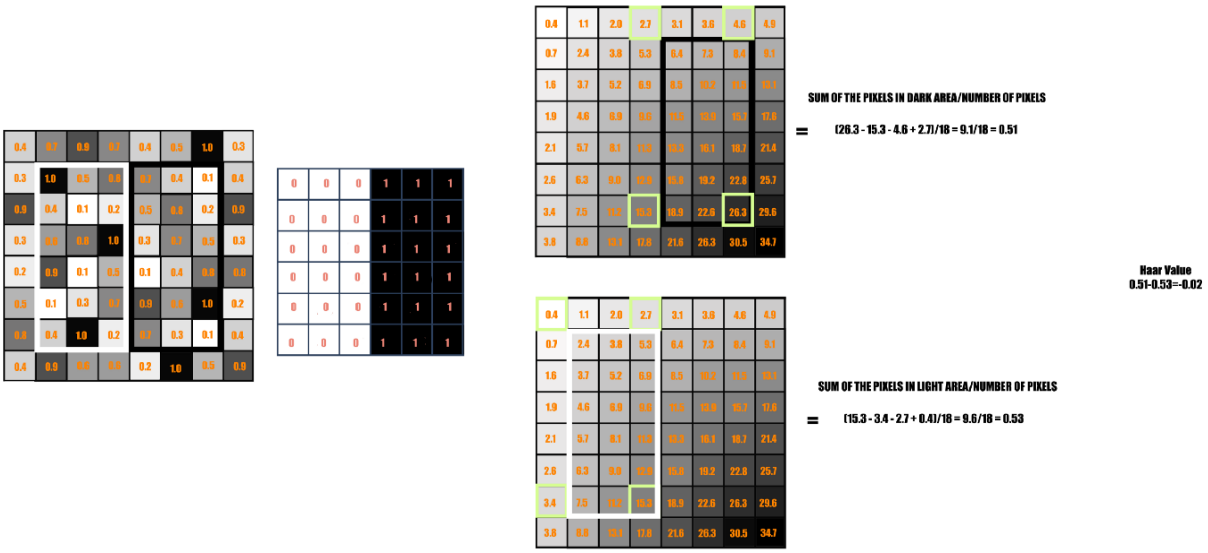
horizontal or in a vertical direction (as shown above). The second set of **three rectangle features** are responsible for finding out if there is a lighter region surrounded by darker regions on either side or vice-versa. The third set of **four rectangle features** are responsible for finding out change of pixel intensities across diagonals.

- Now, the haar features traversal on an image would involve a lot of mathematical calculations. As we can see for a single rectangle on either side, it involves 18 pixel value additions (for a rectangle enclosing 18 pixels). Imagine doing this for the whole image with all sizes of the haar features. This would be a hectic operation even for a high performance machine.



- Fig. The GIF shows the making of an Integral Image. Each pixel in an Integral image is the sum of all the pixels in its left and above.

- To tackle this, they introduced another concept known as **The Integral Image** to perform the same operation. An Integral Image is calculated from the Original Image in such a way that each pixel in this is the sum of all the pixels lying in its left and above in the Original Image. The calculation of a pixel in the Integral Image can be seen in the above GIF. The last pixel at the bottom right corner of the Integral Image will be the sum of all the pixels in the Original Image.



- Fig. Integral Image is used here to calculate the haar value.

- With the Integral Image, only 4 constant value additions are needed each time for any feature size (with respect to the 18 additions earlier). This reduces the time complexity of each addition gradually, as the number of additions does not depend on the number of pixels enclosed anymore.

- In the above image, there is no edge in the vertical direction as the haar value is -0.02, which is very far from 1. Let's see one more example, where there might be an edge present in the image.

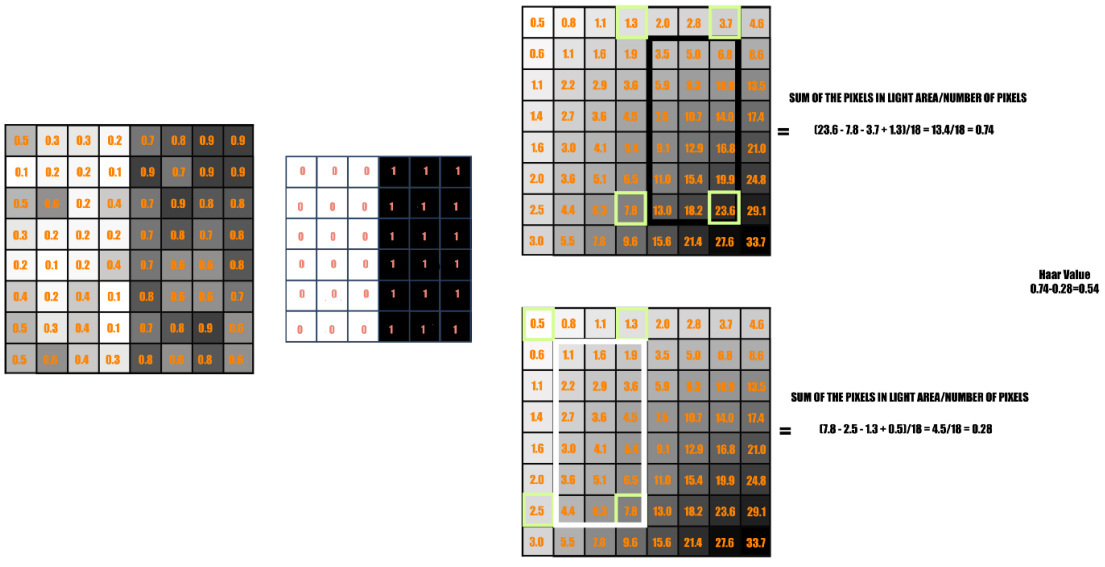


Fig. Haar calculation from Integral Image. This is a case where there is a sudden change of pixel intensities moving vertically from the left towards the right in the image.

- Again repeating the same calculation done above, but this time just to see what Haar value is calculated when there is a sudden change of intensities moving from left to right in a vertical direction. The Haar value here is 0.54, which is closer to 1 in comparison to the case earlier.

Monitor Feature:

This feature is used to find what is the thing which is stolen from the frame which is visible to webcam. Meaning It constantly monitors the frames and checks which object or thing from the frame has been taken away by the thief.

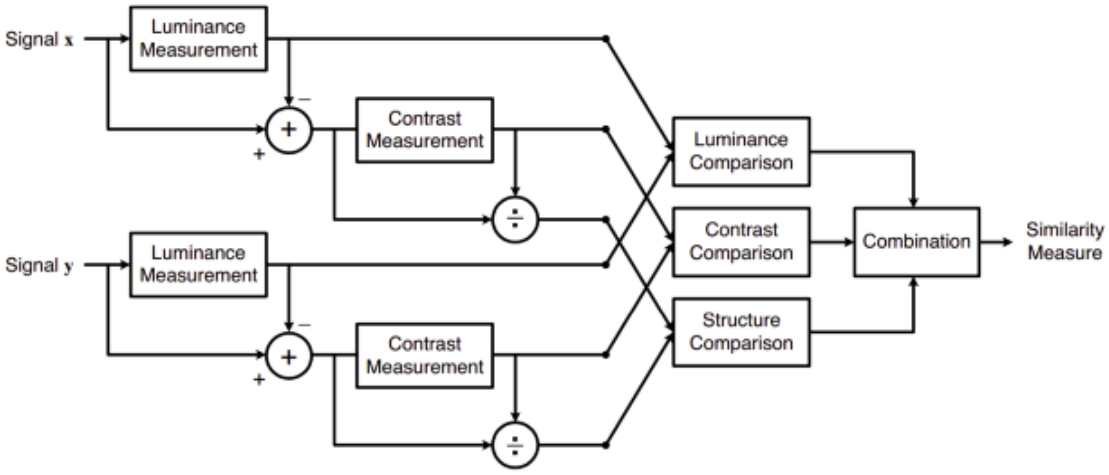
This uses **Structural Similarity** to find the differences in the two frames. The two frames are captured first when noise was not happened and second when noise stopped happening in the frame.

SSIM is used as a metric to measure the *similarity* between two given images. As this technique has been around since 2004, a lot of material exists explaining the theory behind SSIM but very few resources go deep into the details, that too specifically for a gradient-based implementation as SSIM is often used as a loss function.

The Structural Similarity Index (SSIM) metric extracts 3 key features from an image:

- **Luminance**
- **Contrast**
- **Structure**

The comparison between the two images is performed on the basis of these 3 features.



This system calculates the *Structural Similarity Index* between 2 given images which is a value between -1 and +1. *A value of +1* indicates that the 2 given images are **very similar or the same** while a *value of -1* indicates the 2 given images are **very different.** Often these values are adjusted to be in the range [0, 1], where the extremes hold the same meaning.

$$\mu_x = \frac{1}{N}\sum_{i=1}^{N} x_i. \qquad (2)$$

The luminance comparison function $l(\mathbf{x}, \mathbf{y})$ is then a function of μ_x and μ_y.

Luminance: Luminance is measured by *averaging* over all the pixel values. Its denoted by μ (Mu) and the formula is given below,

$$\sigma_x = \left(\frac{1}{N-1}\sum_{i=1}^{N}(x_i - \mu_x)^2\right)^{\frac{1}{2}}. \qquad (4)$$

The contrast comparison $c(\mathbf{x}, \mathbf{y})$ is then the comparison of σ_x and σ_y.

Luminance: Luminance is measured by *averaging* over all the pixel values. Its denoted by μ (Mu) and the formula is given below,

Structure: The structural comparison is done by using a consolidated formula (more on that later) but in essence, we divide the input signal with its *standard deviation* so that the result has unit standard deviation which allows for a more robust comparison.

$$(\mathbf{x} - \mu_x)/\sigma_x$$

Luckily , thanks to scrimmage package in python we don't have to replicate all this mathematical calculation in python since scrimmage has pre build feature that d o all of these tasks for us with just calling its in-built function.

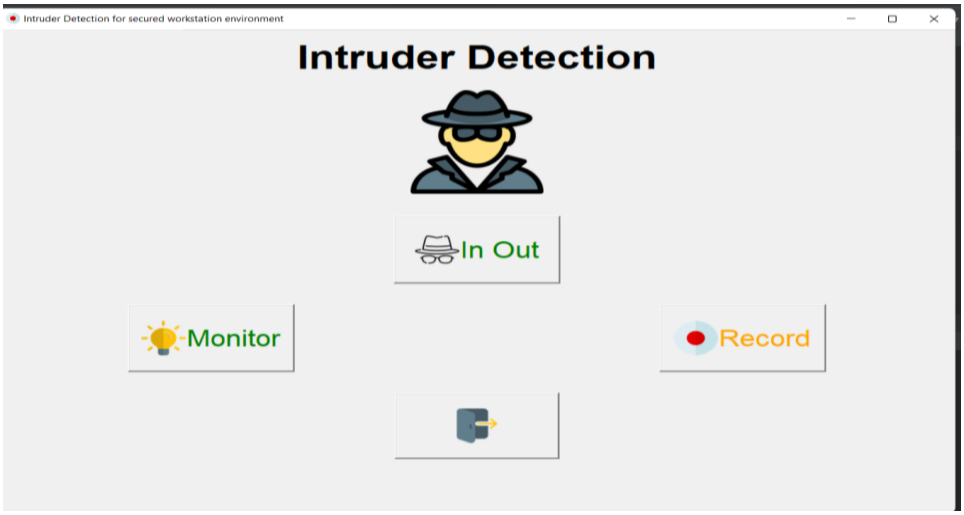
We just have to feed in two images/frames which we have captured earlier, so we just feed them in and its gives us out the masked image with score.

CHAPTER 4

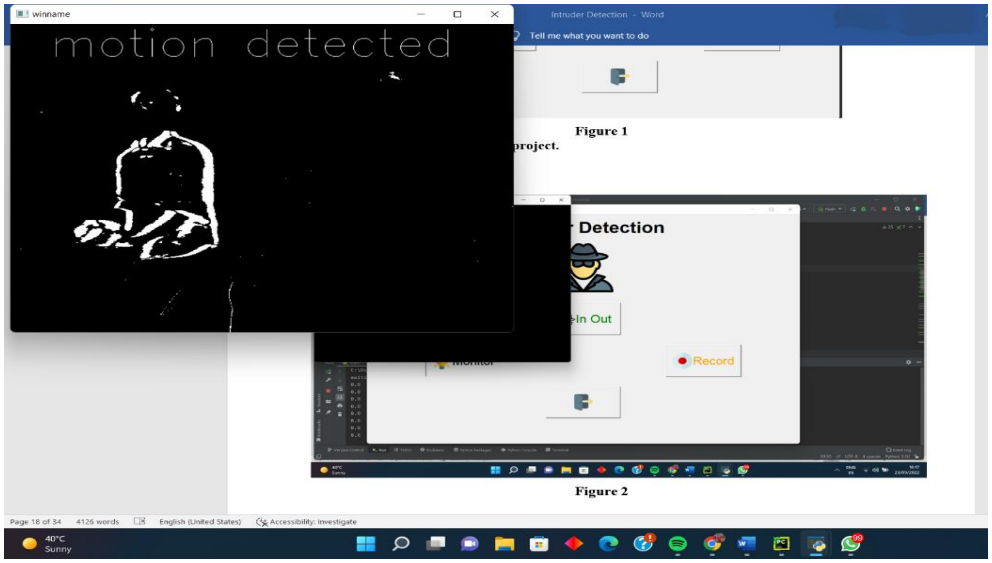
RESULT AND ANALYSIS

4.1 Output Images

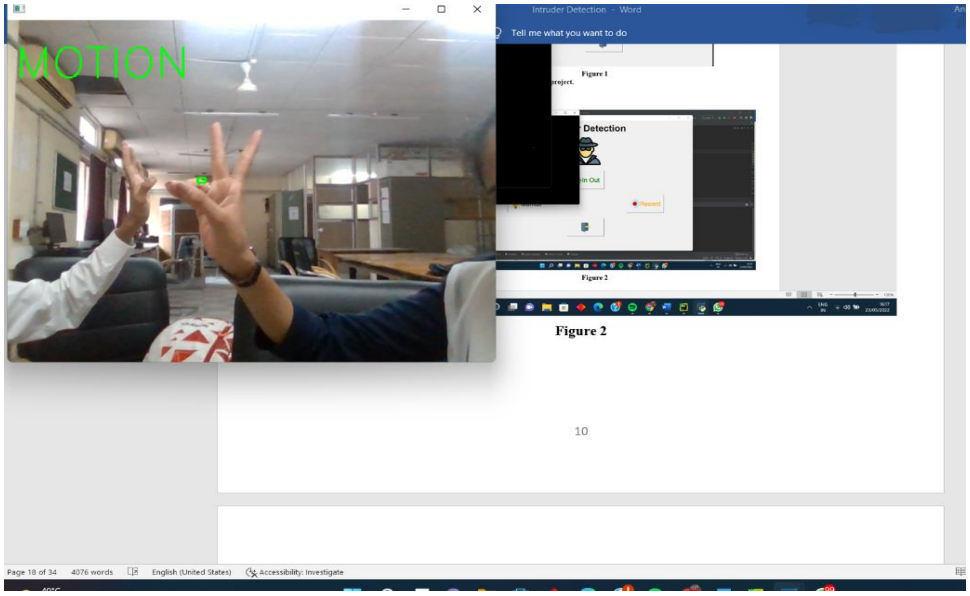
1. Main Module



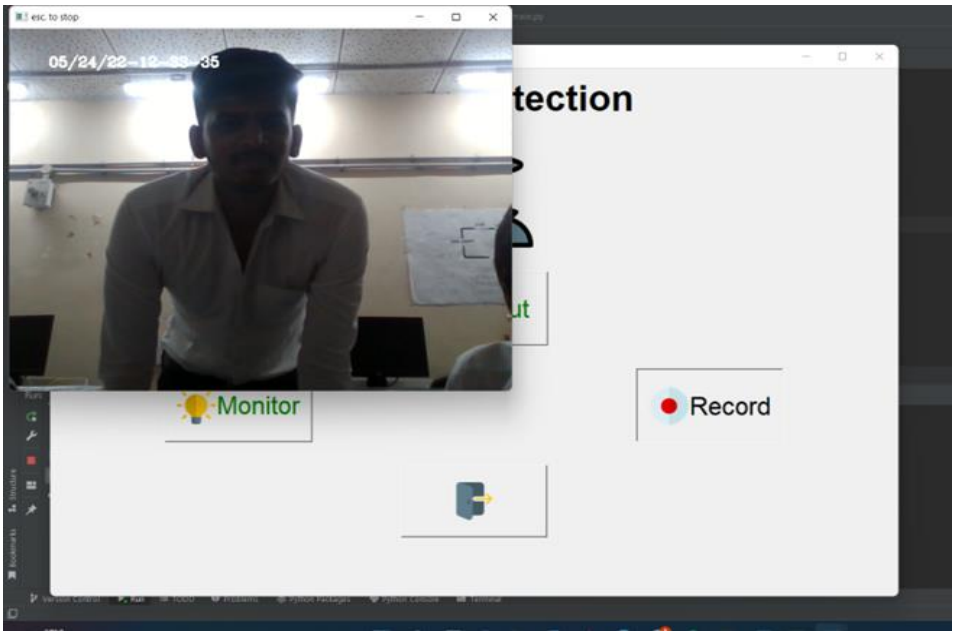
2. Monitor



3. In-Out



4. Record

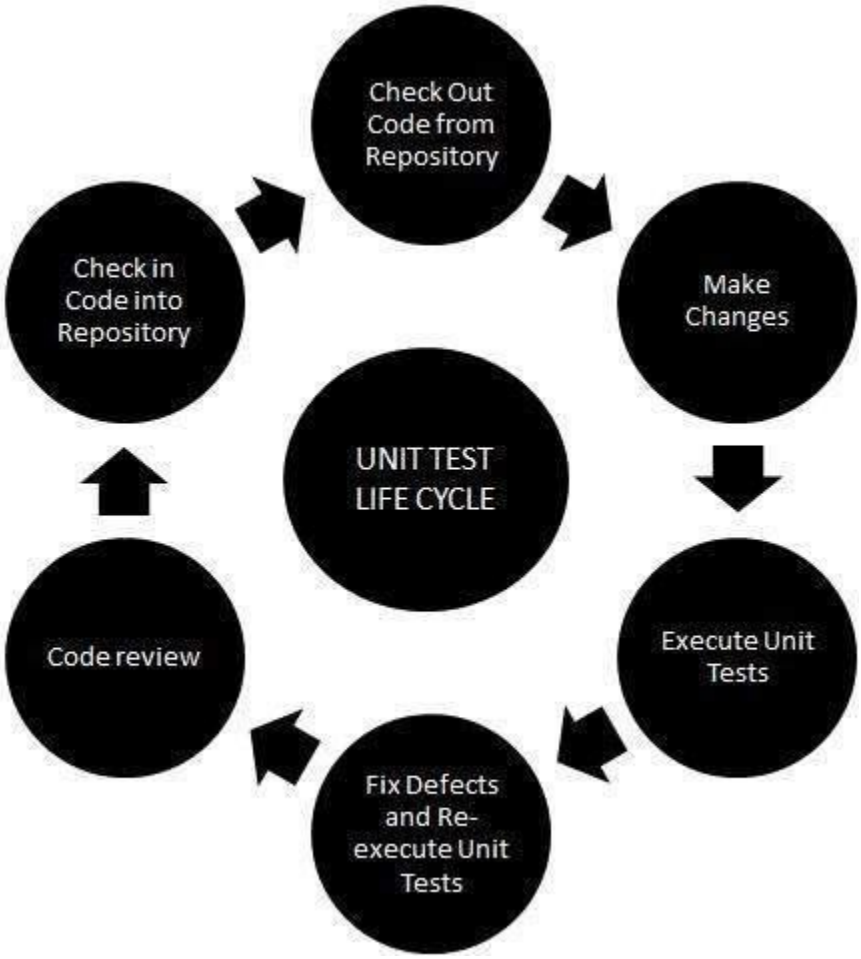


4.2. TESTING

Testing: We have performed unit testing in this application. Unit Testing is a type of software testing where individual units or components of a software are tested. The purpose is to validate that each unit of the software code performs as expected. Unit Testing is done during the development (coding phase) of an application by the developers. Unit Tests isolate a section of code and verify its correctness. A unit may be an individual function, method, procedure, module, or object.

Advantages:

- Reduces Defects in the Newly developed features or reduces bugs when changing the existing functionality.
- Reduces Cost of Testing as defects are captured in very early phase.
- Improves design and allows better refactoring of code
- Unit Tests, when integrated with build gives the quality of the build as well.



Each and every component of application which was coded was tested then and there subsequently in order to reduce the chances of crashing the application. These components were of user interface, back-end services, storage of data in database, etc.

After that system testing is done. System Testing is a type of software testing that is performed on a complete integrated system to evaluate the compliance of the system with the corresponding requirements.

In system testing, integration testing passed components are taken as input. The goal of integration testing is to detect any irregularity between the units that are integrated together. System testing detects defects within both the integrated units and the whole system. The result of system testing is the observed behavior of a component or a system when it is tested.

4.2 Debugging

Debugging is the process of detecting and removing of existing and potential errors (also called as 'bugs') in a software code that can cause it to behave unexpectedly or crash. Toprevent incorrect operation of a software or system, debugging is used to find and resolve bugsor defects.

Debugging in Python is facilitated by pdb module(python debugger) which comes built-in to the Python standard library. It is actually defined as the class Pdb which internally makes use of bdb(basic debugger functions) and cmd(support for line-oriented command interpreters) modules. The major advantage of pdb is it runs purely in the command line thereby making it great for debugging code on remote servers when we don't have the privilege of a GUI-based debugger.
pdb supports-

- Setting breakpoints
- Stepping through code
- Source code listing
- Viewing stack traces

Technology used to make this project :

- as mentioned earlier **Python** language is used.
- Sublime Text Editor is used to write the code.
- Windows 10 OS is used to run and create this minor project.
- HP-ay503tx laptop is used.
  - Intel i5 is used
  - 512GB SSD
  - 8GB RAM
  - In-built webcam hp-true vision
- Terminal to run the code

Platforms already tested,

It is tested on Linux Mint, Linux Ubuntu, Windows 7, Windows 10.

CHAPTER 5
FUTURE SCOPE AND IMPLICATION

Based On the technology improvements such being having the capability of small size but high processing power this project can be broadly used. Below is some future workout on this project.

In this project we have used Haar cascade algorithm for the detection of objects and storing it into the system of administrator we can make it highly effective and efficient by investing more money in processing power required to execute this program. We can add Noice detection and detecting missing object from the secured place

- Creating Portable cctv.
- Adding in-built night vision capability.
- Adding deep learning if having high power device.
- More feature such as
  - Deadly weapon detection
  - Accindent detection
  - Fire Detection
  - Much more...
- Making a stand alone application with no requirements such as python, etc.
- Making standalone device.

Adding DL support would create broad scope in this project such as with DL we would be able to add up much more functionality.

CHAPTER 6

CONCLUSION

The objectives of Intruder Detection for Secured Workstation Environment are successfully fulfilled after all the phases of development. It is able to coordinate the tasks and activities in for administrator of the team. This application is complete in working condition. Its development is done in Python. The data of user (project members) in database is stored effectively and provides data abstraction for the users. This application is able to create secured workstation environment for any workstation by providing brand new features like motion detection of entered person by saving picture of intruder on system. project development is generated and shared in an efficient manner. This project is helping in version controlling of software code in project, making the project development precise, clear and saving time.

REFERENCES

1. Sivakumar, Swetha, and R. GomathiBhavani. "Image Processing Based System for Intrusion Detection and Home Security Enhancement." In 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), pp. 1676-1680. IEEE, 2018.
2. Bhanse, Vivek Kishor, and M. D. Jaybhaye. "Face Detection and tracking using Image processing on Raspberry Pi." In 2018 International Conference on Inventive Research in Computing Applications (ICIRCA), pp. 1099-1103. IEEE, 2018.
3. Kakadiya, R., Lemos, R., Mangalan, S., Pillai, M. and Nikam, S., 2019, June. Ai based automatic robbery/theft detection using smart surveillance in banks. In 2019 3 rd International conference on Electronics, Communication and Aerospace Technology (ICECA) (pp. 201-204). IEEE.
4. Wang, Jin-xiang. "Research and implementation of intrusion detection algorithm in video surveillance." In 2016 International Conference on Audio, Language and Image Processing (ICALIP), pp. 345-348. IEEE, 2016
5. Tian, Y.L., Brown, L., Hampapur, A., Lu, M., Senior, A. and Shu, C.F., 2008. IBM smart surveillance system(S3): event based video surveillance system with an open and extensible framework. Machine Vision and Applications, 19(5), pp.315-327
6. Cuppens, F., &Miege, A. (2002, May). Alert correlation in a cooperative intrusion detection framework. In Proceedings 2002 IEEE symposium on security and privacy (pp. 202-215). IEEE.
7. https://github.com/opencv/opencv/tree/master/data/haarcascades