

SMART ROBOT GUARD



By

M. Muizz Masood

01-132172-021

Hamza Waseeq

01-132172-012

SUPERVISED BY

Sir M. Kashif Naseer

**Department of Computer Engineering
Bahria University Islamabad
2021**

SMART ROBOT GUARD



By

M. Muizz Masood (01-132172-021)

Hamza Waseeq (01-132172-012)

SUPERVISED BY

Sir M. Kashif Naseer

Report is Submitted to the Department of Computer Engineering in
the partial fulfillment of the requirements for the degree of
Bachelors in Computer Engineering.

**Department of Computer Engineering
Bahria University Islamabad**

UNDERTAKING

We certify that the research work presented in this thesis is to be best of our knowledge all sources used, and any help received in the preparation of this study have been acknowledged. We hereby declare that we have not submitted this material, either in whole or in part, for any other degree of this or any other institution.

.....

M. Muizz Masood

01-132172-021

.....

Hamza Waseeq

01-132172-012

DEDICATION

This thesis is dedicated to the sake of Allah, our Creator and our Master, our great teacher and messenger, Mohammed (P.B.U.H), who taught us the purpose of life and our supervisor M. Kashif Naseer who always supported us by giving his substantial and appropriate guidance. We dedicate this effort to our great parents; without their support we would not be able to complete our thesis and who never stop giving of in countless ways. Moreover, the dedication is for the people who always lead us through the valley of darkness with light of hope, our respectable teachers whose support was always with us whenever we needed. Last but not the least we dedicate this research to our kind project co-coordinator, Sir Waleed Manzoor, who always stands by us when things look disappointing, he encouraged and supported us.

ACKNOWLEDGEMENT

First and foremost, we are grateful to Allah Almighty who gave us the strength to sum up our thesis effectively, without HIS blessings we wouldn't be able to do anything. We are gratified to our supervisor M. Kashif Naseer, who helped us for our thesis completion effectively. We are grateful to him for the time, supervision, and encouragement he gave us throughout our work.

After that, we are grateful to our parents who supported us and provide us bravery to complete our work professionally and other family members, our best friends, who encouraged us, gave us confidence, and supported us by all means to complete our thesis. No doubt, without the help and support of our family and friends it was an unmanageable task for us.

ABSTRACT

Robotics and Image Processing are contributing to robot automation and are becoming very popular day by day. The field of Automation is spreading vastly covering almost all sectors of human life leading to development and innovation by students. Smart Robot (S-BOT) provides autonomous indoor services like facial recognition, detection, and obstacle avoidance. S-BOT is a reprogrammable multi-functional device designed to move autonomously and detect the face of a person while entering in a specified area then recognize their faces and check if persons information is in the database. Researchers, educationalists, and hobbyist communities are exploring and developing the field of robotics. Project itself is a doorway to explore and enhance the maximum results from a machine to perform humanly tasks. Robotics has become one of the best domains that attracts researchers and passionate individuals to perform and learn from each other's experiences in the field of Domestic and Service Robots. The services of S-BOT are not static and can be utilized as required by implementing self-made algorithms. Hence S-BOT always remains in development phase since progress in robots depends on the technological contributions made by the researchers.

Keywords:

Automated Robot, Facial Recognition, Facial Detection, Obstacle Avoidance, Python.

Table of Contents

UNDERTAKING	iii
DEDICATION	iv
ACKNOWLEDGEMENT	v
ABSTRACT.....	vi
Chapter 1.....	1
Introduction.....	1
1.1 History of Robotics	1
1.2 Modern History	3
1.3 Project Background.....	4
1.4 Project Objectives	5
1.5 Project Description.....	5
1.6 Project Scope	6
1.7 Methodology	6
Chapter 2.....	7
Literature Review.....	7
2.1 Types of Autonomous Robots	7
2.2 Proposed System.....	7
2.3 Components of Robot	8
2.4 Working of Components.....	11
2.5 Software and Algorithms	14
2.6 Project Division	15
2.7 Working Hierarchy	16
Chapter 3.....	17

Design and Implementation	17
3.1 Hardware Description	18
3.2 Software Design.....	19
3.3 Hardware Design	20
3.4 Hardware Architecture.....	22
Chapter 4.....	23
Software Implementation.....	23
4.1 Facial Detection	23
4.2 Facial Recognition	27
4.3 Database.....	32
Chapter 5.....	33
Results and Limitations.....	33
5.1 Components of S-BOT	33
5.2 Integration.....	37
5.3 Total Workflow of S-BOT.....	38
5.4 Limitations	39
Chapter 6.....	40
Conclusion and Future Work.....	40
6.1 Conclusion	40
6.2 Future Work.....	40
REFERENCES	45
ABBREVIATION.....	47
ANNEXURE.....	48
Data Set Creation	48

Testing of Code.....	48
Facial Detection Algorithm.....	49
Facial Recognition Algorithm.....	50
Pictures Stored in Database	51
Arduino IDE Code	52

List of Figures

Figure 1-1 Unimate	2
Figure 1-2 Shakey	2
Figure 1-3 Famulus	3
Figure 1-4 Quasi-dynamic robot.....	3
Figure 1-5 Aquarobot.....	4
Figure 1-6 Mars Pathfinder.....	4
Figure 1-7 Waterfall Method	6
Figure 2-1 Workflow of Robot	8
Figure 2-2 Aurdino UNO.....	8
Figure 2-3 ATMEGA 328p.....	9
Figure 2-4 Motor Driver L298n.....	9
Figure 2-5 DC gear Metal Motor 12V	9
Figure 2-6 SR04 Ultrasonic Sensor	10
Figure 2-7 12 V DC battery	10
Figure 2-8 Caster Wheel	11
Figure 2-9 Arduino UNO.....	12
Figure 2-10 Motor Driver L298.....	12
Figure 2-11 SR04 Ultrasonic Sensor	13
Figure 2-12 Internal structure of DC Gear Metal Motor	14
Figure 3-1 S-BOT	18
Figure 3-2 Layers of Robot.....	19
Figure 3-3 Hardware Architecture	22
Figure 4-1 Haar-like features	25

Figure 4-2 Integral Images.....	26
Figure 4-3 AdaBoost Classifier	26
Figure 4-4 Cascading Classifiers.	27
Figure 4-5 LBPH Single Cell Result	29
Figure 4-6 Block Diagram of Facial Recognition.....	29
Figure 4-7 Flow Chart of Facial Recognition	30
Figure 4-8 Flow Chart of LBPH	31
Figure 5-1 Arduino IDE.....	34
Figure 5-2 Pycharm Facial Detection code.....	35
Figure 5-3 Pycharm Facial Recognition code.....	36
Figure 5-4 Pictures in Database	37
Figure 5-5 Workflow of S-BOT	38
Figure 6-1 2D Map feeding of Robot	41
Figure 6-2 3D Map feeding of Robot	42
Figure 6-3 2D Target Tracking through JPDAF.....	43
Figure 6-4 Zoomed in Target tracking through PMHT	44

List of Tables

Table 2-1 Software Implementation of S-BOT	15
Table 3-1 Hardware Specifications of S-BOT	17
Table 3-2 Software Specifications of S-BOT	18

Chapter 1

Introduction

In the field of Robotics, Smart Robots are drawing attention of researchers. The aim of these efforts is to solve challenging modern world problems. Research in Robotics and Image Processing is growing at a rapid pace, but some areas need special attention as compared to others. The robot should perceive information capture pictures on its own in different scenarios under various constraints.

Robotics is concerned with the development of systems which harness the strengths from various fields of Engineering to develop integrated systems and machines capable of performing in an autonomous and intelligent manner. It is a growing field in Sciences and Technology which exists at the intersection of Electrical, Mechanical, Computer and Software Engineering.

The main purpose of robot is to design a machine that can assist the human being in different areas of the field. The types of robots you will be able to see are works that are risky, boring, or too difficult. These robots can be found in many fields like automotive, medical and space industries. Over millions of robots are working in these industries nowadays.

1.1 History of Robotics

The first ever fully digitally operated and programmable robot was invented in 1961 by George Devol called “Unimate”. The reason of behind inventing this robot was to assist the human beings and making their work easy. The purpose of “Unimate” was to stack the hot pieces of metal from the die casting machine. It can be seen in Figure shown below.

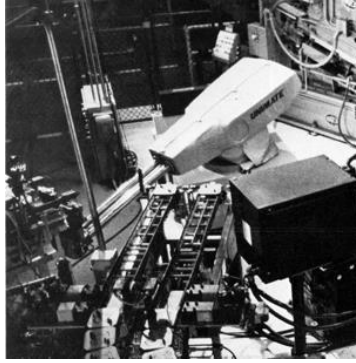


Figure 1-1 Unimate

The academia was making much progress at that time in the creation of robots. After the first ever fully digitalized robot Charles Rosen developed a new robot named as “Shakey”. It was an advanced and a better version of the original Unimate, which was designed for industrial applications. It could move around the place and see through its television eyes and move across the unfamiliar areas.

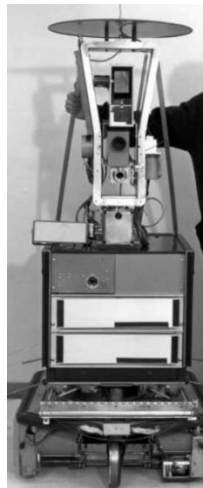


Figure 1-2 Shakey

The first industrial robot with six electromechanically driven axes was invented in 1973 by KUKA named “Famulus”. It was a breakthrough for the automotive industry.

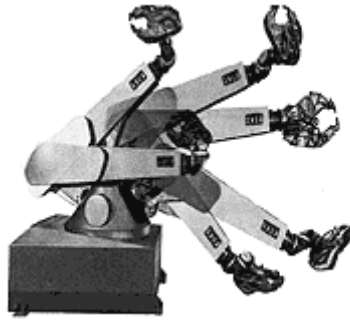


Figure 1-3 Famulus

1.2 Modern History

In 1980, Quasi-Dynamic walking was the first microcomputer. It had the ability to take one step every 10 seconds. It was developed by “Ichiro Kato” at Waseda University, Tokyo. It can be seen in the given figure below.

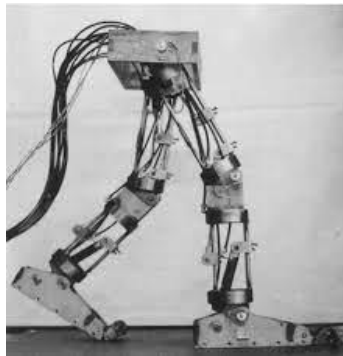


Figure 1-4 Quasi-dynamic robot

In 1985, Japan Marine Technology Department invented the first underwater walking robot named “Aquarobot”. The main purpose of this robot was to carry out the underwater inspecting work accompanied instead of sending Divers. The two main functions of the robot were to measure the flatness of the rock and to observe the underwater structure by TV camera. As shown in the figure below.

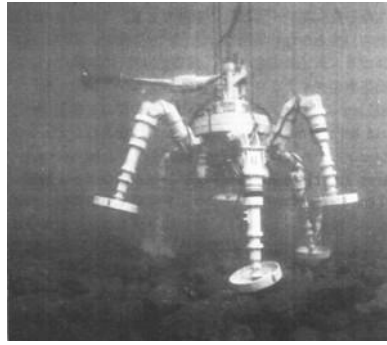


Figure 1-5 Aquarobot

Later, the technology was getting advanced and then NASA sent their wheeled robot named “Mars PathFinder” in 1997 to Mars to send images and data about the Mars on earth.

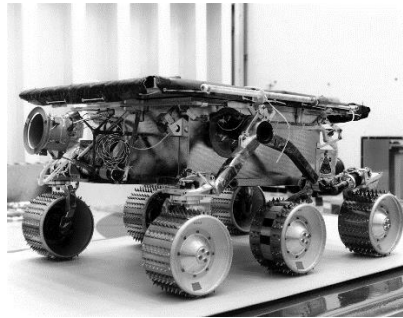


Figure 1-6 Mars Pathfinder

1.3 Project Background

Autonomous robots have gained a lot of interest among the students and researchers. It plays an important part in our daily lives as it has benefited human beings potentially. The main purpose of designing the robots is to ease the mankind in their daily work and handle he tasks by themselves.

Almost all the sectors are in the demand of autonomous robots. Industries like medical or health care, military, space are using autonomous robots nowadays.

According to the report approximately 43% of the crimes were occurred in last 5 years due to the less secure systems of the areas. All over the world people are facing this serious problem of robbery, theft and data stolen. Keeping all these aspects in mind we came up with the idea that if we could design a surveillance robot which could move

autonomously in the area and capture the pictures at real time to detect and recognize a face and then crosscheck the information of a person from the database. It can decrease the percentage to a great extent. This was the starting of our innovative concept and we came up with the project of Smart Robot Guard.

1.4 Project Objectives

The motivation behind this project is to reduce the security lapses that are occurring in our security system. The main objective is to develop a Reprogrammable multi-functional Robot designed to move autonomously then detect and recognize the person entering the premises of the area. The focus of our project will be on reduce the security lapses and stop the intruder from breaching into the sensitive areas. Our system includes an autonomous robot that can move randomly, avoid obstacles, and capture the pictures of a person to recognize him by crosschecking through the database.

Smart Robot Guard will service for the sensitive areas where chances of penetration by an unauthorized person is high.

1.5 Project Description

A Robot is a reprogrammable multi-functional device designed to move autonomously then detect and recognize the person entering the premises of the area. Software and hardware are the two main portions in developing a complete Robot. Robots play a vital role in domestic indoor and outdoor environments for places where the security system is not so robust.

The project is all about building an autonomous security robot having the capabilities of avoiding obstacles, detecting, recognizing, and crosschecking the persons face through database. The main deliverables our robot would be working on is.

- Facial Detection
- Facial Recognition
- Robot Autonomous maneuvering
- Obstacle avoidance

1.6 Project Scope

This project will be used in security system to detect the person in sensitive areas. To provide useful tool for monitoring the people entering in the sensitive areas where breaching is possible. This project is useful in recognizing the person, which means that when it can find any suspicious person it alerts the security personnel in that location.

1.7 Methodology

There are many software developments algorithms available like Haar-Cascade, Eigen faces, Viola Jones. Each algorithm has its own benefits and drawbacks for a given situation. After studying a bit about the algorithms, it was decided to use Viola Jones and LBPH algorithm for the development of the proposed system.

There are many software developments models available like Waterfall Model, Spiral Model, Agile Development and incremental model. Each model has its own benefits and drawbacks for a given situation. After studying a bit about the models, it was decided to use the incremental model for development of the proposed system.

Waterfall methodology is used in our system. It is sequential design process. It flows steadily that is why it called waterfall method.

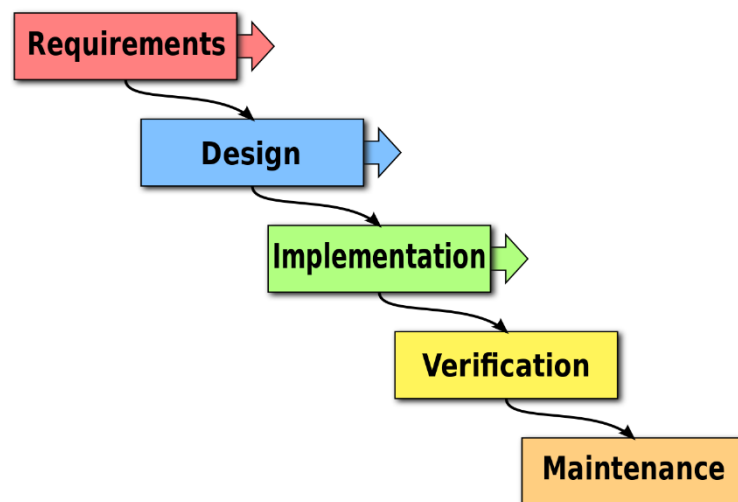


Figure 1-7 Waterfall Method

Chapter 2

Literature Review

The purpose of this chapter is to describe the literature studied and analyzed to develop an autonomous robot. All the discussions and the research paper that are read to finalize the work and select the tools and software by which we should start and end the project are proposed in this chapter. This chapter explains the reference to the different research papers, current existing similar projects, and our proposed working of autonomous robot.

2.1 Types of Autonomous Robots

There are four different types of autonomous robots.

- **Programmable Robots** – These robots can be reprogrammed and can change their function and application.
- **Non-Programmable Robots** – These robots are the exploiter lacking reprogrammable controlling devices.
- **Adaptive Robots** – These robots can be adapted to certain extent and can be used in the required adapted areas.
- **Intelligent Robots** – These robots contain microprocessors and sensors which makes them capable of storing and processing the data and perform efficiently for analyzing and performing tasks.

2.2 Proposed System

Our proposed automated robot with move in a specific area randomly while avoiding obstacles and capture the pictures for recognizing a person entering the premises by crosschecking through the database. Our proposed system will be working on the workflow of the robot that is shown in the figure below.

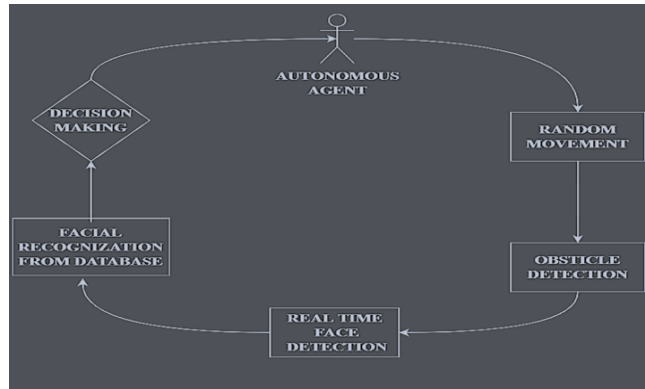


Figure 2-1 Workflow of Robot

When the robot is left in the surveillance area, the robot works in a proper flow that is shown above in Figure 2-1. The robot moves autonomously in random movement while observing that there is no obstacle in front or sides of it. Then after that it detects the face of a person through the camera that is mount on the top of the robot. Then after the face of the is detected the face of a person is recognized and crosscheck the information of the person through the database and then makes the decision if that person is unknown person or authorized personnel.

2.3 Components of Robot

For the purpose of designing the robot there are different tools and equipment that are required by which a robot is made functional. Some of the equipment is used to move the robot and some of the equipment are used to control the robot to function properly. The equipment and components that we have used for designing the robot are shown in the given figures below.



Figure 2-2 Aurdino UNO

Arduino UNO is a microcontroller board that was developed by Arduino.CC. it contains a microcontroller Atmega 328. It is used for the embedded projects like RC cars, robots.

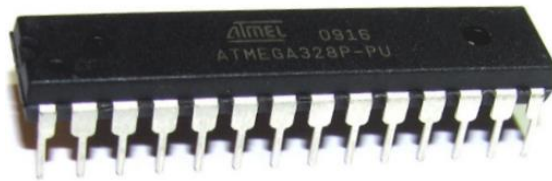


Figure 2-3 ATMEGA 328p

The ATmega 328p is a single-chip microcontroller that is used in Arduinos for getting high performance from very low power consumption. It is mostly used as a processor in the Arduino Boards such as Arduino UNO or Arduino MEGA.

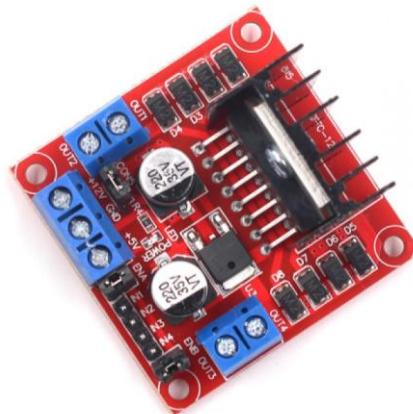


Figure 2-4 Motor Driver L298n

The L298n Motor Driver is a dual H-Bridge motor driver which controls speed and direction of two DC motor simultaneously at one time. This motor driver can drive motors that have voltages between 5v to 35v with the current up to 2Amps.



Figure 2-5 DC gear Metal Motor 12V

DC metal motor is a high temperature resistant, durable motor used in different places for the movement of an object. It consists of a gear box that can be used to adjust the speed of the motor. The speed reduction can be determined by the gear ratio and efficiency of the gear box.



Figure 2-6 SR04 Ultrasonic Sensor

The SR04 Ultrasonic Sensor is a distance Ultrasonic sensor that is mostly used in the field of Robotics to avoid obstacles from contacting them. The SR04 Ultrasonic Sensor is a small efficient sensor that can detect a non-contact range of about 2cm to 4m.



Figure 2-7 12 V DC battery

To operate any electronic device, we need current or voltage. The current or voltage can be given through different ways like Alternating Current (AC) electricity that can be given through the main switch or through the Direct Current (DC) electricity that can be given through batteries. Some of the batteries are rechargeable and some of them are not which can only be used once.



Figure 2-8 Caster Wheel

Caster wheel is the component that is used to move the robot in either direction. When the power is given to the motors those motors make the wheels to move to make robot or machine moving.

2.4 Working of Components.

2.4.1 Arduino UNO

Arduino UNO is a microcontroller board that was developed by Arduino.CC. it contains a microcontroller Atmega 328. It is used for the embedded projects like RC cars, robots.

Arduino UNO is one of the most valuable addition in the field of electronics that consists of a USB interface, 14 Digital Input/Output Pins, 6 analog pins and an Atmega 328 microcontroller. It also contains Tx and Rx pins that are used for serial communication.

There are different types of Arduinos introduced in the market for different purposes of electronics. Some of them are.

- Arduino UNO
- Arduino Due
- Arduino Mega
- Arduino Leonardo

These Arduino boards are normally used to design and construct projects related to digital electronics, embedded systems, robotics, IoT etc.

Pin configuration of Arduino can be seen in the picture shown below.

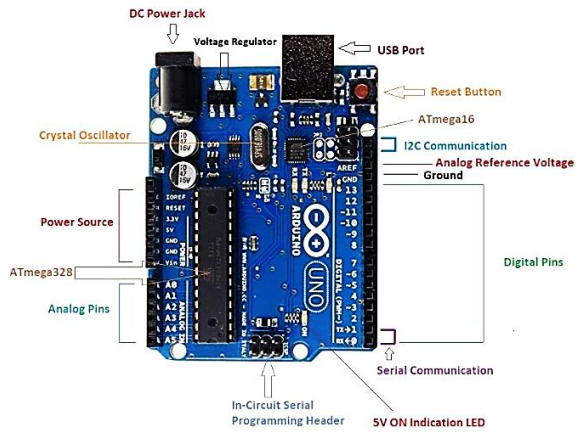


Figure 2-9 Arduino UNO

2.4.2 Motor Driver L298n

The L298n Motor Driver is a dual H-Bridge motor driver which controls speed and direction of two DC motor simultaneously at one time. It is a high-power version of last L293 Motor Driver. This motor driver can drive motors that have voltages between 5v to 35v with the current up to 2Amps.

The L298n Motor Driver contains two screw terminal blocks for the ground pin. The Vcc pin for motor can be used as an input and as an output. It depends on the voltage used by the motors. The L298n Motor Driver contains 2 Input pins, an on-board 78M05 regulator and 3 3.5mm pitch screw terminals.

Pin configuration of the motor driver L298 can be seen in the picture shown below.

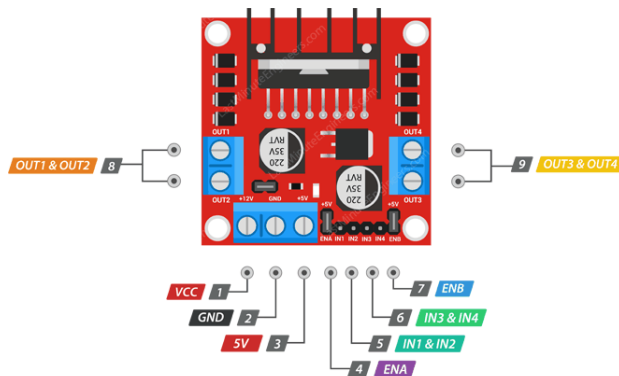


Figure 2-10 Motor Driver L298

2.4.3 SR04 Ultrasonic Sensor

The SR04 Ultrasonic Sensor is a distance Ultrasonic sensor that is mostly used in the field of Robotics to avoid obstacles from contacting them. It consists of two Ultrasonic transducers in which one acts as a transmitter and the second one acts as a receiver. The transmitter converts the electric signal into 40 KHz Ultrasonic sound pulse and the receiver receives the signal. If it receives a 40Khz signal, then it produces an output pulse to determine the distance.

The SR04 Ultrasonic Sensor is a small efficient sensor that can detect a non-contact range of about 2cm to 4m with an accuracy of 3mm. The measuring angle of the sensor is about 15 degrees at the operating current of 15mA or 5V.

Pin configuration of the SR04 Ultrasonic Module can be seen in the picture shown below.

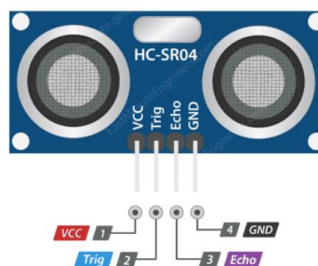


Figure 2-11 SR04 Ultrasonic Sensor

The SR04 Ultrasonic Module has 4 pins, Ground, VCC, Trig and Echo. The Ground and the VCC pins of the module should be associated with the Ground and the 5 volts sticks on the Arduino Board individually and the trig and echo pins to any Digital I/O stick on the Arduino Board. To produce the electric signal, you must set the Trig on a High State for 10 μ s. That will convey an 8-cycle sonic burst which will go at the speed of sound, and it will be gotten in the Echo stick.

2.4.4 DC 12V Gear Metal Motor

DC metal motor is a high temperature resistant, durable motor used in different places for the movement of an object. It is capable of sustaining weight capacity of 3Kg at 12V or 100 RPM. It consists of a gear box that can be used to adjust the speed of the

motor. The speed reduction can be determined by the gear ratio and efficiency of the gear box. It is a low noised and high torqued motor such that it can drive 3Kg object at 100RPM.

The internal structure of the motor can be seen in the image given below.

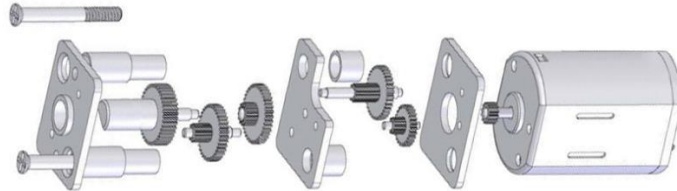


Figure 2-12 Internal structure of DC Gear Metal Motor

2.5 Software and Algorithms

In this project, we will be using Python Language on Open CV. When a person will enter the premises of the specific area the robot will detect the face through the camera and then recognize the person. Then it will crosscheck the information of the person through database. All of this work will be done with the help of Python and SQL.

The algorithm we will use for facial detection is Viola Jones Algorithm and Local Binary Patterns Histogram (LBPH) Algorithm. To crosscheck the information of the person from the database we used SQLite database.

The Software part is divided into three divisions.

1. Facial Detection
2. Facial Recognition
3. Database

The table given below shows the tools used for software implementation of Smart Robot Guard.

Table 2-1 Software Implementation of S-BOT

Operating System	Arduino IDE
Face detection	Pycharm, Python
Facial Recognition	Pycharm, Python
Database	SQL
Autonomous Movement	Arduino IDE

2.6 Project Division

The project is divided into different categories depending upon the nature of work and the complexity of the tasks. The major sections of both software and hardware covered by our project are as follows:

2.6.1 Software Architecture

The software architecture is an important part of the robot. The purpose of software architecture is to execute all the work by which the robot will function autonomously and make it for the surveillance purpose. The software architecture is divided into these:

- Installation of Arduino IDE & Pycharm Open CV
- Creating and Building packages in python and Arduino
- Testing of motors and Sensors
- Software Simulations on Arduino IDE
- Testing of codes for Image Processing

2.6.2 Hardware Architecture

The hardware architecture is an important part of the robot. The purpose of hardware architecture is designing the robot that will perform function like autonomous movement and surveillance. The hardware architecture is divided into these:

- Hardware assembly
- Communicating of Software with Hardware components
- Autonomous Movement

- Facial Recognition

2.7 Working Hierarchy

There is a hierarchy through which the robot is assembled and made functional. All the work from designing to assembling the robot is done through the hierarchy. The working hierarchy of the S-BOT is given below:

- Installation of Arduino IDE
- Installation of Pycharm, Python
- Creating and Building packages
- Software simulations
- Autonomous Movement (Simulation)
- Testing Motor Driver
- Testing Ultrasonic Sensors
- Face Detection
- Facial Recognition
- Robot Hardware Assembly
- Autonomous Movement
- Testing of motors movement
- Object Detection and avoidance

Chapter 3

Design and Implementation

In this chapter we will discuss the Hardware implementation of the S-BOT. In this project, we have used a differential drive robot base to execute different tasks such as autonomous movement, Obstacle avoidance, facial recognition. Furthermore, this chapter describes about the robots named S-BOT hardware and software design. The robot heavily uses Arduino IDE for its component's communication. Arduino IDE is a basic platform and an open-source application for Windows, Linux, macOS. It is used to write and upload the code to Arduino boards by using C or C++. The simple architecture of the robot allows us to focus on the basic steps involved in the development of a robot which can be used for security purposes in sensitive areas.

The processing of the data is achieved by the Haier Notebook 11 Intel Core m3 machine. One NoteShip Webcam for image processing purpose and SR04 Ultrasonic Sensor for obstacle avoidance.

The tools and software used in the project are listed in below table 3.1 and table 3.2.

Table 3-1 Hardware Specifications of S-BOT

Name	S-BOT
Processing Machine	Haier Notebook 11 Intel Core m3 machine
Structure	Aluminum and Fiber
Microcontroller Board	Arduino UNO
Sensor	SR04 Ultrasonic Sensor
Camera	NoteShip Webcam

Table 3-2 Software Specifications of S-BOT

Operating System	Arduino IDE
Face detection	Voila Jones Algorithm
Facial Recognition	Local Binary Pattern Histogram (LBPH)
Database	SQLite
Autonomous Movement	Arduino IDE

This section briefly discusses the working of our robot. The processing system of our robot is based on Haier Notebook 11 Intel Core m3 machine. Other physical components include NoteShip Webcam for image processing purpose and SR04 Ultrasonic Sensor for obstacle avoidance.



Figure 3-1 S-BOT

3.1 Hardware Description

We have defined our robot into three basic layer architecture. The first layer or the bottom layer of the robot is in which it contains the wheels and DC Motors that will be responsible for the wheels to rotate in either direction.

The second layer or the middle layer of the robot contains SR04 Ultrasonic Sensors that will be used to detect and avoid the obstacles. Then, we have an Arduino to which every component of the robot is connected except the camera and that will be responsible to making decisions for every component to work for which it is designed. When the robot should move forward, backward, left, or right. Every decision is made by the help of Arduino. Then, we have the batteries that are connected to the components of the robot to make it them work for their designated purpose.

Then the upper layer or the third layer of the robot has a camera mounted on top for image Processing purpose. The camera will be connected to our processing machine (Haier Notebook 11 Intel Core m3) for recognition and fast processing at real time facial Recognition.

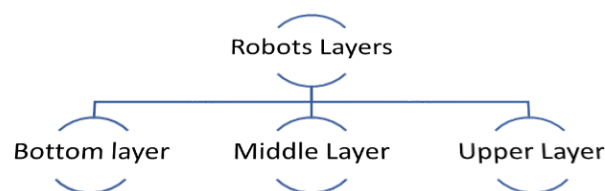


Figure 3-2 Layers of Robot

3.2 Software Design

The approach we used to develop the software architecture is to develop components or modules of the software and then connect them using Pycharm, Arduino. The platform most commonly used for connecting software to the hardware for getting desired output from the hardware components is Arduino.

We used Python language for image processing purpose on Pycharm. We used Voila Jones Algorithm for Facial Detection and Local Binary Pattern Histogram (LBPH) for Facial Recognition. The reason why Voila Jones Algorithm is a good option to avail for Facial Detection is that it is a robust algorithm with very high detection rate or true-positive rate and very low false positive rate.

We used Local Binary Pattern Histogram (LBPH) for Facial Recognition because it is a robust algorithm against monotonic gray scale transformation. It is user-friendly algorithm with fast and accurate results.

3.3 Hardware Design

This section briefly discusses hardware components we have used for the development of robot. An Autonomous Robot perceives information from environment using its sensors. After data analysis, a robot takes decisions and interacts with the environment to perform the desired tasks, such as object detection, obstacle avoidance and facial recognition. integral part of the data acquired for recognition purposes, which employs various sensors like NoteShip Webcam for image processing purpose and SR04 Ultrasonic Sensor for obstacle avoidance. It also has a motor which can be used to control its position remotely. NoteShip Webcam is a good quality camera which gives 720p image quality with low power source.

Obstacle avoidance requires basic information by a robot to maneuver in a cluttered environment. SR04 Ultrasonic Sensor is the more commonly used sensor due to their availability, small size, and precision. The range of this sensor is from minimum 2cm to maximum 4m.

Controlling and managing the speed of a robot is a critical requirement in Robot motion and maneuvering. Layered design approach is a common approach used by our teams, where first layer is the bottom or the base of the Robot. This layer can either be designed indigenously using Acrylic or Aluminum sheets. Laser range finders or scanners, batteries and some other external sensors could be mounted on the base of the Robot.

Some teams use commercially available expensive robots instead of designing from scratch by themselves. Some of the commercially available Robots are NAO, Reem, Care-O-bot, DARwinOP, PR2 and Otto bock etc. Some robots have their own operating system like NaoQi for Nao while some are programmable with OS like PR2. These Robots differ in shape, capacity, degree of freedom and cost. Second layer is the upper body the Robot. The Robot sensors are used for detection of the obstacles.

In S-BOT the base along with other hardware components including NoteShip Webcam and SR04 Ultrasonic Sensor is controlled by Arduino IDE on the main system i.e. Haier Notebook 11 Intel core m3 machine.

Structural System

Research and structural production which provides support and stability. This also includes system base and wheeling systems.

Propulsion System

It deals with motion and dynamics of robot. It deals with the actions or process of pushing or pulling the object to drive. This includes gears, motors, and wheels.

Sensor and Tool

This phase will comprise of efforts and works on arms, actuators, grippers, and manipulators.

Feedback System

This phase will utilize the achieved output of the system for causing variations in order to acquire the desired or required results.

Control System

This includes all work related to joysticks, wiring, and other controllers.

Programming

All work and efforts related to program where Arduino IDE is used to develop, control, and operate the S-BOT.

3.4 Hardware Architecture

The hardware of the S-BOT is divided into three different layers. These layers contains different components of the robot which are added on different layers of the robot to make it functional. The layers which the S-BOT contains are:

- Bottom Layer
- Middle Layer
- Top layer

The components that are attached to which layer can be seen in the figure shown below.

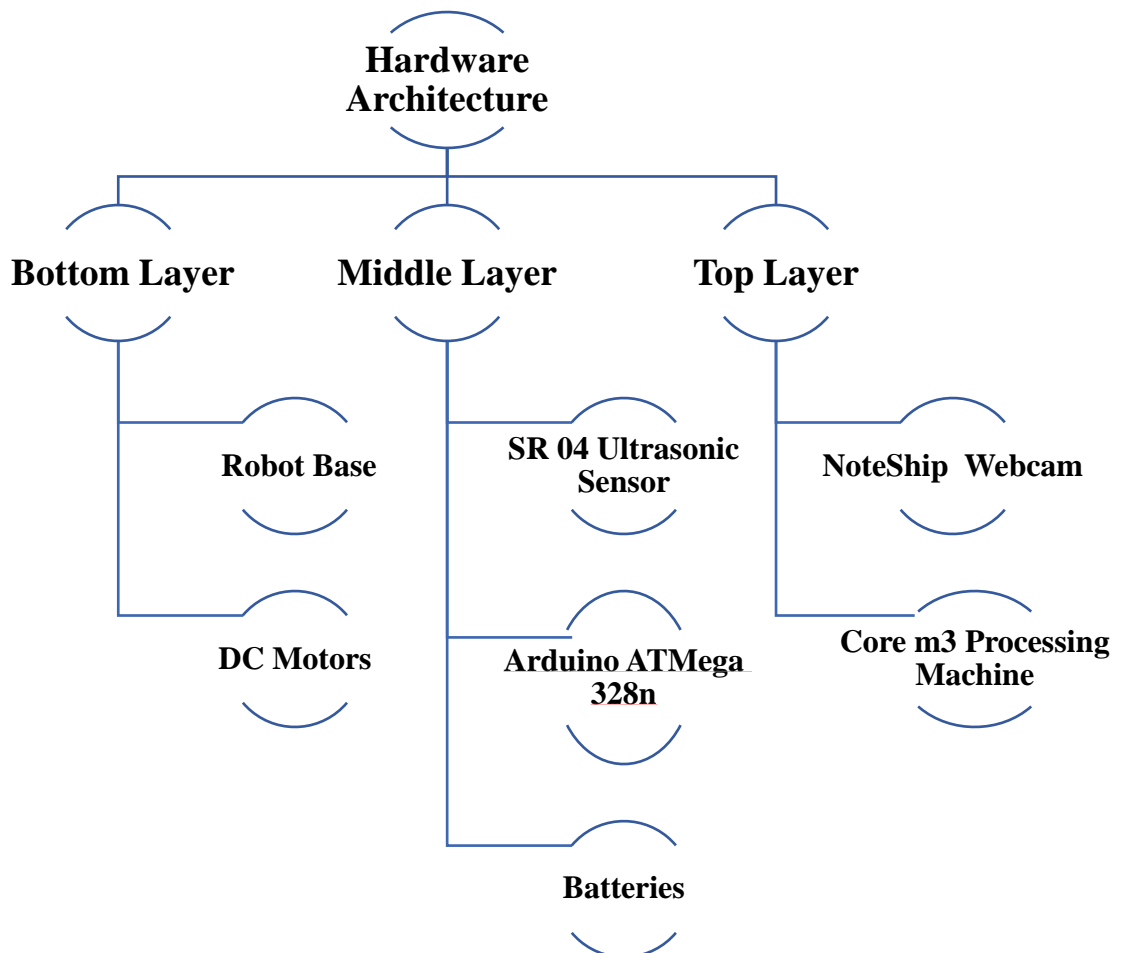


Figure 3-3 Hardware Architecture

Chapter 4

Software Implementation

In this chapter we will discuss the software implementation of the S-BOT. In this project, we have used Python Language on Pycharm software on OpenCV. When a person enters the premises of the specific area the robot detects a face from the camera then recognize the person entering the premises and crosscheck the information of the person through database.

The algorithm we used to for facial detection is Voila Jones Algorithm and Local Binary Patterns Histogram (LBPH) Algorithm. To crosscheck the information of the person from the database we used SQLite database.

The Software part is divided into three divisions.

1. Facial Detection
2. Facial Recognition
3. Database

4.1 Facial Detection

Face detection is a process that identifies faces in digital images. It is concerned with detecting instances of an object such as human faces, buildings, trees, cars, etc. The primary aim of face detection algorithms is to determine whether there is any face in an image or not. Face-detection algorithms focuses on the detection of frontal human faces. It is analogous to image detection in which the image of a person is matched bit by bit or pixel by pixel.

For facial detection there are many algorithms by which we can detect the human face. Some of them are:

- Haar-Cascade
- Eigen faces
- 3D morphable model

- Voila Jones

The algorithm we used for Facial detection is **Voila Jones Algorithm** which is one of the best algorithms for facial detection.

4.1.1 Voila Jones Algorithm

Viola Jones algorithm is named after two computer vision researchers Paul Viola and Michael Jones who proposed the “Rapid Object Detection” method in 2001. It can be trained to detect variety of object classes. A human can do this easily, but a computer needs precise instructions and constraints. To make these task more manageable, Viola–Jones requires full view frontal upright faces. Thus, in order to be detected, the entire face must point towards the camera and should not be tilted to either side. While it seems, these constraints could diminish the algorithm's utility because the detection step is most often followed by a recognition step.

4.1.2 How does Voila Jones Algorithm Works?

Despite being an outdated framework, Viola-Jones is a powerful algorithm, and its application has proven to be exceptionally notable in real-time face detection.

The Viola Jones algorithm works on four main steps on which this algorithm works:

1. Selecting Haar-like features
2. Creating an integral image
3. Running AdaBoost training
4. Creating classifier cascades

What are Haar-like features?

A Haar-Feature is like a kernel in CNN, except that in a CNN, the values of the kernel are determined by training, while a Haar-Features are manually determined. Haar-like features are digital image features used in object recognition. Like a human face shares some universal properties of the human face like the eye’s region is darker than its neighbor pixels, and the nose region is brighter than the eye region.

To find out which region is lighter or darker is to sum up the pixel values of both regions and compare them. The sum of pixel values in the darker region will be smaller than the sum of pixels in the lighter region.

There are 3 types of Haar-like features:

- Edge features
- Line features
- Four-sided features

Edge features and Line features are useful for detecting edges and lines respectively. The four-sided features are used for finding diagonal features.

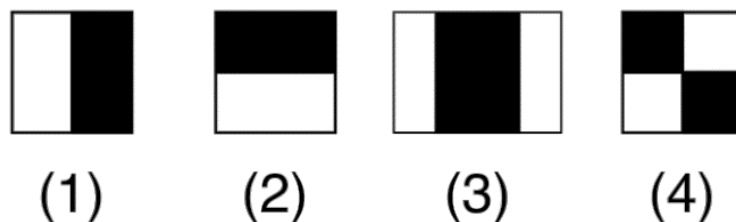


Figure 4-1 Haar-like features

As we can see in the diagram the first two images are “**Edge features**”, used to detect edges. The third is a “**Line feature**”, while the fourth is a “**Four-sided feature**”, that are used for finding diagonal features.

What are integral images?

The integral image plays its part in allowing us to perform these intensive calculations quickly so we can understand whether a feature or several features fit the criteria. It is used as a quick and efficient way to calculate the sum of pixel values in an image or rectangular part of an image.

By using these integral images, we can save a lot of time calculating the summation of all the pixels in a rectangle as we only have to perform calculations on four edges of the rectangle.

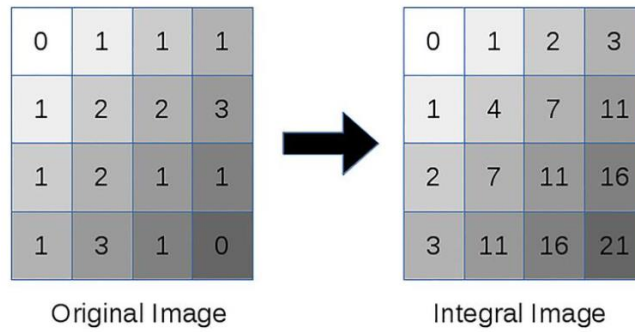


Figure 4-2 Integral Images

How is AdaBoost used?

AdaBoost is a machine learning algorithm that we use to identify best features in an image. The number of features that are present in the 24×24 image is nearly 160,000, but only a few of these features are important to identify a face. So, we use the AdaBoost algorithm to identify the best features in the 160,000 features.

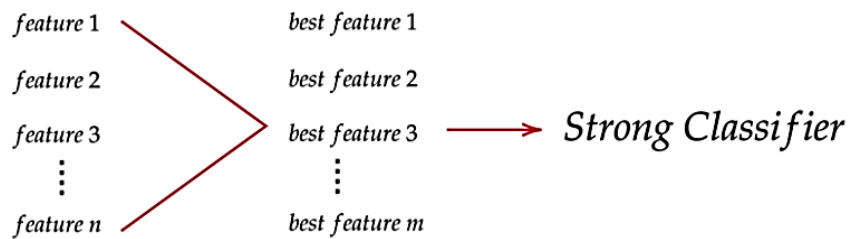


Figure 4-3 AdaBoost Classifier

What are Cascading Classifiers?

Cascading classifier is used for achieving the speed necessary for real-time face detection. It is used to quickly discard non-faces and avoid wasting precious time and computations. we divide the process of identifying a face into multiple stages. In the first stage, the subregion passes through the best features which identifies the nose bridge or eyes. When the subregion gets **Positive** then it means it is a face and if it gets a **Maybe** then it is sent to another step until the it reaches its last stage.

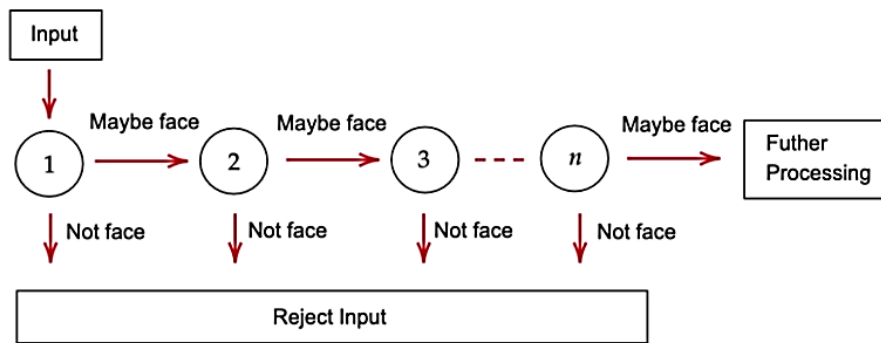


Figure 4-4 Cascading Classifiers.

4.2 Facial Recognition

Face recognition is a process of identifying or verifying the identity of a person using their face. It captures, analyzes, and compares patterns of the person's facial details that matched a human face from a digital image against a database of faces. It is used to authenticate a person through ID verification. Facial recognition algorithms measure the facial features from the given image of a person and check from the database if the given persons data is in the database.

There are three processes of recognizing the face of a person.

1. Face detection
2. Face capture
3. Face match

Face recognition systems capture an incoming image from a camera device in a two-dimensional or three-dimensional way depending on the characteristics of the algorithm. Face recognition systems use computer algorithms to pick out specific, distinctive details about a person's face. These details, such as distance between the eyes, distance between eyes or shape of the chin. Then these details are converted into a mathematical representation and compared to data on other faces collected in a face recognition database. These algorithms are designed in such a way that they only include certain details that can be used to distinguish one face from another.

Facial Recognition systems work on two processes **Identification** and **Authentication**. For example, the identification process answers to the question “Who are you?” and Authentication process answers to “Are you really who you say you are?”.

For facial recognition there are many algorithms by which we can recognize a person’s face. Some of them are:

- Eigen Faces
- 3D morphable model
- Local binary patterns histogram (LBPH)

The algorithm we used for Facial recognition is **Local Binary Patterns Histogram (LBPH)** Algorithm which is one of the best algorithms for facial detection.

4.2.1 Local Binary Pattern Histogram (LBPH)

Local Binary Pattern Histogram is a local binary operator widely used for facial recognition due to its simplicity and discriminative power. LBPH algorithm was proposed in 2006. The steps involved to achieve this are:

- Creating dataset
- Face acquisition
- Feature extraction
- Classification

4.2.2 How LBPH Algorithm Works?

In LBPH all face images are viewed as the formation of small patterns that the LBP operator could recognize successfully. The image is divided into cells (4 x 4 pixels) for the encoding of features. It is contrasted by using a clockwise or counterclockwise bearing of surrounding pixel values. The value of each neighbor's intensity is compared to the central pixel. The location is assigned a 1 or a 0 depending on the difference whether it is higher or lower than 0. The result gives a single cell an 8-bit value.

42	10	110	IS VAL > CENTRE VALUE YES = 1 NO = 0	1	0	1
6	28	50		0	0	1
90	46	28		1	1	0

Figure 4-5 LBPH Single Cell Result

4.2.3 Block Diagram for Facial Recognition

There are many different types of Facial Recognition algorithms but for every algorithm there is a block diagram by which that Facial Recognition algorithm works. Some of the things are common in every algorithm that is used for the recognition purpose.

The way Facial Recognition algorithms works is shown in the block diagram given below.

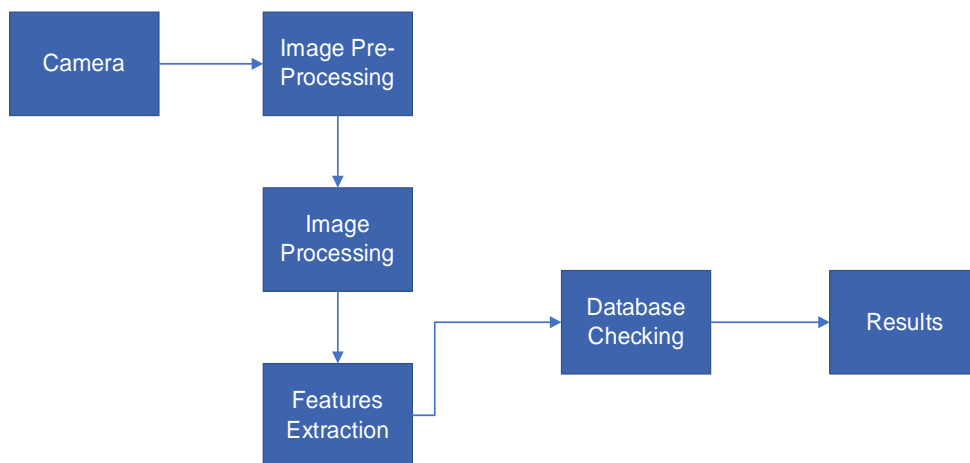


Figure 4-6 Block Diagram of Facial Recognition

4.2.4 Flow Chart of Facial Recognition

There are different types of Facial Recognition algorithms that are used in different areas for the surveillance purposes. But, for every algorithm there is a flowchart diagram that shows how that Facial Recognition algorithm works. Some of the things are common in every algorithm that are used for the recognition purposes.

The flow by which Facial Recognition algorithms go till the end of the execution is shown in the flow chart given below.

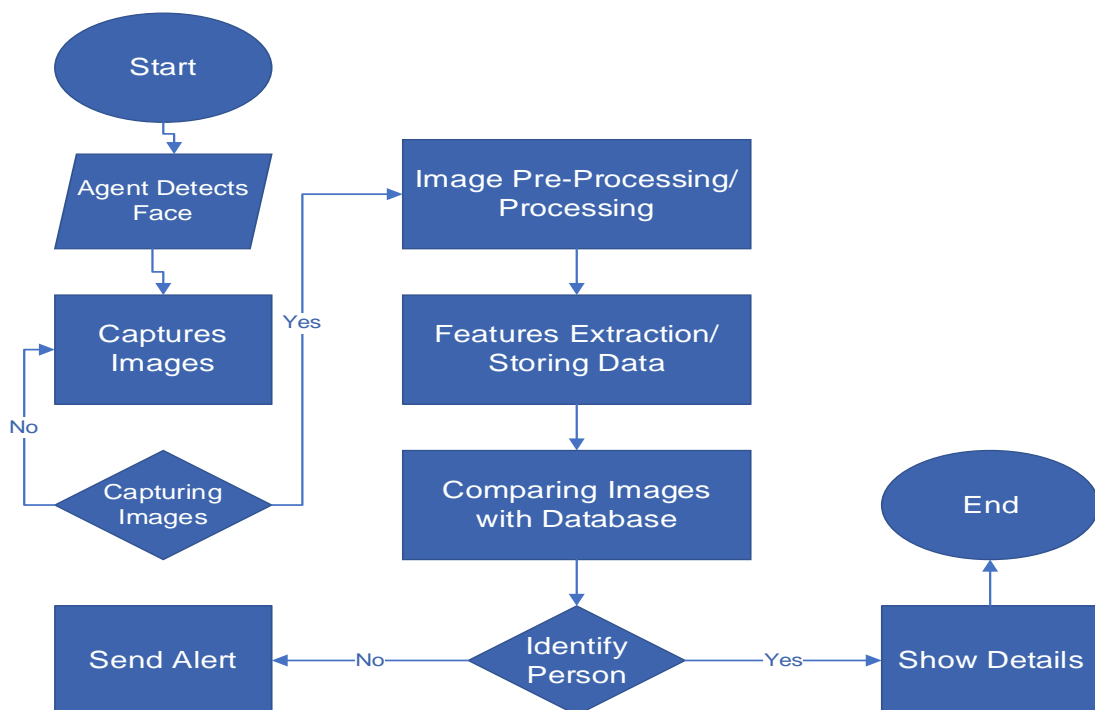


Figure 4-7 Flow Chart of Facial Recognition

4.2.5 LBPH Flowchart

There are different types of Facial Recognition algorithms that are used in different areas for the surveillance purposes. But, for every algorithm there is a flowchart diagram that shows how that Facial Recognition algorithm works. Some of the things are common in every algorithm that are used for the recognition purposes. As we are using LBPH algorithm it has its own flowchart that shows how that algorithm works.

The flow by which LBPH algorithms go till the end of the execution is shown in the flow chart given below.

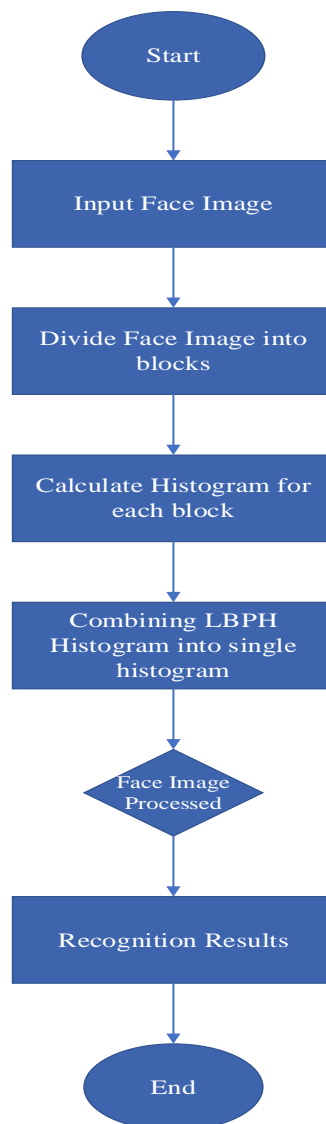


Figure 4-8 Flow Chart of LBPH

4.3 Database

A database management system (DBMS) is a package designed to define, retrieve, and manage data in a database. The DBMS captures the images and stores them in a database. It receives a command from a database administrator (DBA) and prompts the system to perform the necessary action.

There are many DBMS software we can use to perform our tasks. Some of them are:

- MySQL
- Microsoft Access
- Oracle
- SQLite

The DBMS software we used for storing and managing the images in the database is **SQLite**. As, it is easy and user-friendly software from other software.

4.3.1 SQLite

SQLite is Database Management System that contains C library. SQLite is not a client – server database engine. It follows PostgreSQL syntax. It is a good choice for embedded database software for client storage in application software. It is a widely used software for OS, browsers, and mobile phones.

4.3.2 Applications of SQLite

SQLite is a good choice for embedded database software for client storage in application software. It is vastly used in different Operating Systems (OS) of phones and laptops. Some of them are listed below.

- Blackberry OS
- Google Android
- Symbian OS
- Apple macOS (it was added as an option)

Chapter 5

Results and Limitations

In this chapter we will discuss the outcomes and results of our project. One of the deliverables of our project was to make the Smart Robot Guard autonomous. Making S-BOT autonomous means to define the movement of motors of the robot so make them move in an autonomous random movement. We will leave the robot and it will do the work by itself.

Another deliverable of our project was collision avoidance. This entailed that the S-BOT had to be able to avoid obstacles that comes in its path. We were able to implement path correction with collision detection and avoidance.

The following section explains the workflow of the whole project. For implementing autonomous Robot, we can leave it on ground and let it perform the tasks that we assigned it to do.

5.1 Components of S-BOT

As discussed in earlier chapters, the S-BOT consists of two parts:

- Hardware
- Software

5.1.1 Hardware

In hardware we have a structure of S-BOT then for the purpose of movement and controlling the robot we used Arduino UNO and a microcontroller ATmega 328n. Apart from these, in our hardware, we have SR04 Ultrasonic Sensors, DC Gear Motors, Motor driver L298, batteries and Camera.

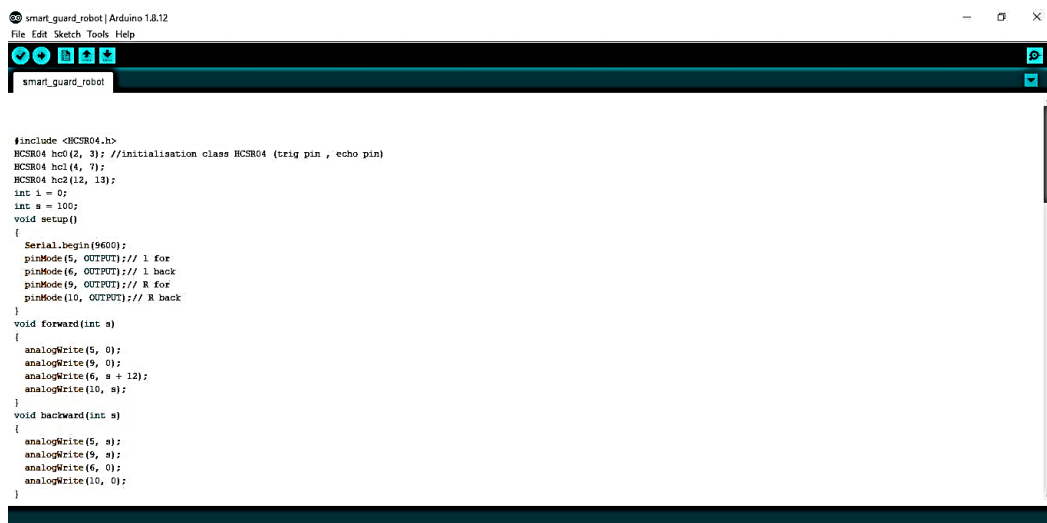
Arduino UNO

We used Arduino Uno for our system which will act as a controller for all the motors and sensors. Most of the testing with this project involved debugging code of the

Arduino UNO. For debugging process, we used various different methods from changing the values to changing the motors for getting accurate results. Oftentimes, it was hard to decide whether an issue was occurred in the hardware wiring or in the source code. We utilized the PC screen to expressly yield the code executed and other checking equipment to analyze potential equipment issues.

The greatest obstacle we needed to overcome was ensuring that each sensor works at right time to detect and avoid the obstacle that might come in its way. This was guaranteed with various calibration methods so that all the sensors and motors were working properly and were changing their position at the right time.

We used Arduino IDE compiler for Arduino coding. There were a lot of debugging issues which arrived initially but after some alterations and some coding techniques we were able to solve our problems. Through debugging and different calibration methods for calibration we were successfully getting the desired output of the S-BOT.



```
smart_guard_robot | Arduino 1.8.12
File Edit Sketch Tools Help
smart_guard_robot

#include <HCSR04.h>
HCSR04 hc0(2, 3); //initialisation class HCSR04 (trig pin , echo pin)
HCSR04 hc1(4, 7);
HCSR04 hc2(12, 13);
int i = 0;
int s = 100;
void setup()
{
  Serial.begin(9600);
  pinMode(5, OUTPUT); // 1 for
  pinMode(6, OUTPUT); // 1 back
  pinMode(9, OUTPUT); // R for
  pinMode(10, OUTPUT); // R back
}
void forward(int s)
{
  analogWrite(5, 0);
  analogWrite(9, 0);
  analogWrite(6, s + 12);
  analogWrite(10, s);
}
void backward(int s)
{
  analogWrite(5, s);
  analogWrite(9, s);
  analogWrite(6, 0);
  analogWrite(10, 0);
}
```

Figure 5-1 Arduino IDE

5.1.2 Software

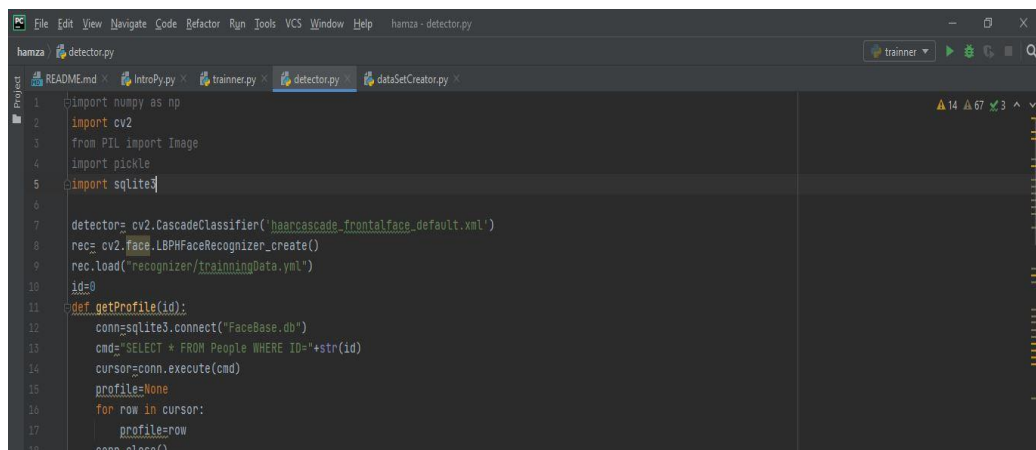
In the software part of our system, we did

- Facial Detection
- Facial Recognition
- Database

Facial Detection

Face detection is a process that identifies faces in digital images. The primary aim of face detection algorithms is to determine whether there is any face in an image or not. Face-detection algorithms focuses on the detection of frontal human faces. It is analogous to image detection in which the image of a person is matched bit by bit or pixel by pixel.

The face detection algorithm which we used for our project is Viola Jones Algorithm. We have implemented the facial detection algorithm on Python OpenCV. For that we had to download the libraries and then run the code with those libraries. There were no such issues that occurred at the time of facial detection. It was working properly and was successfully detecting faces.



```
1 import numpy as np
2 import cv2
3 from PIL import Image
4 import pickle
5 import sqlite3
6
7 detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
8 rec = cv2.Face_LBPHFaceRecognizer_create()
9 rec.load('recognizer/trainingData.yml')
10 id=0
11 def getProfile(id):
12     conn=sqlite3.connect("FaceBase.db")
13     cmd="SELECT * FROM People WHERE ID="+str(id)
14     cursor=conn.execute(cmd)
15     profile=None
16     for row in cursor:
17         profile=row
18     conn.close()
```

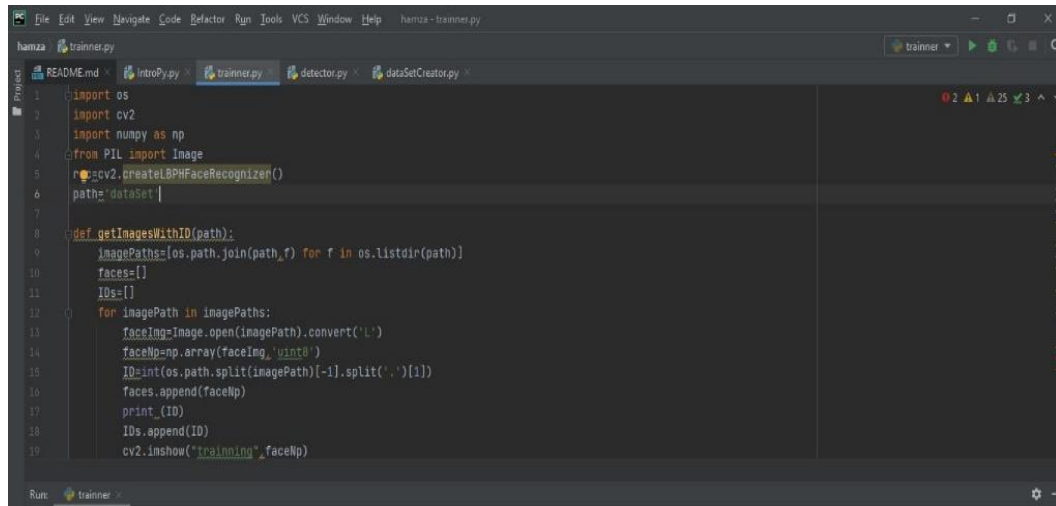
Figure 5-2 Pycharm Facial Detection code

Facial Recognition

Face recognition is a process of identifying or verifying the identity of a person using their face. It captures, analyzes, and compares patterns of the person's facial details that matched a human face from a digital image against a database of faces. It is used to authenticate a person through ID verification.

The face Recognition algorithm which we used for our project is Local Binary Pattern Histogram (LBPH). We have implemented the facial recognition algorithm on Python OpenCV. For that we had to download the libraries and then run the code with those

libraries. There were some issues that were occurring at the time of facial recognition but were resolved successfully. It was working properly and was successfully recognizing faces and crosschecking through the database.



```
1 import os
2 import cv2
3 import numpy as np
4 from PIL import Image
5 face_cascade = cv2.CascadeClassifier('')
6 path = 'dataSet/'
7
8 def getImagesWithID(path):
9     imagePaths = [os.path.join(path, f) for f in os.listdir(path)]
10    faces = []
11    IDs = []
12    for imagePath in imagePaths:
13        faceImg = Image.open(imagePath).convert('L')
14        faceNp = np.array(faceImg, dtype='uint8')
15        ID = int(os.path.splitext(imagePath)[-1].split('.')[1])
16        faces.append(faceNp)
17        print_(ID)
18        IDs.append(ID)
19        cv2.imshow("training", faceNp)
```

Figure 5-3 Pycharm Facial Recognition code

Database

A database management system (DBMS) is a package designed to define, retrieve, and manage data in a database. The DBMS captures the images and stores them in a database. It receives a command from a database administrator (DBA) and prompts the system to perform the necessary action.

The DBMS software we used is **SQLite** to store and manage the images in the database as it is easy and user-friendly software from other software. SQLite was linked with Python Open CV for crosschecking it from the database. There were no such issues that occurred at that time. It was working properly and was successfully storing and managing the images and then recognizing faces and crosschecking through the database.

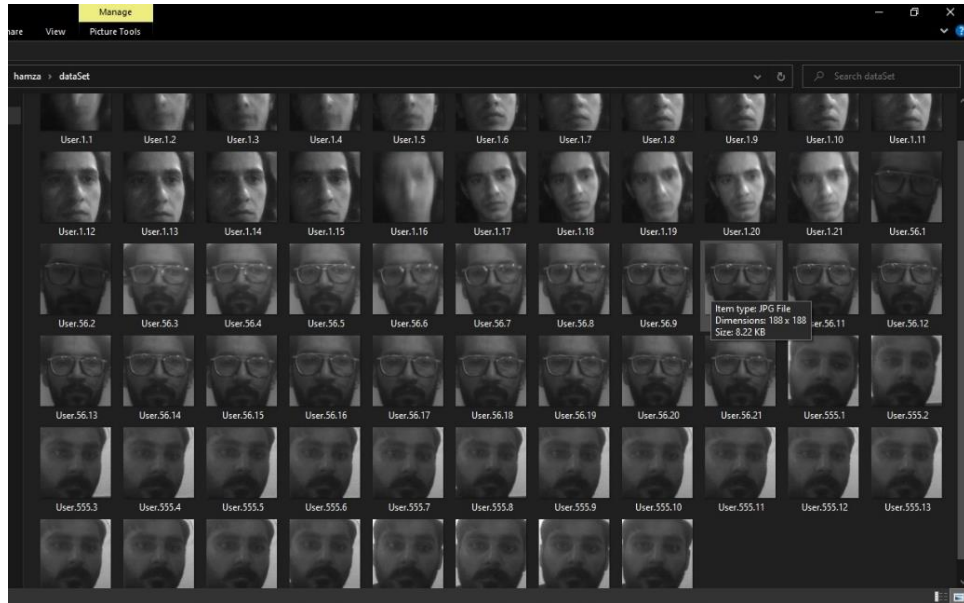


Figure 5-4 Pictures in Database

5.2 Integration

Our project is divided into different functions. To make our application functional, we have to integrate all the functions together for the desired output of our S-BOT. All the hardware was integrated with the code through Arduino IDE to the Arduino. All the motors and sensors were integrated on Arduino. We have used Ultrasonic Sensors for detecting the obstacles and avoid them. they were also integrated on Arduino through Arduino IDE software. We are using another processing machine (Haier Notebook 11 Intel Core m3) for recognition and fast processing at real time facial Recognition. These components were attached on different layers of the S-BOT to keep it clean and understandable for us as well as the examiners and supervisors who will examine the S-BOT. All the parts or the components are integrated in the code to provide the proper functionality and desired output.

5.3 Total Workflow of S-BOT

When assembling the robot there is a proper workflow by which everything is done. From designing to assembling the robot there is a workflow that everyone follows to get the end results. The robot moving autonomously in random movement, detecting and recognizing the face to testing and calibrating everything is done through a proper workflow.

The workflow of the S-BOT is shown in the figure below:

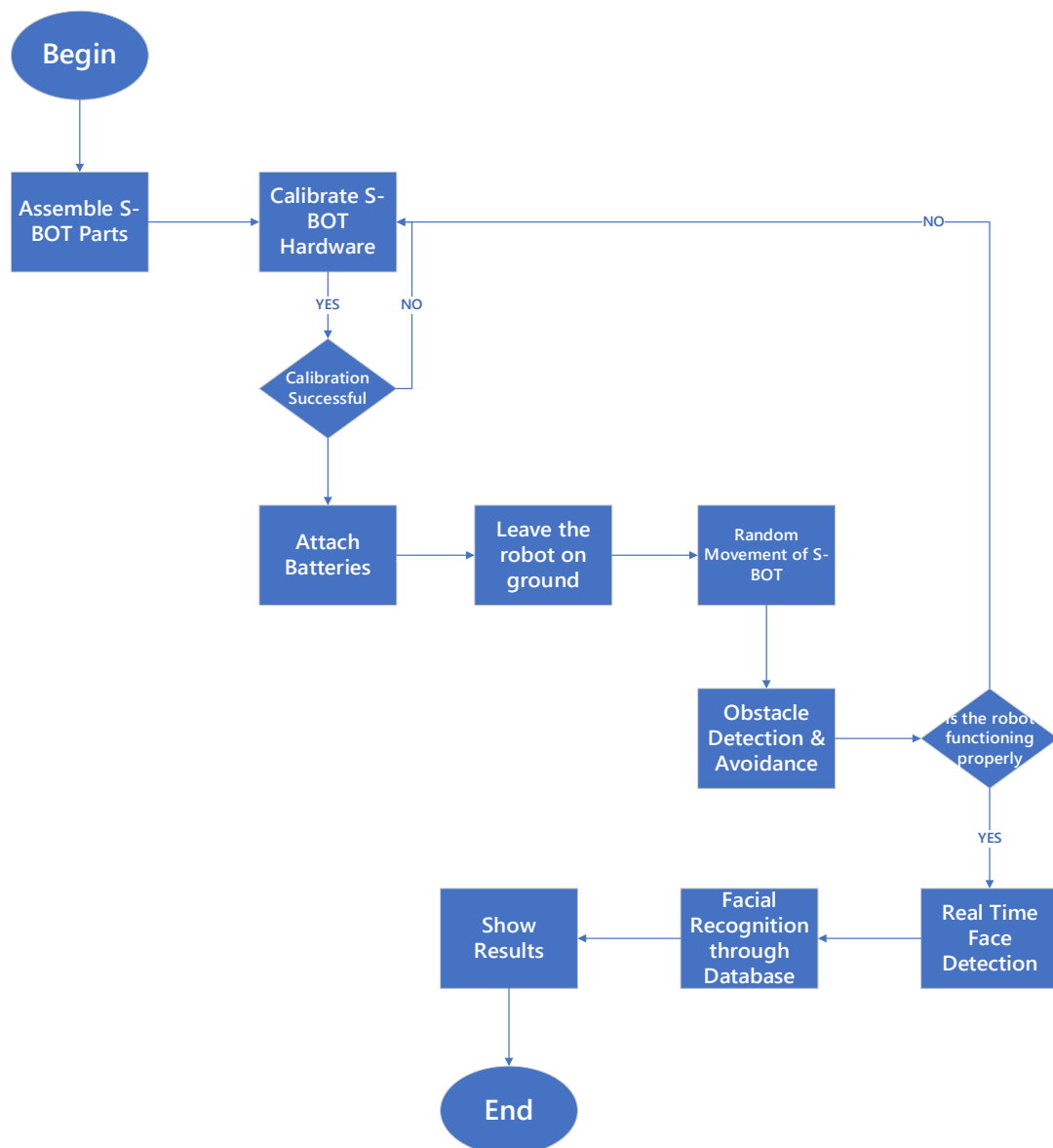


Figure 5-5 Workflow of S-BOT

5.4 Limitations

There are always some limitations of when designing a machine or robot that we have to keep in mind while using it. While keeping those limitations in mind we can get our desired output accurately and easily.

Following are the limitations and constraints that lies while building an autonomous Smart Robot Guard

- Lighting conditions for recognizing the face is one of the major limitations as we are using Noteship Webcam for feature extraction for facial recognition process. Noteship Webcam cannot be used in an outdoor environment also it has a perception up to 2-6 meter that cannot cover a wider area having an aperture of 55 degrees. This can only be resolved by using fully equipped camera such as LiDAR 2D laser scanner.
- If light in the background of the person is high, then it would be difficult for the camera to detect and recognize the face due to the low light on the persons face.
- Robot base is not able to move on rough surface or slopes as it has tires and motors that can does not support such places which are not smooth for its movement or has more slope. Moreover, it cannot climb up stairs due to its bi-directional movement.
- Another limitation is that S-BOT requires more processing speed to recognize the faces at real time processing so it cannot process fast if we use any other slow processor like Tegra TK1 processor or any other processor.
- The motors used in our robot is just to move the S-BOT and mount a camera on top of the layer so if we will put more weight that its weight carrying capacity then it might not be able to move from its place.

Chapter 6

Conclusion and Future Work

The thesis discusses the techniques and methods implemented by our team to acquire autonomous services by S-BOT. We used a structure of Aluminum and Fiber. The base of contains wheels, DC Motors and Motor Driver. We used Pycharm, Python and its library packages to implement all the desired functionality in achieving the tasks of facial recognition. Then we used Arduino IDE for autonomous Movement of S-BOT and Obstacle detection and avoidance.

6.1 Conclusion

By seeing all the scenarios and aspects then working and designing of the S-BOT we can conclude that after implementing all the work described by us can say that we are able to engineer an autonomous S-BOT. Although there are some limitations in the project which we have discussed in the previous chapter. The S-BOT can move autonomously in the area and detect the obstacles to avoid them.

Adding three Ultrasonic Sensors improved its accuracy that can avoid the obstacles that are in the front and sides of the robot. In our S-BOT we have used Arduino UNO for controlling all the parts of the robot from DC motors to Ultrasonic Sensors.

6.2 Future Work

The future of this system will be to reduce the security issues to almost zero level. The extension of this system would be that we will try to the autonomous random movement of the robot to assigning the map by map feeding algorithm and make the camera and facial recognition process wireless to reducing the wiring of the system. Other than that, we will add tracking of the unknown person to make it easy for security personnel.

In future the Smart Robot Guard that we have designed can be worked on to improve its shortcomings. Most importantly better cameras like a Lidar 2D Scanner for image processing in low light environment or any other good night vision camera that can recognize the face in low light. a TERABEE sensor can be used to detect obstacles with

more accuracy and in all directions of the S-BOT. we can use high powered motors which have more RPM and Power. We were not successful in using these sensors, motors and cameras as we could not get the funds required in time.

To improve efficiency of the system, we can use different other methods with facial recognition techniques like Skin texture analysis. Unique lines, patterns and spots are expressed as mathematical model. Different experiments have shown that this increase accuracy up to 20-25%.

6.2.1 Extension of Work in Future

After developing the robot there are some extra features that can be done on this robot like

- Map feeding
- Suspect tracking

6.2.2 Map Feeding

This would also greatly simplify the path feeding or map feeding technique required by the Smart Robot. This technique enables a robot to develop and maintain a model of its environment based on the spatial information recorded over time. This spatial information is gathered from external sensors, however internal sensor is also important to give robots current position and change of location in the map. 2D maps can be generated and further processing can be done on it. These are usually preferred unless there are some constraints to use 3d mapping.

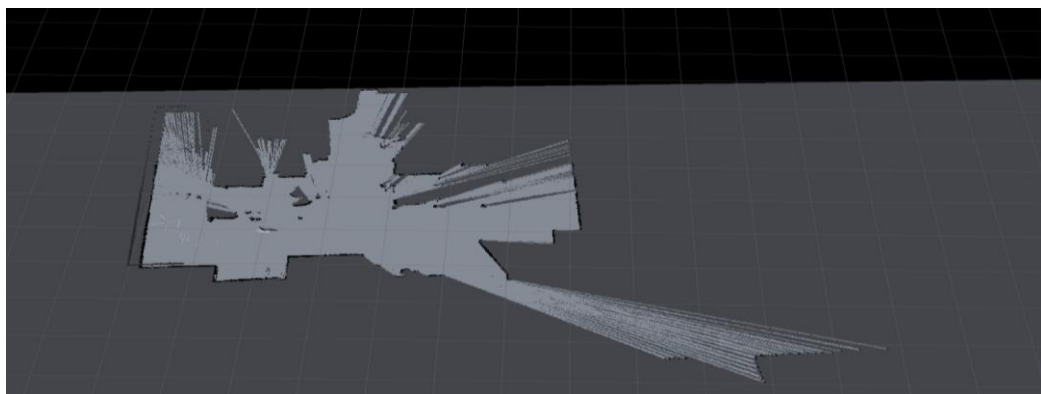


Figure 6-1 2D Map feeding of Robot

There are some places where we can use 3D Mapping.

3D Mapping

3D maps are rarely used because of the processing time and system requirements. Real Time Appearance based Mapping is discussed.

Real Time Appearance based Mapping

Rather than relying on accurate sensing techniques appearance-based mapping provides a low-cost solution to create maps for low cost and resource limited robots. It uses appearance signature to identify places and objects. For appearance-based mapping and localization loop closure detection is essential. Loop closure detection is the process of determining whether the data coming is from 0 and size for real-time long-term operations and large environments. Appearance based mapping are good for loop closure detection in range based metric mapping and for also topological mapping for path planning, navigation and exploration.

Real time appearance-based mapping is an RGBD graph-based mapping approach. It uses appearance-based loop closure detection. This technique uses bag of words approach and determines that the new upcoming image is from a previous location or a new one. When a new loop closure is detected, a constraint is added in graph, a graph optimizer minimizes the error in map.

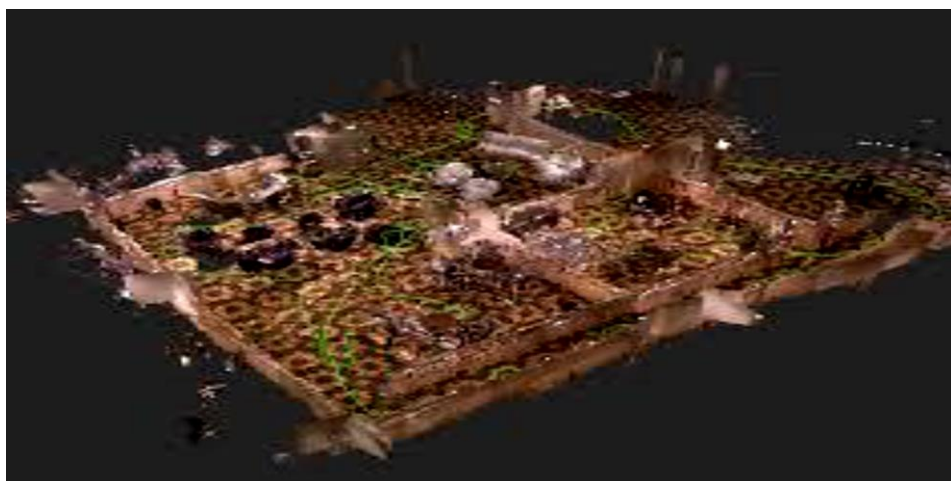


Figure 6-2 3D Map feeding of Robot

6.2.3 Suspect Tracking

When a person is recognized if persons information is not in the database, then it will give an alert and track the suspect where he is. There are different algorithms that can be used for the suspect tracking like Joint Probabilistic Data Association Filter (JPDAF) algorithm for real time tracking and Probabilistic Multi-Hypothesis Tracking (PMHT) algorithm for real time human tracking.

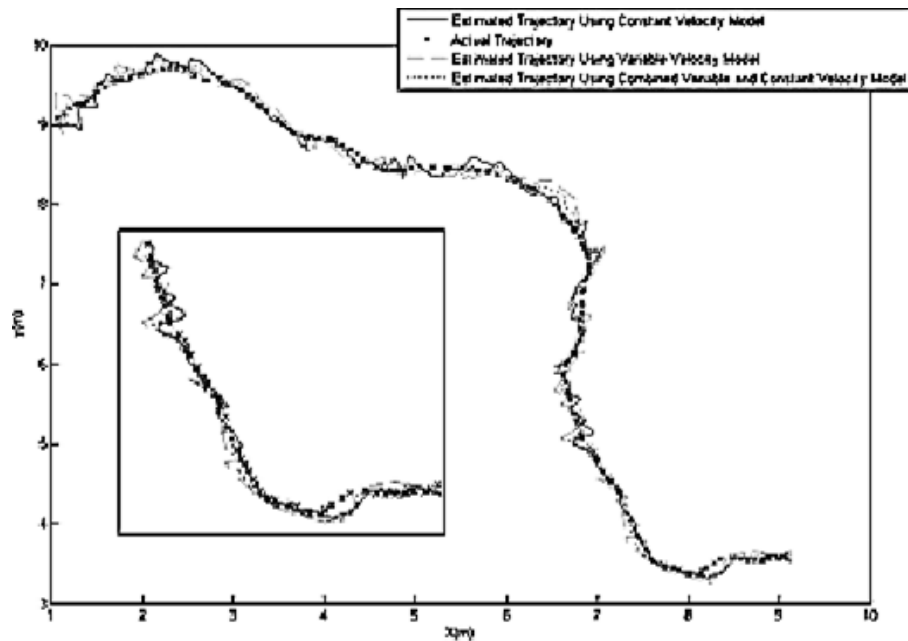


Figure 6-3 2D Target Tracking through JPDAF

In the given figure above, it is showing the track at which the robot was moved to for the suspect tracking purpose through JPDAF suspect tracking algorithm.

The next figure shows the multiple tracks at which the robot was moving for the purpose of suspect tracking through PMHT suspect tracking algorithm.

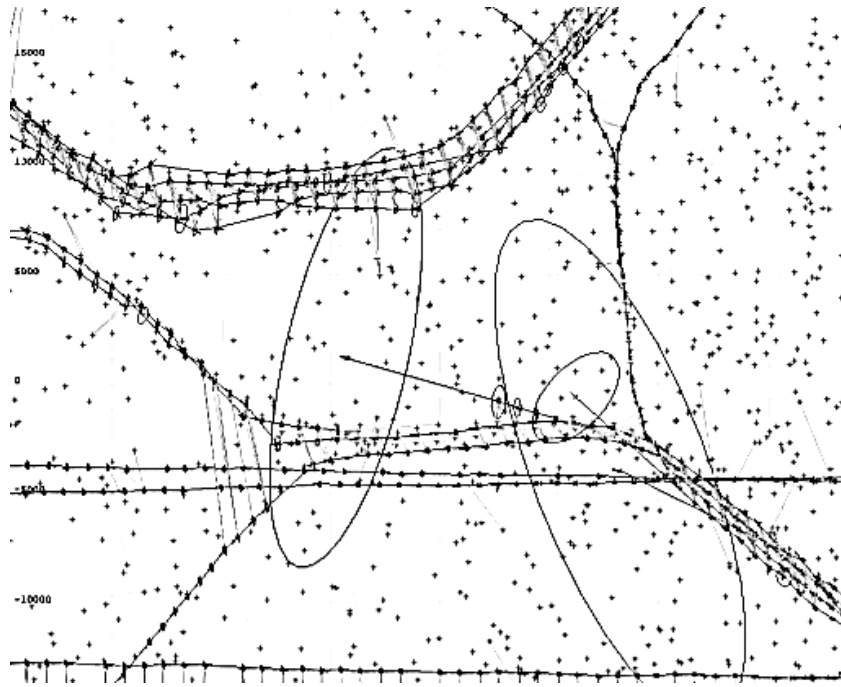


Figure 6-4 Zoomed in Target tracking through PMHT

Automated Robots has become a vast field of Robotics and has made itself as a doorway to explore and enhance the maximum results from a machine to make them perform humanly tasks. Robotic has become one of the best domains that encourages researchers and passionate individuals to perform and learn from each other's experiences in the field of Domestic and Service Robots. These robots can work and d humanly tasks by different modifications and enhancements in the codes by which it will be working.

REFERENCES

- [1] Chitta, Sachin, Ioan Sucan, and Steve Cousins. "Moveit! " *Robotics & Automation Magazine*, IEEE 19.1 (2012): 18-19.
- [2] "Arduino Reference", *Arduino.cc*,2019. [Online].
- [3] Lamere, P., Kwok, P., Walker, W., Gouvêa, E. B., Singh, R., Raj, B., & Wolf, P. (2003, September). Design of the CMU sphinx-4 decoder.
- [4] Brooks, R. (1986). A robust layered control system for a mobile robot. *IEEE journal on robotics and automation*, 2(1), 14-23.
- [5] Tsumura, N., Ojima, N., Sato, K., Shiraishi, M., Shimizu, H., Nabeshima, H., ... & Miyake, Y. (2003). Image-based skin color and texture analysis/synthesis by extracting hemoglobin and melanin information in the skin. *ACM Transactions on Graphics (TOG)*, 22(3), 770-779.
- [6] Kavraki, L. E., Svestka, P., Latombe, J. C., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE transactions on Robotics and Automation*, 12(4), 566-580.
- [7] Warren, Charles W. "Global path planning using artificial potential fields." *Robotics and Automation*, 1989. Proceedings. 1989 IEEE International Conference on. IEEE, 1989
- [8] Fazlur Rahman, M. and Adhy Sasongko, R. (2018). Obstacle Avoidance using Ultrasonic Sensor. *Journal of Physics: Conference Series*, 1005, p.012037.
- [9] Moravec, Hans Peter. "Robot". *Encyclopedia Britannica*, 4 Feb. 2021.
- [10] Horn, Berthold, Berthold Klaus, and Paul Horn. *Robot vision*. MIT press, 1986.
- [11] Asada, Haruhiko, and J-JE Slotine. *Robot analysis and control*. John Wiley & Sons, 1986.
- [12] Liu, James NK, Meng Wang, and Bo Feng. "iBotGuard: an Internet-based intelligent robot security system using invariant face recognition against

intruder." *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 35.1 (2005): 97-105.

[13] Miwa, Hiroyasu, et al. "Development of a new human-like head robot WE-4." *IEEE/RSJ international conference on intelligent robots and systems*. Vol. 3. IEEE, 2002.

[14] Hu, C., et al. "Efficient face and gesture recognition techniques for robot control." *CCECE 2003-Canadian Conference on Electrical and Computer Engineering. Toward a Caring and Humane Technology (Cat. No. 03CH37436)*. Vol. 3. IEEE, 2003.

[15] Khatib, Oussama. "Real-time obstacle avoidance for manipulators and mobile robots." *Autonomous robot vehicles*. Springer, New York, NY, 1986. 396-404.

[16] Bhagat, Kirti, et al. "Obstacle avoidance robot." *International Journal of Science, Engineering and Technology Research (IJSETR)* 5.2 (2016): 439-442.

[17] Vairavan, R., et al. "Obstacle Avoidance Robotic Vehicle Using Ultrasonic Sensor, Arduino Controller." *International Research Journal of Engineering and Technology* 2.5 (2018): 2140-2143.

[18] Basyal, Lochan, Sandeep Kaushal, and Gurjeet Singh. "Facial Recognition Robot with Real Time Surveillance and Automation." *International Journal of Creative Research Thoughts* 6.1 (2018): 2320-2882.

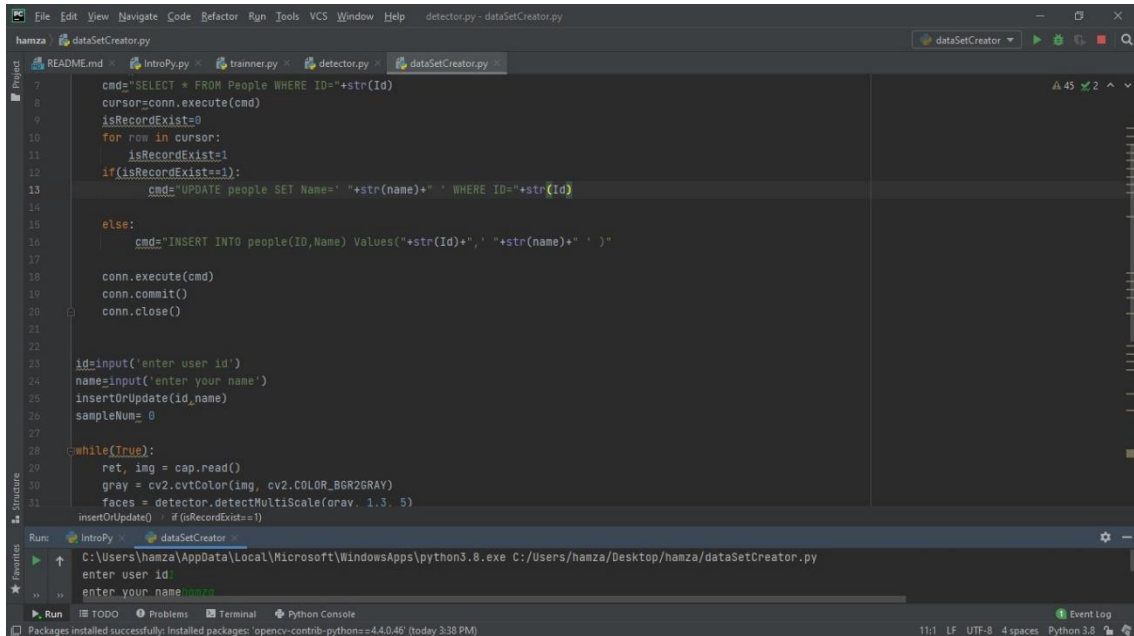
[19] Oltean, Stelian-Emilian. "Mobile robot platform with arduino uno and raspberry pi for autonomous navigation." *Procedia Manufacturing* 32 (2019): 572-577.

ABBREVIATIONS

S-BOT	Smart Robot
RPM	Rotation Per Minute
SQL	Structured Query Language
PR2	Programmable Robot 2
OS	Operating System
DC	Direct Current
US	Ultrasonic Sensor
DBA	Database Administrator
IDE	Integrated Development Environment
ID	Identification
Kg	Kilogram
KHz	KiloHertz
OpenCV	Open Computer Vision
LBPH	Local Binary Pattern Histogram
KUKA	Keller und Knappich Augsburg
PMHT	Probabilistic Multi-Hypothesis Tracking
JPDAF	Joint Probabilistic Data Association Filter
LIDAR	Light Detection and Ranging

ANNEXURE

Data Set Creation

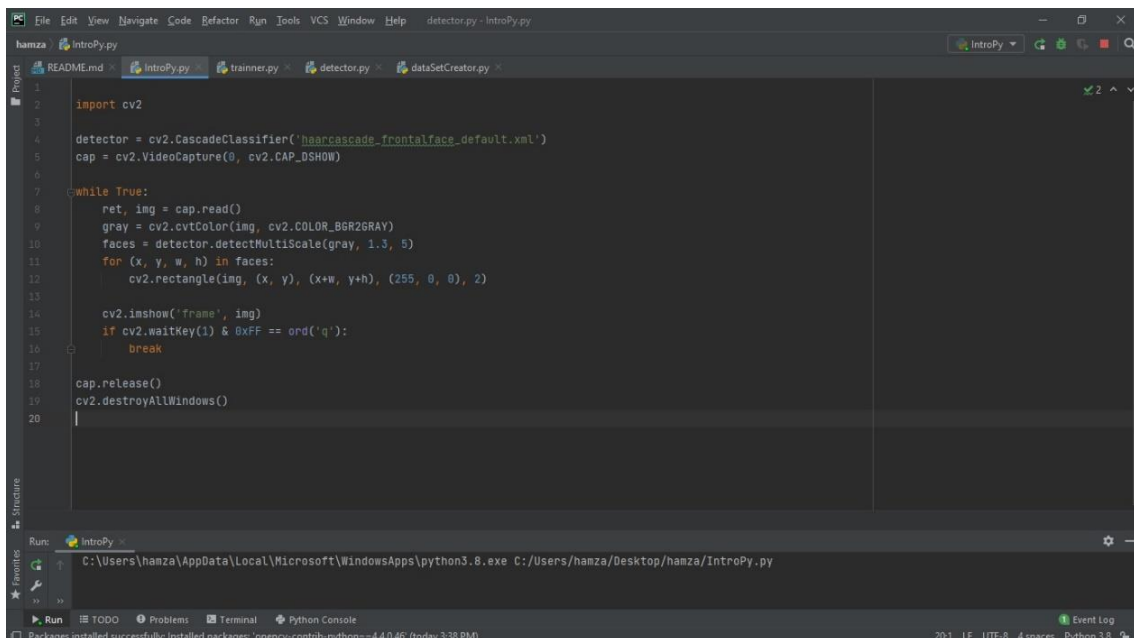


```
7 cmd="SELECT * FROM People WHERE ID="+str(Id)
8 cursor=conn.execute(cmd)
9 isRecordExist=0
10 for row in cursor:
11     isRecordExist=1
12 if(isRecordExist==1):
13     cmd="UPDATE people SET Name="+str(name)+" WHERE ID="+str(Id)
14
15 else:
16     cmd="INSERT INTO people(ID,Name) Values("+str(Id)+','+"str(name)+' )"
17
18 conn.execute(cmd)
19 conn.commit()
20 conn.close()
21
22
23 id=input('enter user id')
24 name=input('enter your name')
25 insertOrUpdate(id_name)
26 sampleNum= 0
27
28 while(True):
29     ret, img = cap.read()
30     gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
31     faces = detector.detectMultiScale(gray, 1.3, 5)
32     insertOrUpdate() if (isRecordExist==1)
```

Run: IntroPy - dataSetCreator -
C:\Users\hamza\AppData\Local\Microsoft\WindowsApps\python3.8.exe C:/Users/hamza/Desktop/hamza/dataSetCreator.py
enter user id:
enter your name:hamza

Event Log
Packages installed successfully: Installed packages: 'opencv-contrib-python==4.4.0.46' (today 3:38 PM) 11:1 LF UTF-8 4 spaces Python 3.8

Testing of Code



```
1 import cv2
2
3
4 detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
5 cap = cv2.VideoCapture(0, cv2.CAP_0SHOW)
6
7 while True:
8     ret, img = cap.read()
9     gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
10    faces = detector.detectMultiScale(gray, 1.3, 5)
11    for (x, y, w, h) in faces:
12        cv2.rectangle(img, (x, y), (x+w, y+h), (255, 0, 0), 2)
13
14    cv2.imshow('frame', img)
15    if cv2.waitKey(1) & 0xFF == ord('q'):
16        break
17
18 cap.release()
19 cv2.destroyAllWindows()
20
```

Run: IntroPy -
C:\Users\hamza\AppData\Local\Microsoft\WindowsApps\python3.8.exe C:/Users/hamza/Desktop/hamza/IntroPy.py

Event Log
Packages installed successfully: Installed packages: 'opencv-contrib-python==4.4.0.46' (today 3:38 PM) 20:1 LF UTF-8 4 spaces Python 3.8

Facial Detection Algorithm

```
1 import numpy as np
2 import cv2
3 from PIL import Image
4 import pickle
5 import sqlite3
6
7 detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
8
9 rec = cv2.face.LBPHFaceRecognizer_create()
10 rec.load('recognizer/trainingData.yml')
11 id=0
12 def getProfile(id):
13     conn=sqlite3.connect("FaceBase.db")
14     cmd="SELECT * FROM People WHERE ID="+str(id)
15     cursor=conn.execute(cmd)
16     profile=None
17     for row in cursor:
18         profile=row
19     conn.close()
20     return profile
21
22 cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)
23 font = cv2.cv.InitFont(cv2.cv.FONT_HERSHEY_COMPLEX_SMALL, 5, 1, 0, 4)
24 while(True):
25     ret, img = cap.read()
26     getProfile()
```

Run: IntroPy - detector - Process finished with exit code 1

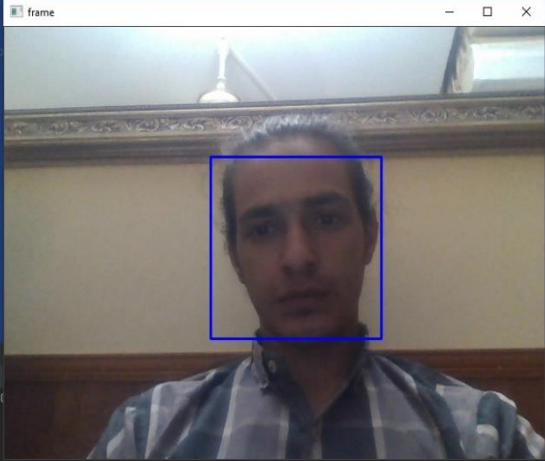
```
1 import cv2
2
3 detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
4 cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)
5
6 while True:
7     ret, img = cap.read()
8     gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
9     faces = detector.detectMultiScale(gray, 1.3, 5)
10    for (x, y, w, h) in faces:
11        cv2.rectangle(img, (x, y), (x+w, y+h), (255, 0, 0), 2)
12
13    cv2.imshow('frame', img)
14    if cv2.waitKey(1) & 0xFF == ord('q'):
15        break
16
17 cap.release()
18 cv2.destroyAllWindows()
```

Run: IntroPy - C:\Users\hamza\AppData\Local\Microsoft\WindowsApps\pyt

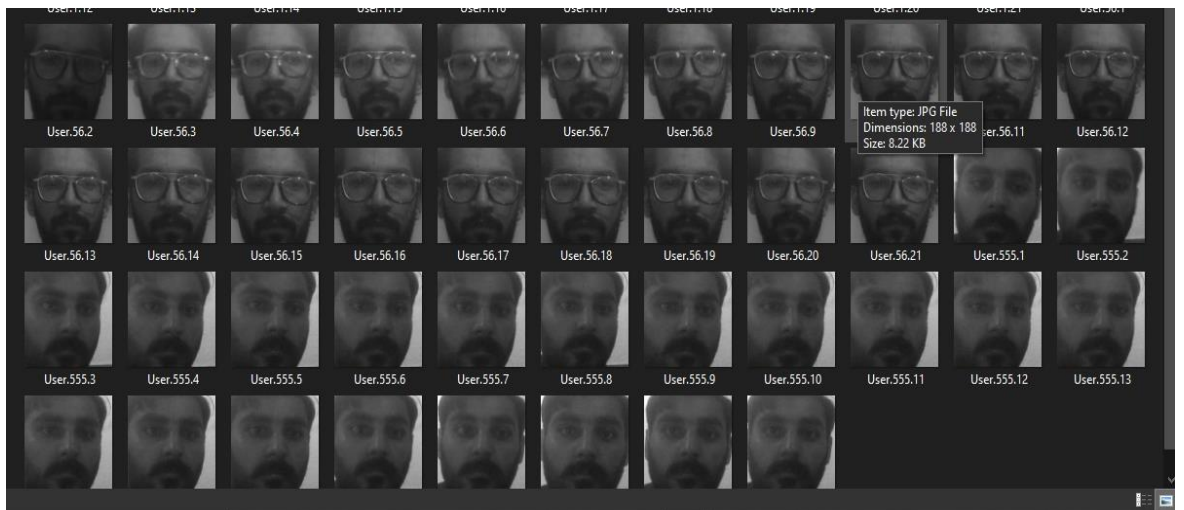
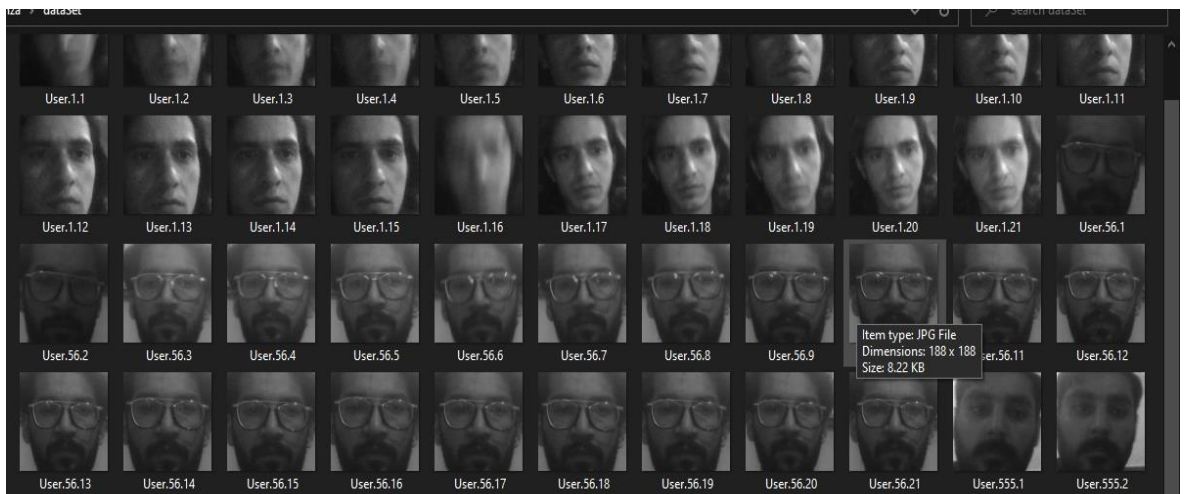
Facial Recognition Algorithm

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help hamza - dataSetCreator.py
hamza dataSetCreator.py IntroPy.py dataSetCreator.py detector.py
import sqlite3
cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)
detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
def insertOrUpdate(Id, Name):
    conn=sqlite3.connect("FaceBase.db")
    cmd="SELECT * FROM People WHERE ID="+str(Id)
    cursor=conn.execute(cmd)
    isRecordExist=0
    for row in cursor:
        isRecordExist=1
    if(isRecordExist==1):
        cmd="UPDATE people SET Name=' "+str(name)+" ' WHERE ID="+str(Id)
    else:
        cmd="INSERT INTO people(ID,Name) Values("+str(Id)+", '"+str(name)+" ')"
    conn.execute(cmd)
    conn.commit()
    conn.close()
insertOrUpdate()
Run: dataSetCreator
C:\Users\hamza\AppData\Local\Microsoft\WindowsApps\python3.8.exe C:/Users/hamza/Desktop/hamza/dataSetCreator.py
enter user id 0
enter your name mujiz
Process finished with exit code 0
PyCharm 2020.3.5 available
Update...
```

```
File Edit View Navigate Code Refactor Run Tools VCS Window Help hamza - IntroPy.py
hamza IntroPy.py IntroPy.py dataSetCreator.py detector.py
import cv2
detector = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')
cap = cv2.VideoCapture(0, cv2.CAP_DSHOW)
while True:
    ret, img = cap.read()
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    faces = detector.detectMultiScale(gray, 1.3, 5)
    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x+w, y+h), (255, 0, 0), 2)
    cv2.imshow('frame', img)
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break
cap.release()
cv2.destroyAllWindows()
while True:
Run: IntroPy
C:\Users\hamza\AppData\Local\Microsoft\WindowsApps\python3.8.exe
frame
PyCharm 2020.3.5 available
Update...
```



Pictures Stored in Database



Arduino IDE Code

```
smart_guard_robot | Arduino 1.8.12
File Edit Sketch Tools Help

smart_guard_robot

#include <HCSR04.h>
HCSR04 hc0(2, 3); //initialisation class HCSR04 (trig pin , echo pin)
HCSR04 hc1(4, 7);
HCSR04 hc2(12, 13);
int i = 0;
int s = 100;
void setup()
{
  Serial.begin(9600);
  pinMode(5, OUTPUT); // L for
  pinMode(6, OUTPUT); // L back
  pinMode(9, OUTPUT); // R for
  pinMode(10, OUTPUT); // R back
}
void forward(int s)
{
  analogWrite(5, 0);
  analogWrite(9, 0);
  analogWrite(6, s + 12);
  analogWrite(10, s);
}
void backward(int s)
{
  analogWrite(5, s);
  analogWrite(9, s);
  analogWrite(6, 0);
  analogWrite(10, 0);
}
```

```
smart_guard_robot | Arduino 1.8.12
File Edit Sketch Tools Help

smart_guard_robot

}
void right(int s)
{
  analogWrite(5, 0);
  analogWrite(9, 0);
  analogWrite(6, s);
  analogWrite(10, 0);
}
void left(int s)
{
  analogWrite(5, 0);
  analogWrite(9, 0);
  analogWrite(6, 0);
  analogWrite(10, s);
}
void stop()
{
  analogWrite(5, 0);
  analogWrite(9, 0);
  analogWrite(6, 0);
  analogWrite(10, 0);
}
void loop()
{
  if (hc0.dist() > 30 && hc1.dist() > 30 && hc2.dist() > 30)
  {
    forward(s);
  }
  if (hc0.dist() < 30 && hc1.dist() > 30 && hc2.dist() > 30)
  {
    right(s);
  }
}
```



```
smart_guard_robot | Arduino 1.8.12
File Edit Sketch Tools Help

smart_guard_robot
right(s);
}
if (hc0.dist() > 30 && hc1.dist() > 30 && hc2.dist() < 30)
{
left(s);
}

if (hc1.dist() < 30)
{
if (i == 0)
{
stop();
delay(1000);
backward(s);
delay(1000);
stop();
delay(1000);
right(s);
delay(5000);
stop();
delay(1000);
forward(s);
delay(5000);
stop();
delay(1000);
right(s);
delay(5000);
stop();
delay(1000);
}
i++;
}
}
```

```
smart_guard_robot | Arduino 1.8.12
File Edit Sketch Tools Help

smart_guard_robot

stop();
delay(1000);
backward(s);
delay(1000);
stop();
delay(1000);
left(s);
delay(5000);
stop();
delay(1000);
forward(s);
delay(5000);
stop();
delay(1000);
left(s);
delay(5000);
stop();
delay(1000);

i = 0;
}
}

Serial.println( hc0.dist() );
Serial.println( hc1.dist() );
Serial.println( hc2.dist() );
Serial.println( "-----" );
delay(50);
} //return curent distance in serial
```