

[SMART CCTV HOME CAMERA WITH FACE RECOGNITION]



UNIVERSITI TEKNIKAL MALAYSIA MELAKA

[SMART CCTV HOME CAMERA WITH FACE RECOGNITION]

[THENNARASU A/L PARAMASIVAN]



This report is submitted in partial fulfillment of the requirements for the Bachelor of [Computer Science (Computer Networking)] with Honours.


UNIVERSITI TEKNIKAL MALAYSIA MELAKA

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2021

DECLARATION


I hereby declare that this project report entitled
[SMART CCTV HOME CAMERA WITH FACE RECOGNITION]
is written by me and is my own effort and that no part has been plagiarized
without citations.

STUDENT :  _____ Date : 06/9/2021
(THENNARASU A/L PARAMASIVAN)



UNIVERSITI TEKNIKAL MALAYSIA MELAKA

I hereby declare that I have read this project report and found
this project report is sufficient in term of the scope and quality for the award of
Bachelor of [Computer Science (Software Development)] with Honours.

SUPERVISOR :  _____ Date : 8/9/21
(PM DR MOHD FAIZAL ABDOLLAH)

DEDICATION

I would be so grateful because God still give me strength and passion to finish this project.

A special feeling to gratitude to my loving parents whose words of encouragement and push for tenacity ring in my ear.



Last but not least, special thanks to my lovely and supportive, Pm Dr Mohd Faizal Abdollah. Thanks for all motivation, for being there for me throughout the entire Progress to complete this project.

ACKNOWLEDGEMENTS

Primarily I would thanks to God for being able to complete this project with success. Then I would like to thank my supervisor, Pm Dr Mohd Faizal Abdollah, whose valuable guidance has been ones that helped me patch this project and make it full proof success his suggestions and his instructions has served as the major contributor towards the completion of this project.

Then I would like to thank my parents and friends who helped me with their valuable suggestions and guidance has been helpful in various phases of the completion of the project.



ABSTRACT

Most of smart CCTV home camera that have face recognition are having problem in recognize people properly. The current algorithm technique that used in the face recognition could not recognize people properly if they are stranger or thieves at in front of the house door. This is very worrying if the CCTV could not recognize people. With the proper algorithm for face recognition, it can indirectly solve this problem. This cost for running this project is minimal and the beginner is also can easily build on his own. Devices like Raspberry Pi board, Raspberry Pi High quality camera and Raspberry Pi 6mm lens camera are the main devices for this project. Meanwhile, the system will send notification to the user if stranger or unknown people are detected. The system also easy to use which allow users to easily understand and use the face recognition camera. Finally, the users can use the proper algorithm for the CCTV camera and the camera can recognize people properly. By having this smart CCTV camera residents can avoid burglary problem getting worst among the residents.

ABSTRAK

Kebanyakan kamera rumah CCTV pintar yang mempunyai pengecaman wajah menghadapi masalah mengenali orang dengan betul. Teknik algoritma semasa yang digunakan dalam pengecaman wajah tidak dapat mengenali orang dengan betul jika mereka orang tidak dikenali atau pencuri di depan pintu rumah. Ini sangat membimbangkan jika CCTV tidak dapat mengenali orang. Dengan algoritma yang tepat untuk pengecaman wajah, secara tidak langsung ia dapat menyelesaikan masalah ini. Kos untuk menjalankan projek ini adalah minimum dan pemula juga dapat membina sendiri dengan mudah. Peranti seperti papan Raspberry Pi, kamera berkualiti tinggi Raspberry Pi dan kamera lensa Raspberry Pi 6mm adalah peranti utama untuk projek ini. Sementara itu, sistem akan menghantar pemberitahuan kepada pengguna sekiranya orang asing atau orang yang tidak dikenali dikesan. Sistem ini juga mudah digunakan yang membolehkan pengguna memahami dan menggunakan kamera pengenalan wajah dengan mudah. Akhirnya, pengguna dapat menggunakan algoritma yang tepat untuk kamera CCTV dan kamera dapat mengenali orang dengan betul. Dengan adanya kamera CCTV pintar ini, penghuni dapat mengelakkan masalah pencurian menjadi lebih teruk di kalangan penghuninya.

TABLE OF CONTENTS

	PAGE
DECLARATION.....	II
DEDICATION.....	III
ACKNOWLEDGEMENTS.....	IV
ABSTRACT	V
ABSTRAK	VI
TABLE OF CONTENTS.....	VII
LIST OF TABLES	XIV
LIST OF FIGURES	XV
LIST OF ABBREVIATIONS	XVIII
CHAPTER 1: INTRODUCTION.....	1
1.0 Introduction.....	1
1.1 Project Background.....	1
1.2 Problem Statement	3
1.3 Objective	4
1.4 Scope.....	4
1.5 Project Contribution.....	5
1.6 Report Organization.....	6
1.7 Summary	7

CHAPTER 2: LITERATURE REVIEW.....	8
2.0 Introduction.....	8
2.1 Framework.....	8
2.2 CCTV.....	10
2.2.1 Advantage of CCTV.....	11
2.2.2 Weakness of CCTV.....	12
2.2.3 Importance of smart CCTV camera in home.....	13
2.2.4 Relation between Smart CCTV and face recognition.....	14
2.3 Smart CCTV with face recognition.....	15
2.3.1 Face recognition algorithms.....	15
2.3.2 Face recognition algorithms architecture.....	21
2.3.2.1 Eigenfaces Architecture.....	21
2.3.2.2 Fisherfaces Architecture.....	22
2.3.2.3 Local Binary Patterns (LBP) Architecture.....	24
2.3.2.4 Scale-Invariant Feature Transform (SIFT) Architecture.....	25
2.3.2.5 Convolutional Neural Network (CNN) Architecture.....	26
2.3.2.6 OpenFace Architecture.....	27
2.3.2.7 Histogram of Oriented Gradients Architecture.....	28
2.4 Related Work.....	29
2.4.1 Approach.....	29
2.4.2 Device.....	30

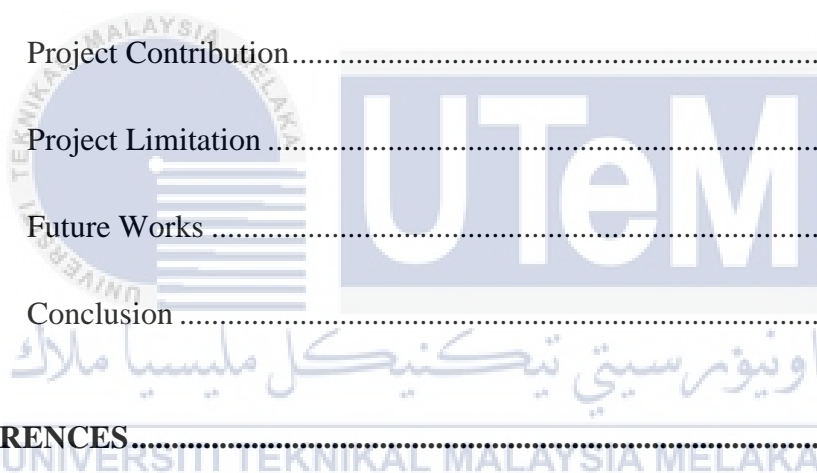
2.4.2.1	Raspberry Pi.....	30
2.4.2.2	CCTV.....	31
2.4.3	Platforms.....	31
2.4.3.1	Web-based	31
2.4.3.2	E-mail	31
2.4.4	Extra Function	32
2.4.4.1	Intruder Alert Notification.....	32
2.4.5	Architecture of current related work.....	32
2.4.5.1	Architecture of Face recognition Security Applications in Home	32
2.4.5.2	Architecture of Smart Security System for Sensitive Area	33
2.4.5.3	Architecture pf Web and Mobile based Facial Recognition.....	34
2.5	Proposed Solutions.....	36
2.5.1	Parameters.....	37
2.5.2	Proposed parameters, algorithm and CCTV	38
2.5.3	Proposed Architecture	39
2.6	Critical Review	39
2.7	Conclusion	41
CHAPTER 3: METHODOLOGY		42
3.0	Introduction.....	42
3.1	Methodology	42
3.1.1	Software Development Life Cycle (SDLC)	43

3.1.2	Waterfall Model.....	43
3.1.2.1	Requirement Analysis Phase	44
3.1.2.2	System Design Phase	46
3.1.2.3	Implementation Phase.....	47
3.1.2.4	System Testing Phase	48
3.1.2.5	System Deployment Phase	48
3.1.2.6	System Maintenance Phase.....	49
3.1.3	Project Milestones	49
3.1.4	Project Gantt Chart	50
3.2	Conclusion	51
CHAPTER 4: ANALYSIS AND DESIGN.....		52
4.0	Introduction.....	52
4.1	Problem Analysis.....	52
4.2	Requirement Analysis.....	52
4.2.1	Data Requirement	53
4.2.1.1	Images of valid user face	53
4.2.2	Software Requirement	53
4.2.2.1	Raspberry Pi Operating System.....	54
4.2.3	Hardware Requirement.....	54
4.2.3.1	Raspberry Pi 4 Model B	54
4.2.3.2	Raspberry Pi High Quality Camera	55

4.2.3.3	Raspberry Pi 6mm Lens Camera	56
4.2.3.4	MicroSD (32GB)	57
4.3	Detailed Design.....	58
4.3.1	Circuit Design	58
4.3.2	Architecture Design	59
4.3.3	General Flow Chart.....	60
4.3.3.1	Flow Chart for OpenCV module	61
4.3.3.2	Flow Chart for HOG algorithm module	62
4.3.3.3	Flow Chart for Modified HOG algorithm module	63
4.4	Conclusion	65
CHAPTER 5: IMPLEMENTATION.....		66
5.0	Introduction.....	66
5.1	Software Development Environment Setup.....	66
5.2	Software Configuration Management.....	69
5.2.1	Configuartion environment setup	69
5.2.2	OpenCV Coding in Raspberry Pi Board.....	69
5.2.3	Histogram of Oriented Gradient (HOG) algorithm coding in Raspberry Pi Board.....	71
5.2.4	Gmail notifications coding ib Raspberry Pi Board.....	75
5.3	Hardware Configuration Management	76
5.3.1	Connection between Raspberry Pi board and Raspberry Pi High Quality Camera.....	78

5.3.2	Connection successful between Raspberry Pi and Raspberry Pi camera.....	79
5.4	Implementation Status	80
5.5	Conclusion	81
CHAPTER 6: TESTING		82
6.0	Introduction.....	82
6.1	Test Plan.....	82
6.1.1	Test Environment.....	82
6.1.2	Test Schedule	83
6.2	Test Strategy	83
6.2.1	Classes of Tests.....	84
6.3	Test Design	84
6.3.1	Test Descriptuon.....	84
6.3.1.1	Communication between Raspberry Pi board and Raspberry Pi High Quality Camera test	84
6.3.1.2	Communication between Raspberry Pi High Quality Camera and OpenCV coding	85
6.3.1.3	Accuracy of HOG algorithm in recognizing face test	86
6.3.1.4	Testing email notifications	87
6.3.2	Test Data.....	87
6.4	Test Result and Analysis.....	87
6.4.1	Test of TC_1	88

6.4.2	Test of TC_2	90
6.4.3	Test of TC_3	92
6.4.4	Test of TC_4	97
6.4.5	Analysis Summary	99
6.5	Conclusion	99
CHAPTER 7: PROJECT CONCLUSION		100
7.0	Introduction	100
7.1	Project Summarization	100
7.2	Project Contribution	101
7.3	Project Limitation	101
7.4	Future Works	102
7.5	Conclusion	102
REFERENCES		104



LIST OF TABLES

	PAGE
Table 1.1: Summary of Problem Statement	3
Table 1.2: Summary of Research Question	3
Table 1.3: Summary of Research Objective	4
Table 1.4: Summary of project contributions	5
Table 5.3: Details of pin number and slot	77
Table 5.4: Status of Implementation	80
Table 6.1.3: Table of Test Schedule.....	83
Table 6.2.1: Table of classes of test.....	84
Table 6.3.1.1: Table of testing communication Raspberry Pi and Raspberry Pi camera	85
Table 6.3.1.2: Table of testing communication OpenCV coding and Raspberry Pi camera	86
Table 6.3.1.3: Table of testing accuracy of HOG algorithm in face recognition	86
Table 6.3.1.4: Table of testing email.....	87
Table 6.4.1.3: Result of Output	89
Table 6.4.2.2: Result of Output	89
Table 6.4.3.1: Comparison output of algorithm accuracy on valid user “Tony”	93
Table 6.4.3.2: Comparison output of algorithm accuracy on valid user “Elon”	94
Table 6.4.3.3: Comparison output of algorithm accuracy on valid user “Bella”	95
Table 6.4.3.4: Comparison output of algorithm accuracy on valid user “Arasu”	96
.....	96
Table 6.4.3.5: Result of Output	96
Table 6.4.4: Result of Output	98

LIST OF FIGURES

	PAGE
Figure 2.1: Framework	9
Figure 2.2.3: Top ten burglary incidence and corresponding detriment value (L.S.Chiew, (2019), Analysis Of Burglary Crime)	13
Figure 2.3.2.1: Eigenfaces Architecture (Debaraj, 2017)	22
Figure 2.3.2.2: Fisherfaces Architecture (Delpiah, 2019)	23
Figure 2.3.2.3: Local Binary Patterns (LBP) Architecture (Marwan, 2017)	24
Figure 2.3.2.4: Scale-Invariant Feature Transform (SIFT) Architecture (L Yao, 2018)	25
Figure 2.3.2.5: Convolutional Neural Network (CNN) Architecture (N Ma, 2018)	27
Figure 2.3.2.6: OpenFace Architecture (Stephanie, 2018)	28
Figure 2.3.2.7: Histogram of Oriented Gradients Architecture (Kosuke Mizuno, 2018)	29
Figure 2.4.5.1: Architecture Smart Homes and Cities (Nashwan Adnan Othman, 2018)	33
Figure 2.4.5.2: Architecture Smart Security System (Danish Ali Chowdhry, 2019)	34
Figure 2.4.5.3: Architecture Web and Mobile based Facial Recognition (Maria, 2017)	35
Figure 2.5.3: Proposed solution for smart CCTV home camera.	39
Figure 3.1.2: Flow of Waterfall Model Phase.	44
Figure 4.2.2.1: Raspberry Pi Operating System.	54
Figure 4.2.3.1: Raspberry Pi 4.	55
Figure 4.2.3.2: Raspberry Pi High Quality Camera.	56
Figure 4.2.3.3: Raspberry Pi 6mm Lens Camera	57
Figure 4.2.3.4: MicroSD (32GB)	58

Figure 4.3.1: Circuit Design of Raspberry Pi Camera	58
Figure 4.3.2: Architecture of System.....	59
Figure 4.3.3: Flow Chart of system	60
Figure 4.3.3.1: Flow Chart of OpenCV module	61
Figure 4.3.3.2: Flow Chart of HOG algorithm module	62
Figure 4.3.3.3 Flow Chart of HOG algorithm module.....	64
Figure 5.1: Flowchart of system.....	67
Figure 5.2.1: Deployment of the face recognition.....	69
Figure 5.2.2: OpenCV coding for detecting people face from the video frame..	70
Figure 5.2.2.1: OpenCV coding for showing known person name with frame on face	71
Figure 5.2.3: HOG algorithm coding to get known images to train for recognition.	71
Figure 5.2.3.0: Original HOG algorithm face encoding	72
Figure 5.2.3.1: Coding of HOG algorithm that modified	73
Figure 5.2.3.2: HOG coding for compare images and show accuracy of recognition	74
Figure 5.2.3.3: HOG coding for getting face parameters in image frame	75
Figure 5.2.4: Gmail notification coding when unknown person detected.....	76
Figure 5.3: Raspberry Pi 4 board	77
Figure 5.3.1: Prototype of Smart CCTV camera with face recognition	78
Figure 5.3.2: Raspberry Pi Camera and Board Connection coding.....	78
Figure 5.3.2.1: Command to check Raspberry Pi camera connection.	79
Figure 5.3.2.2: Raspberry Pi Camera successfully connected and capture image.	79
Figure 6.4.1: Command to test TC_1.	88
Figure 6.4.1.1: Path of testing images saved.	89
Figure 6.4.1.2: Success output from the communication.....	89
Figure 6.4.2: Test TC_2 by running the coding in Raspberry Pi	91
Figure 6.4.2.1: Output showing face successfully detected.....	91
Figure 6.4.3: Test TC_3 by running the HOG command in Raspberry Pi board	92
Figure 6.4.3.1: Test TC_3 by running the SIFT command in Raspberry Pi board	92

Figure 6.4.4: Invalid user faces that used to test notifications 97
Figure 6.4.4.1: User receive email notification named “Alert Intruder”..... 98
Figure 6.4.4.2: Raspberry Pi successfully send notification with the image of intruder to user..... 98



LIST OF ABBREVIATIONS

FYP	-	Final Year Project
HOG	-	Histogram of Oriented Gradients
CNN	-	Convolutional Neural Network
SIFT	-	Scale-Invariant Feature Transform
LBP	-	Local Binary Patter
CCTV	-	Closed-circuit television
SMS	-	Short Message Service

LIST OF ATTACHMENTS

		PAGE
Appendix A	Coding of the system	103
Appendix B	Manual of system	104



CHAPTER 1: INTRODUCTION

1.0 Introduction

Smart CCTV are becoming increasingly popular with the advances in both machine vision and semiconductor technology. Usually, CCTV was only able to capture images, while with the smart camera concept, a camera will have ability to generate specific information from the images that it has captured. We will address the context of the project which is applying face recognition in smart CCTV home camera. The research problem will be formulated into three research questions. Based on the research questions, the current problem that faced by users for the smart CCTV home camera will be clearly shown. After that, the research objective can be generated, and all works done in this project will be based on the research objective.

Firstly, project background will be discussed for the purpose of doing this project. The background of smart CCTV will be described. Secondly, the research problem will be summarized. After that the research question and objective will be conducted. Project scope and contribution will also be discussed. Lastly, the report organization will be described to make sure the project is carried out in correct flow.

1.1 Project Background

A smart camera, also known as an intelligent camera is a machine vision system that in addition to image capture circuitry that can collect specific informations. Usually smart CCTV camera implemented at the indoor or outdoor of the house to make sure the house is secure. Video monitoring for watching a specific area has

become a necessity in a world where everyone needs to keep their valuables safe and protected. To fix this problem, people use a smart surveillance device for location such as bank vaults and homes where human presence is not available. They continuously monitor the area with cameras. (Jain, A. Basantwani, S. Kazi, O. & Bang, Y, 2017).

People could not continuously monitor the camera for 24 hours because they have other real life things to do and they also get tired or might wasting their time on it. To avoid this thing, people came out with new solution called intruder detection with sensors. Intruder detection is in the protected zone that can be performed by physical protection that acts inside the objects perimeter. Intruder detections also can be done using active protection elements. One of the active protection elements is alarm systems and sensors. (Vel'as, A., Kutaj, M., & Ďurovec, M, 2017). They added motion sensor that connected to the smart camera, which will avoid people to monitor 24 hours.

The smart camera was redesign and implement at home security system with the capability for detect movements. The smart camera was implemented with PIR sensor that will detect the movement around the sensor that will activate the smart camera for capture pictures or record the surrounding situation. When the sensor detect suspicious object movement, the alarm also will go on to scare the intruder. (Nico Surantha, W. R, 2018). People also do not need to monitor the camera for 24 hours but the system still lack of few things in term of notify the owner of house when movement is detected at their house. The sensor also detect any movement include animals movement like cat and dog which will activate the smart camera and also the alarm.

The system was not efficient enough to secure the home. After that, the system have been embedded with notifier to notify user by using SMS-based. Once the sensor detect movement, it will sent SMS to the house owner. So the house owner can view the smart camera to make sure the surroundings are safe. (Nwalozie G. C1, A. A, 2015). Eventhought, the system have notifier but it keep notify any movement include animal movement. Therefore, the smart camera need to have face recognition which will be more efficient to detect people compare to the motion detector.

There is the camera with face recognition that have been created by the various group. They used different type of algorithm technique for the face recognition but there are still a few problems facing by face recognition in term of algorithm technique. Some people are harder to recognize by face recognition. However, there should be use the right face recognition algorithm to make a subject easier to recognize.

(J.R.Beveridge, 2018). Therefore, the smart camera need to have suitable face recognition algorithm technique thats more efficient to detect the people face.

1.2 Problem Statement

The residents require home security that needs to maintain the security of their house to avoid any intruders or thieves. Detecting intruders at the in front of the house door must be done by detecting their faces and notify the user, so the user will be alert on surrounding activities at the house. As such, the absence of such a system is a problem for residents. The Research Problem (RP) is summarized in Table 1.1.

Table 1.1 Summary of Problem Statement

No	Research Problem
RP1	The current algorithm technique that used in the face recognition could not recognize people properly if they are stranger or thieves at in front of the house door.

Thus, Research Questions (RQ) which are depicted in Table 1.2 is constructed to identify the research problem as discussed in the previous section.

Table 1.2 Summary of Research Question

No	Research Question
RQ1	What are the features for face detection that required for smart CCTV Home Camera?
RQ2	How are we going to detect and recognize people face using smart CCTV Home Camera?
RQ3	How to validate the result of the face recognition from the smart CCTV Home Camera?

RQ1: What are the features for face detection that required for smart CCTV Home Camera?

This research question is used to study about the face detection features for smart CCTV Home Camera.

RQ2: How are we going to detect and recognize people face using smart CCTV Home Camera?

This research question is formulated by considering the method in how does the CCTV can detect and recognize people face. Thus, it is important to know how to apply the system at in front of the house door.

RQ3: How to validate the result of the face recognition from the smart CCTV Home Camera?

This research question is to know how the result of the smart CCTV Home Camera with face recognition.

1.3 Objective

Table 1.3: Summary of Research Objective

RP	RQ	RO	Research Objective
RP1	RQ1	RO1	To identify features for face detection in smart CCTV Home Camera.
	RQ2	RO2	To develop face recognition in smart CCTV Home Camera.
	RQ3	RQ3	To test and verify face recognition in smart CCTV Home Camera.

RO1: To identify features for face detection in smart CCTV Home Camera.

Firstly, we need to investigate what is the features that required and how it can detect and recognize people face at house area.

RO2: To develop face recognition in smart CCTV Home Camera.

We discuss about the smart CCTV system that can recognize people that in front of the house door.

RO3: To test and verify face recognition in smart CCTV Home Camera.

After discussing the applied method for the system, we need to provide an alert notification to the user when detect unknown people.

1.4 Scope

Scope of project is going to be conducted as follow:

The sample dataset from the system will be used in this project. A dataset is obtained from the smart home CCTV camera. The camera will capture the image of the people and send it to the system. The system will compare the image with the image in the system and will send it to the user. The user will obtain the image through email when the image is different from the image in the system.

This project just focuses the methods used by the system to capture the people image in front of the house door and compare with the images that in the system data and send notifications to the user when unknown people image is detected. This project used smart CCTV home camera with face recognition to get images of the people.

1.5 Project Contribution

The contribution of this project is summarized in table 1.4.

Table 1.4 Summary of project contributions

RP	RQ	RO	RC	Research Contribution
RP1	RQ1	RO1	RC1	The range of the camera that can recognize peoples face in front of house door.
	RQ2	RO2	RC2	The methods to capture people image using face recognition.
	RQ3	RO3	RC3	The accurate of face recognition system with alert notification to the user.

Table 1.4 shows the project contribution based on the research problem, research question and research objective.

1.6 Report Organization

Chapter 1: Introduction

This chapter will be discussed about the introduction, project background, problem statement, research question, research objective, scope, project contribution and report organization.

Chapter 2: Literature Review

This chapter will study about the related works such as the features for the face recognition, the importance of face recognition at in front of house door and type and parameter of face recognition that will used in this project. The related works will be contributed to the next chapter.

Chapter 3: Methodology

This chapter will explain the method used to analyze face recognition and project methodology. Project schedule and milestone will discuss in this chapter to finish the project in time.

Chapter 4: Analysis and Design

This chapter will introduce the problem analysis, requirement of software and hardware. The experimental design such as logical design and physical design will also conduct in this chapter. Environment will setup based on the requirements.

Chapter 5: Implementation and Testing

This chapter will describe the method to capture image and do comparison in order to notify user when stranger image captured. The test result and analysis will conclude in this chapter.

Chapter 6: Conclusion

This chapter will summarize all chapters as a conclusion. Project summarization, limitation and future work will discuss.

1.7 Summary

As a conclusion, this project is aimed to create a system that can recognize people face by using smart CCTV home camera to make sure user house more secure. The problem statement, research objectives and scope within this chapter is for identifying and discuss the problem statement, as well as creating a possible solution. A report organization is also made in this chapter to show the sequence of the project.



CHAPTER 2: LITERATURE REVIEW

2.0 Introduction

The chapter will study about the related works such as the home environment to implement and monitor using CCTV and the face recognition algorithm technique that will be used in this project.

2.1 Framework

The figure 2.1 shows the framework for this chapter. This framework is used to follow what will be done in this chapter. Other than that, this framework also very useful in this chapter to know what is contained in the title.

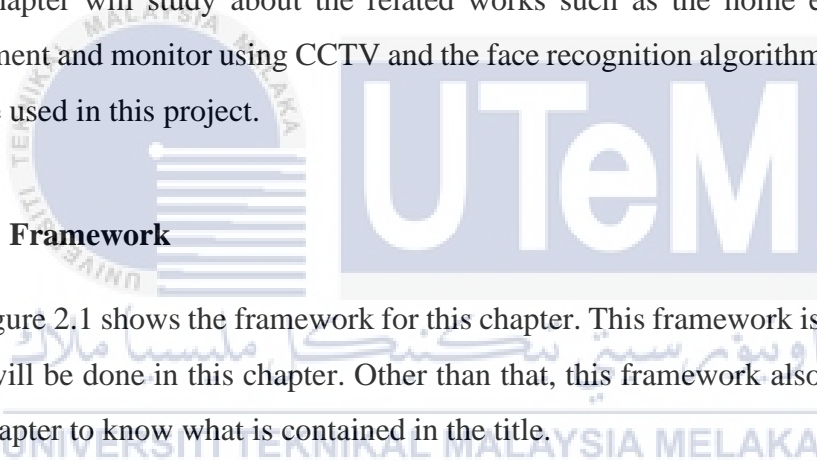
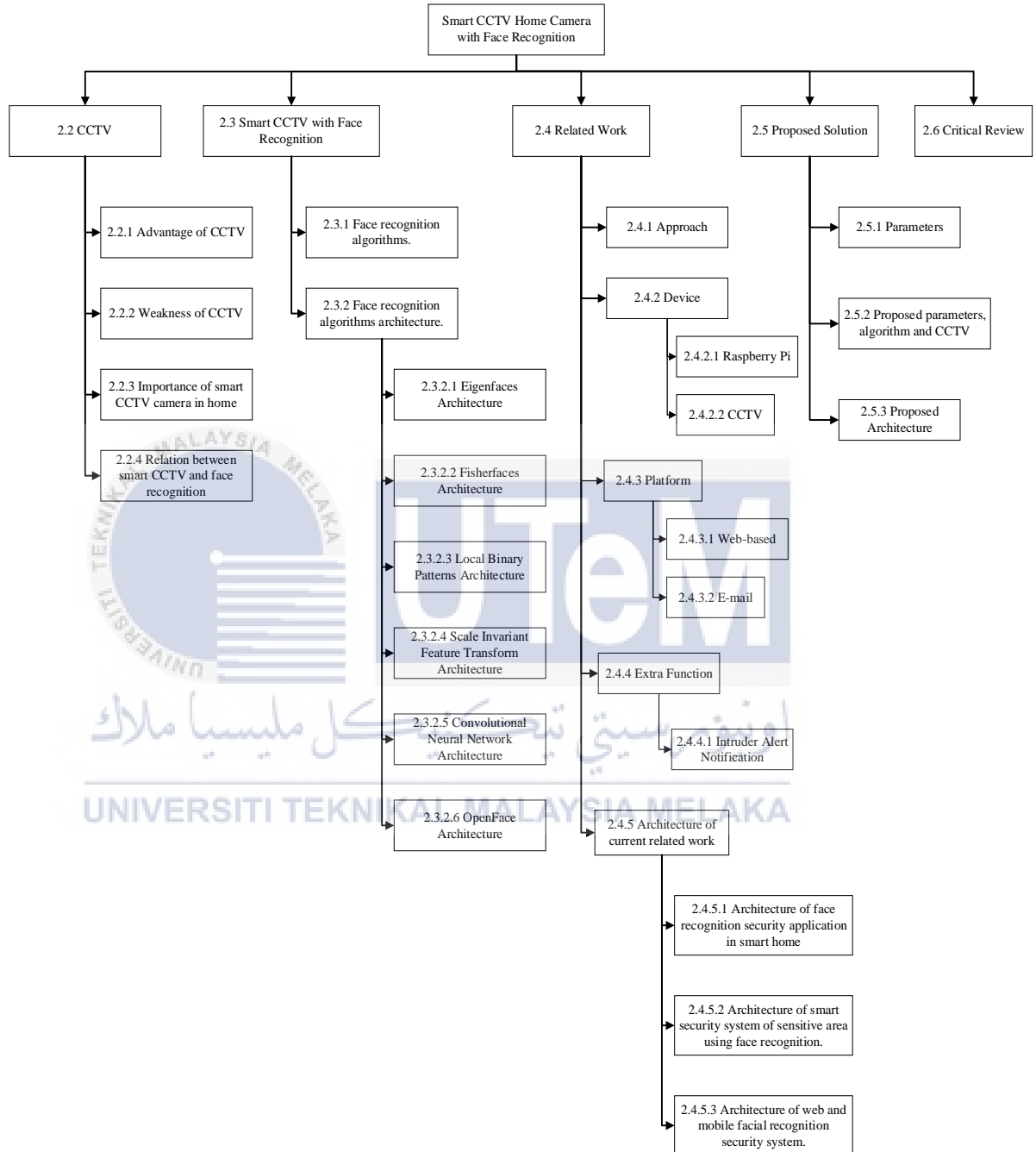


Figure 2.1 Framework



2.2 CCTV

Closed-circuit television, or video monitoring, is the abbreviation for CCTV. In contrast to “regular” television, which is broadcast to the general audience, “closed-circuit” television is broadcast to a small (closed) number of monitors. CCTV networks are widely used to track and prevent illegal activity and to monitor traffic violations, although they may also be used for other purposes. German scientists invented CCTV technology in 1942 to track the launch of V2 rockets. It was also used by American scientists as part of the atomic bomb trials. (Hemming, M, (2006)).

There are many types of CCTV systems. The CCTV systems are divided into 3 types which is analog, digital and network or IP. Analog and digital services operate in somewhat different ways, but current CCTV networks transform analogue to digital using conversion tools and hardware. The analog CCTV need to relay continuous video signals, use Bayonet Neill-Concelman (BNC) connectors on coaxial cables. They have a low resolution but are inexpensive and reliable. In an analogue system, there are more peripherals, for example, normal coaxial cables do not normally relay audio. Analog signals can be converted to digital, making digital conversion more cost-effective and for older equipment. (Gill, M, (2006))

The digital camera, digitalize signals. These digital cameras do not need a video capture card because videos are saved directly to a monitor, but they do necessitate a relatively significant amount of storage space for recordings, because they are usually highly compressed. The network or IP CCTV use a video server to stream videos over the internet which can be used for analogue or digital cameras. The benefits of network and IP CCTV include Wi-Fi and audio capabilities, Distributed Artificial Intelligence (DAI) for image analysis, remote connectivity, Power Over Ethernet (POE), and higher resolution. All three systems are still in use, with IP camera systems and digital video cameras being the most common.

2.2.1 Advantage of CCTV

The CCTV have several advantages for the users. One of the advantages is its help to deter crime. This is the most important and apparent advantage of adding CCTV. We will be able to see their impact on users almost instantly after they have been installed and if they are hidden, we can begin to experience a sense of confidence, which is invaluable. We will deter crime from happening if we mount cameras in our home or at work. Mischief-makers are intimidated enough by the sight of the camera looking back at them, as well as the prospect of being spotted red-handed, to be on their best behavior, knowing that their name and criminal acts have been captured. If we are having issues with burglary, punctuality, or efficiency, the CCTV will help us solve the problem. It gives us leverage and protects our home and workplace from being easy targets for criminals. (JH Hong, (2020))

CCTV also helps to monitor scenarios and activities of the surroundings. CCTV systems are incredibly simple to use, and they can be mounted almost anywhere as long as there is a power source nearby. They come in a variety of sizes and shapes, some of which are small enough to be covered in trees, photos, and photo frames. We can purchase hidden CCTV or mountable CCTV, depending on our needs. CCTV useful because they allow us to keep track of who is visiting our home and workplace, as well as what is going on there. This is an excellent method for detecting unknown individuals and keeping track of their behavior. (B Min, (2020))

CCTV also help to gather evidence for crime. When we need to track people's behavior and speech, even after an incident, having CCTV mounted in strategic locations comes in handy. Not only do modern CCTV have high-quality recording capability, but they also have audio capabilities. They are more effective than ever at filming a sequence of events to the precise pictures and smooth music. CCTV useful in legal situations where the eyewitness may have forgotten a key fact or may not be giving an accurate description of what happened. The judicial authority will see the sequence of events as they happened with the help of a CCTV. CCTV footage will assist us in making accurate and reasonable choices while resolving conflicts, particularly in domestic and professional situations. (A Nassauer, (2018))

2.2.2 Weakness of CCTV

Even though there are advantages in using CCTV but there is also weakness of having CCTV. One of the weakness is CCTV could not stop theft. Users may use CCTV to film video for further viewing, as well as to aid in the capture of suspects and the administration of justice. They cannot, though, interrupt a crime in the middle of it. Unlike an alarm system, they do not notify neighbors or the police.

Other than that, the CCTV can be vulnerable. When we, as video camera owners, want to keep up with the new surveillance technologies, we must keep in mind that intruders and criminals are doing the same. A cunning trespasser would be well aware of them and may have devised a method of evading detection. Furthermore, tech-savvy suspects may have figured out how to uninstall or isolate the devices from their power source. Whether they recognize the CCTV as fakes or dummies, they would be almost ineffective in crime prevention. In the worst-case scenario, hackers will use the Internet to wreak havoc on our CCTVs device and use them to spy on us instead. As a result, CCTV are susceptible to disruption or violence. (J Kim, (2019))

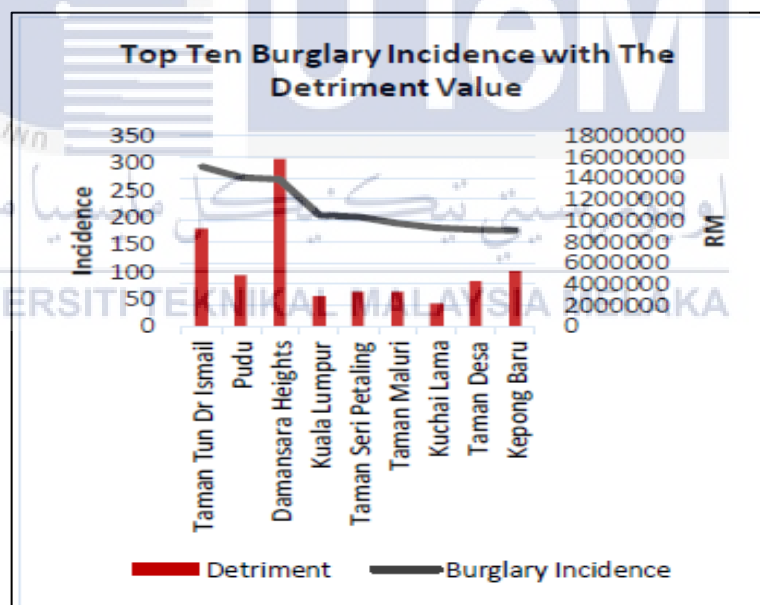
The CCTV can be a costly affair. Though dummy CCTV are inexpensive, actual CCTV can cost hundreds of dollars or even thousands of dollars based on the functionality and amount of CCTV and surveillance systems purchased. Installing them and keeping them up to date come at a price. If we are considering installing them ourselves, don't do so unless we are well-versed in electrical systems otherwise, we risk destroying it. (C Hakim, (2018))

Finally, the current CCTV which implemented with face recognition have weakness in identifying valid user. The CCTV having trouble in recognize valid user due to no proper face recognition algorithm and CCTV lens that cannot capture people face more clearly. Sometime the CCTV have trouble in term of lighting to identify valid people face and could not recognize them well. Some CCTV images are blurry and fuzzy which will be hard to recognize people face. As a result, the CCTV will consider them as theft and started to capture their image.

2.2.3 Importance of smart CCTV camera in home.

With the increasing numbers of crime that have been committed yearly, the government has made several attempts to prevent crimes. According to the Malaysia Crime and Safety Report, Malaysia has been categorized as the most dangerous country in South-Eastern Asia region with a crime index of 65.56 compared to Vietnam (53.45), Cambodia (52.72) and Indonesia (52.16) as stated in (OSAC, 2016). Generally, the burglary is the most regular type of crime compared to the other types of the crimes in Malaysia and the same trend is appear for burglary. The number of losses for burglary far exceeded than other types of the crimes. (L.S.Chiew, (2019), Analysis Of Burglary Crime). Figure below shows the one of the areas where the higher cases of burglary in Malaysia.

Figure 2.2.3 Top ten burglary incidence and corresponding detriment value
(L.S.Chiew, (2019), Analysis Of Burglary Crime)



To avoid this burglary problem getting worst among residents, the smart CCTV camera should be implemented at the resident's house. After the burglary incidence, the CCTV gained lots of popularity as people starting to realize the benefit of it. CCTV is not good enough to give a better security to the house, therefore CCTV system have been created to workable and give optimum responses to highly complex problems.

According to the research shows that the potential future benefits for the CCTV system are amazing in their own scope. This CCTV system gives promise of optimum responses to highly complex problem areas. For example, capture images and record videos and make an early warning or response to the problems. Moreover, smart CCTV home camera with face recognition could provide better control by minimizing distortion and increasing precision in term of security of the resident's house. This system also has its own benefit in term of maintenance of the systems and its better performance of its functions.

Therefore, this smart CCTV and systems are used in this method, making it one of the most promising solutions for long-term performance and increase reliability. The ultimate goals of research in this area are to understand and capture the composition and microstructure of all new materials which is face recognition that is critical to the manufacture of successful smart CCTV. The knowledge gained by studying the behaviour of CCTV system with the face recognition that can help to accelerate the security of the resident's house.

2.2.4 Relation between Smart CCTV and face recognition

CCTV and face recognition have a lot of relation which made into face recognition CCTV systems. Facial Recognition CCTV is one of the most recent advancements in the world of video monitoring and surveillance systems, is the ideal technology for businesses that need to adopt complete and proactive security policies. Face recognition requires CCTV because its need images to recognize people and that images can be captured by CCTV. CCTV Surveillance's Facial Recognition CCTV system dynamically analyses photographs of persons from incoming video streams to those stored in a specified access control list, sending warnings when a positive match is found. In 1960, the first semi-automated system for facial recognition to locate the features such as eyes, ears, nose and mouth on the photographs. In 1970, used 21 specific subjective markers such as hair colour and lip thickness to automate the recognition. (Goldstein, (1970)). In 1988, used standard linear algebra technique to the face recognition. (Kirby, (1988)). The face

recognition CCTV yields a good level of performance despite occlusions of the face, use of glasses, scarves, changes of facial expression and rotations of the face. It also prohibits the use of photos to impersonate users. With this user-friendly system, the CCTV Surveillance Face Recognition CCTV solution gives improved management choices for all surveillance and security databases, enabling improved productivity through faster searches and identifications, and will pay for itself by lowering losses and improving bottom line profits.

2.3 Smart CCTV with face recognition.

In this section will discuss about face recognition algorithm technique for smart CCTV system. This section only focuses on algorithm technique. In this project, we are focusing on the best method of algorithm for the smart CCTV that is more efficient and suitable to use for capture and recognize people face that is in front of the house.

2.3.1 Face recognition algorithms.

As we all know, there are different types of algorithms for the face recognition. There is currently no procedure that has shown acceptable outcomes in all situations. A comparative study of various techniques based on results obtained from different type of groups. There is different type of parameters that used by the face recognition algorithms to detect people face. These parameters make the algorithm more precise in detecting people face. So, there is a multiple algorithms or technique for face recognition that implemented in CCTV.

One of the algorithms is Eigenfaces for recognition. Eigenfaces is an appearance-based approach to face recognition that aims to capture variance in a series of face images and use this detail to encode and compare individual face images in a holistic manner. Eigenface extract specific facial data, which may or may not be linked to human intuition about face features like the eyes, nose and lips. Capturing the statistical difference between facial images is one way to do so. (Dr. Sheng Zhang,

2011). Eigenfaces represent facial pictures. Each face image can be interpreted with a limited number of parameters to minimize computation and space complexity. (Dr. Matthew Turk, 2011).

Other than that, fisherfaces are one of the algorithms for face recognition. Finding a subspace that contains the majority of the data variation is one way to represent the input data. This can be obtained with the use of Principal Components Analysis (PCA). PCA produces a series of eigenfaces when added to face images. These eigenfaces are the eigenvectors associated with the training data's highest eigenvalues in the covariance matrix. The eigenvectors discovered thus refer to the LS solution where it is a very effective way to view data, so it keeps the data variation while removing any unwanted current similarities. (Aleix Martinez, 2011).

Furthermore, local binary patterns (LBP) is also one of the face recognition algorithms. In computer vision, this is a kind of visual descriptor that is used for classification. It has since been discovered to be a useful function for texture classification and it has also been discovered that combining LBP with the Histogram of Directed Gradients (HOG) descriptor significantly enhances detection efficiency on certain datasets. (Xiaoyu Wang, 2009). LBP divide the windows into cells that examined. LBP concatenate histograms for all cells. This results in a function vector for the whole window.

After that, there is an algorithm that called scale-invariant feature transform (SIFT) is a computer vision feature recognition algorithm that detects and describes local features in images. Item recognition, robotic mapping and navigation, image stitching, 3D modeling, motion recognition, camera tracking, human wildlife identification and match moving are some of the applications in the SIFT. Object SIFT key points are first retrieved and stored in a database from a collection of reference images. An object in a new image is identified by comparing each element in the new image against the database. (T.Lindeberg, 2014)

Another algorithm facial recognition is to use a convolutional neural network (CNN) (CS231n) (Deshpande, 2016c) (Deshpande, 2016a) (Deshpande, 2016b). CNN has an architecture that allows it to work with two-dimensional images as inputs. A

CNN is made up of multiple layers known as secret layers. The layers are made up of a number of neurons. A neuron receives an input and has a definite weight. It will have an output after applying its logic to the input. The first layer's input is an image of a face. The expected class, which is the individual, is the performance of the last layer. It is preferable to use a simple example to get a greater understanding of the identification mechanism. As a result, instead of people, the letters "X" and "O" are used as grades. (Nicolas, 2017)

Furthermore, Histogram of Oriented Gradients (HOG) is also another type of facial recognition algorithm. The histogram of oriented gradients (HOG) is a feature descriptor for face recognition in computer vision and image processing. The method counts the number of times a gradient orientation appears in a certain area of a picture. Edge orientation histograms, scale-invariant feature transform descriptors, and shape contexts are all comparable methods, but this one differs in that it is computed on a dense grid of evenly spaced cells and employs overlapping local contrast normalization for increased accuracy. The whole image is divided into smaller sections, with gradients and orientation computed for each one. The gradients and orientations of the pixel values are used to construct the histograms. (Kosuke Mizuno, 2018)

Lastly, Face recognition library OpenFace (Amos, Ludwiczuk, & Satyanarayanan, 2016). It is based on the FaceNet (Schroff & al.) systems developed by Google. For facial recognition, OpenFace employs a deep convolutional neural network. It makes use of a tweaked version of FaceNet's nn4 network. 500k photos were used to train OpenFace. The features are extracted using a deep neural network. The faces are recognized and extracted from the images using dlib's HOG-based pre-trained detector. After that, the faces are preprocessed before being employed in the convolutional neural network.

Table 2.3.1 Algorithms.

Face recognition algorithms	Parameter	Nodes	Advantage	Disadvantage
Eigenfaces	Eyes, Nose, mouth, eyebrow	271 x 271	<p>(1) Without any major low-level or mid-level processing, raw intensity data is used directly for learning and recognition.</p> <p>(2) No knowledge of face geometry or reflectance is necessary.</p> <p>(3) data compression is done via a low-dimensional subspace representation.</p> <p>(4) recognition is easy and efficient when compared to other matching systems.</p>	<p>(1) Learning takes a long time, making it difficult to keep the face database up to date.</p> <p>(2) When more face classes are represented by the same face space, the likelihood of class overlapping increases.</p>
Fisherfaces	Eyes, mouth, Nose	105x105	<p>(1) Immune to noise-induced images and blurring effect on the image.</p> <p>(2) Better classification of different classes image.</p>	<p>(1) More difficult to discover the projection of face space than Eigenface</p> <p>(2) The ratio of between-class scatter to within-class dispersion</p>

			(3) Can classify the training set to deal with different people and different facial expression	takes a long time to calculate. (3) Require larger storage of the face and more processing time in recognition
local binary patterns (LBP)	Eyes, mouth, Nose	8x8	(1) High discriminative power (2) Computational simplicity (3) Invariance to grayscale changes and (4) Good performance.	(1) It is not rotation invariant. (2) The size of the features grows exponentially with the number of neighbors, resulting in increased computational complexity in terms of time and space. (3) The structural information it captures is restricted. The magnitude information is discarded and just the pixel difference is used.
scale-invariant feature transform (SIFT)	Eyes, mouth, Nose	500x500	(1) Can produce a high number of features that cover the picture densely throughout a wide variety of sizes and places.	(1) With lighting changes and blur, it doesn't function properly.

			<p>(2) Performance of SIFT is close to real-time performance.</p> <p>(3) Features are not affected by occlusion or clutter.</p>	<p>(2) Not effective for low powered devices.</p> <p>(3) Quite slow in processing recognition.</p>
Convolutional Neural Network (CNN)	Eyes, mouth, Nose	28 x 28	<p>(1) Reduce calculation time as compared to a traditional neural network.</p> <p>(2) Convolution greatly reduces computing without losing the data's substance.</p> <p>(3) Great at image categorization.</p> <p>(4) Use the same knowledge to all image places.</p>	<p>(1) It substantially slower because of operations like maxpool.</p> <p>(2) The training process will take a long time if the machine does not have a powerful GPU.</p> <p>(3) The ConvNet requires a huge dataset to train and process.</p>
Histogram of Oriented Gradients (HOG)	Eyes, Nose, Mouth	8 x 8	<p>(1) The coarseness of the binning employed by HOG will not lose information's.</p> <p>(2) Great at capturing edges and corner in images.</p>	<p>(1) HOG only uses 1 direction for each pixel.</p> <p>(2) Do not have scale and rotation invariant.</p> <p>(3) Require more complex reasoning</p>

			<p>(3) Able to extract global features.</p> <p>(4) Does not require GPU, can run in CPU.</p> <p>(5) Able to handle occlusions and overlaps.</p>	
OpenFace	Eyes, mouth, Nose	128x128	<p>(1) It is Open Source.</p> <p>(2) Require few human resources.</p> <p>(3) Produce remarkable results on the Labeled Faces in the Wild (LFW) benchmark.</p>	<p>(1) Cannot perform tasks on non-stable images.</p> <p>(2) The training process take a long time.</p>

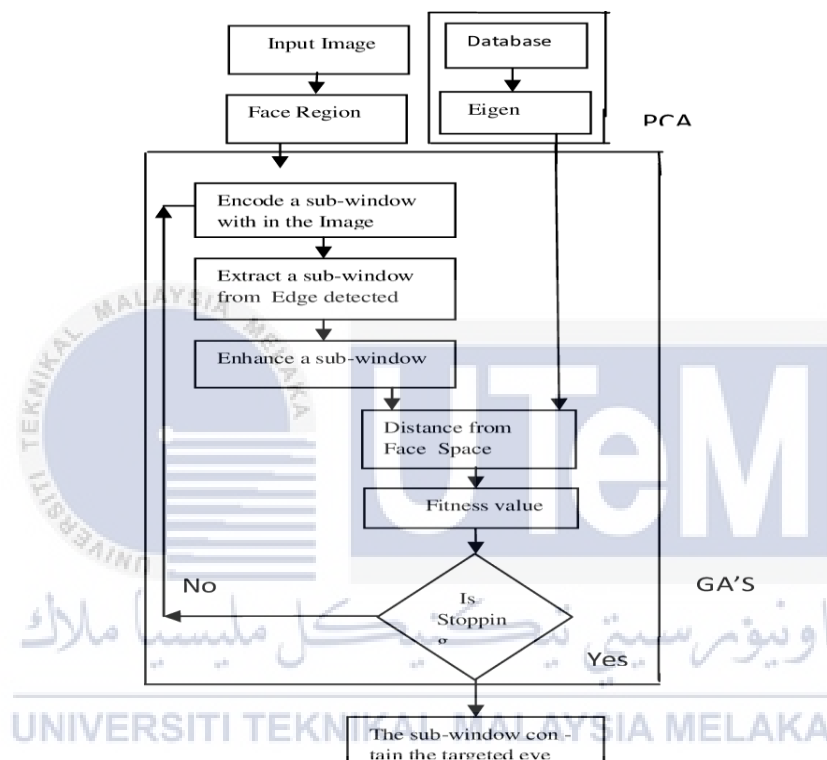
2.3.2 Face recognition algorithms architecture.

2.3.2.1 Eigenfaces Architecture.

Eigenfaces are a great foundation for a face recognition system since they have a high recognition accuracy and are relatively insensitive to illumination fluctuations. Figure 2.3.2.1 shows that the architecture of eigenfaces. The architecture divided into four sub-stages. On the first stage, the eigenfaces use PCA to create an eigen space using database of eyes and skin texture that have been stored. The second stage, using the YCbCr color scheme, extract the skin region from the input image to create a skin probability picture. In the third stage, the algorithm creates an initial population by encoding sub-windows in various location of the image. In the fourth stage, algorithm use the GA and fitness operators, locate the sub-window containing the eye of interest. To summarize the process, an eigen space is generated which serves as a starting point

for the rest of the calculations. The skin area is then retrieved from the input picture. The picture of the skin retrieved is used as the input image for the Genetic algorithm. Different sub-windows were then extracted using GA operators, and the fitness value of all sub-windows was computed. The outcomes, which will be obtained after some iteration, will be based on the best fitness values windows.

Figure 2.3.2.1 Eigenfaces Architecture (Debaraj, 2017)

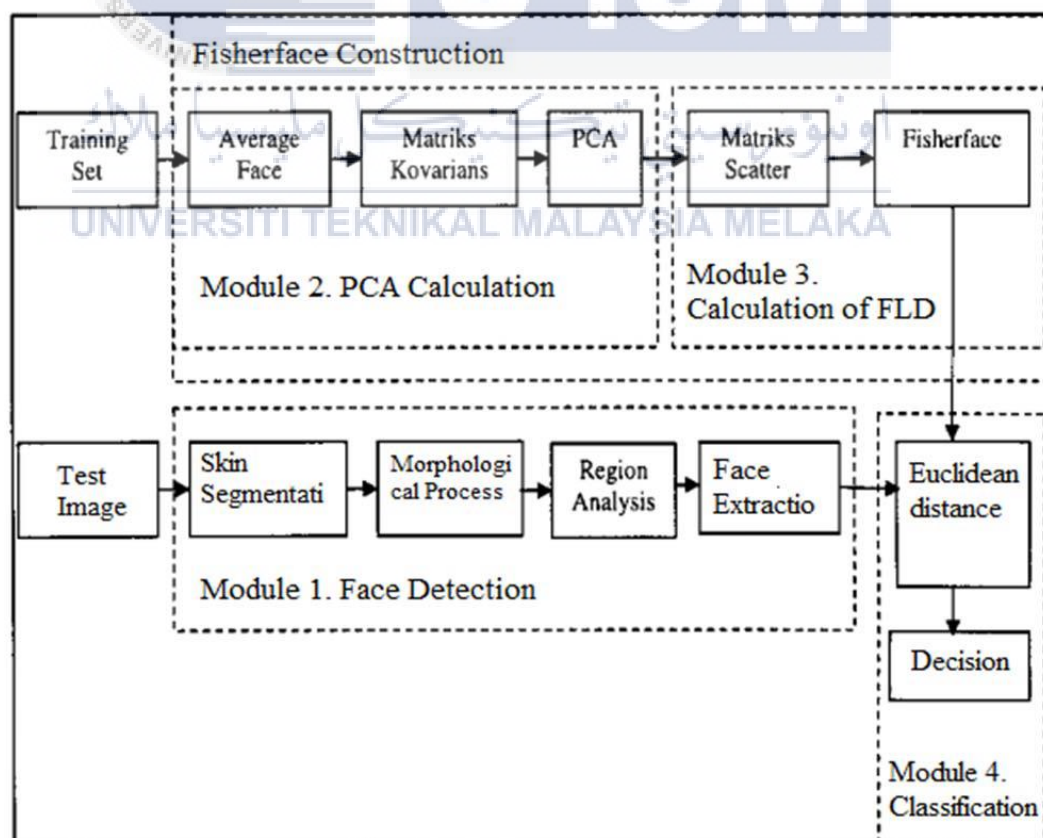


2.3.2.2 Fisherfaces Architecture.

Fisherface is a common face recognition algorithm that is commonly regarded as better to other approaches such as eigenface due to the effort made during the training phase to optimize the separation between classes. Figure 2.3.2.2 shows the architecture of the Fisherfaces. The architecture of Fisherface divided into four main modules. The first module is face detection. In this module, the image that captured is undergoes skin segmentation to track face in the image. Once the skin segmentation done, the detected face will go through morphological process where the image will be

reduplicated, affixation and compounded. After that, the image goes through region analysis where pixels that corresponded to a part of face are grouped and marked as one region. Then the face part is extracted for find similarities with the image in the database. The second part module PCA calculation. In this stage module, many images are added to train the fisherface algorithm. The image will pass through covariance matrix to measure random variable that get change in the images with same face, but different face expression and data is stores in matrix. The PCA will collect all of the matrix data and transfer to third module. The third module is calculation of FLD. In this module, scatter matrix process will occur where the scatter plot used to visualize bivariate relationship between combination variables and transfer to module four. Module four is classification. In this module the captured image and the trained image from database will be differentiated to identify the percentage of difference and similarities. Once this process is finished, the decision will be made whether the image is similar to database image.

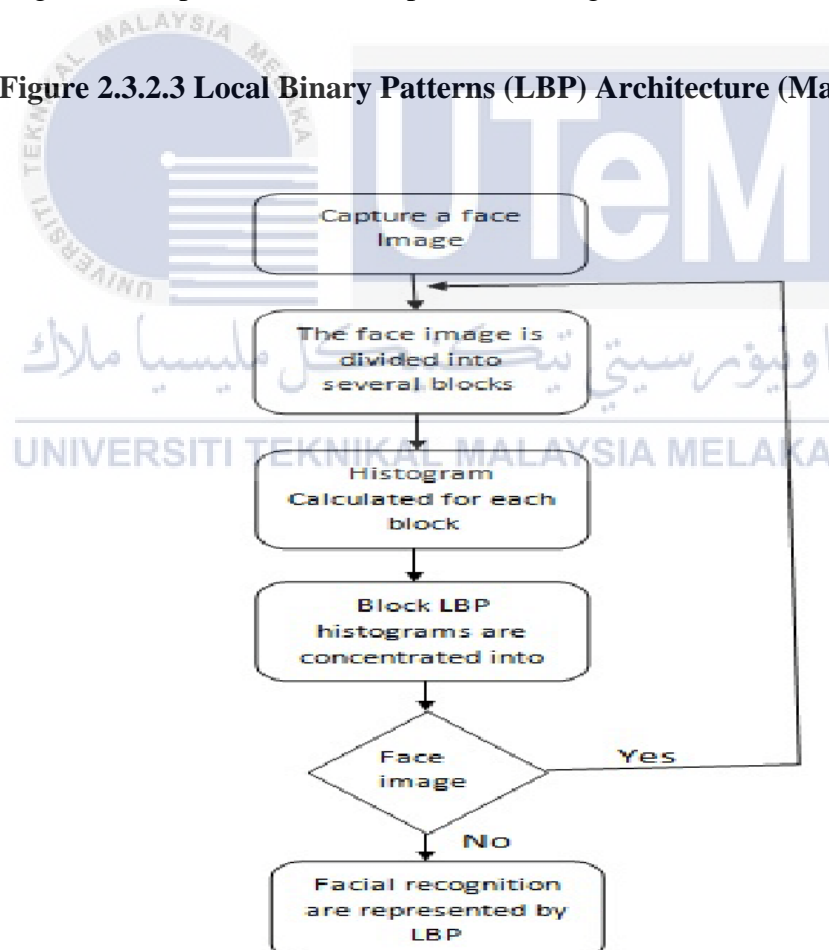
Figure 2.3.2.2 Fisherfaces Architecture (Delpiah, 2019)



2.3.2.3 Local Binary Patterns (LBP) Architecture.

Simple yet effective texture operator that identifies pixels in an image by thresholding each pixel's neighborhood and treating the result as a binary number. Figure 2.3.2.3 shows the local binary pattern (LBP) architecture. In this architecture, the input will be the training image set. The image that captured will be initialize as $temp = 0$, for each image in the training image set as I . The image will be divided into several blocks and calculate each block in histogram. The pattern of histogram will be initialized as $H=0$, before it makes the pattern of histogram. Each center pixel will compute the pattern label of LBP1. It will increase the corresponding bin by 1. The algorithm will find the highest LBP feature for each face and combined into single vector. After that, the algorithm compares with test face image. The output will be feature extracted from face image and compared with center pixel and recognition with unknown face image.

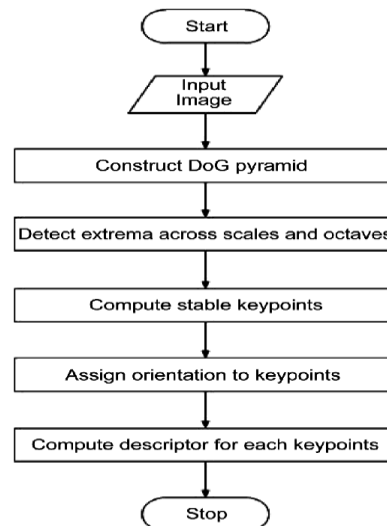
Figure 2.3.2.3 Local Binary Patterns (LBP) Architecture (Marwan, 2017)



2.3.2.4 Scale-Invariant Feature Transform (SIFT) Architecture.

Determine scale by maximizing DoG in scale and in space, the major gradient direction is local orientation. SIFT create histograms of gradient orientation for numerous tiny windows. Figure 2.3.2.4 shoes the SIFT architecture. According to the architecture of SIFT, it has four stages to obtain the features of an image. One of the stages is Scale-space Extreme Identification (SSED) and this stage is divided into two parts which is the building of the Difference of Gaussian (DoG) and the detection of extrema key points. The initial step in the process is to create a pyramid of photos with various blurring degrees and resolutions. To generate a sequence of pictures with the same resolution but changing blurring levels, the initial picture is repeatedly convolved with distinct Gaussian functions defined by their variable standard deviations, known as scaling. The difference in intensity levels of consecutive blurred pictures makes up the pixels of a DoG picture. Pixels recognized as local peaks or minima of DoG pictures across octaves are called key points. This is determined by comparing a DoG pixel to its neighbors at the current and nearby scales. This will occur in compute stable key points stage and assign orientation to key points. The chosen candidate key points (CKs) are screened to exclude any CKs that are unstable. Unstable key points are those with low contrast or those that are part of an image's edge. The selected CKs are fitted with a 3D quadratic function to determine their contrast with nearby points. The feature descriptor is orientation histograms expressed as a vector once a key point orientation has been chosen.

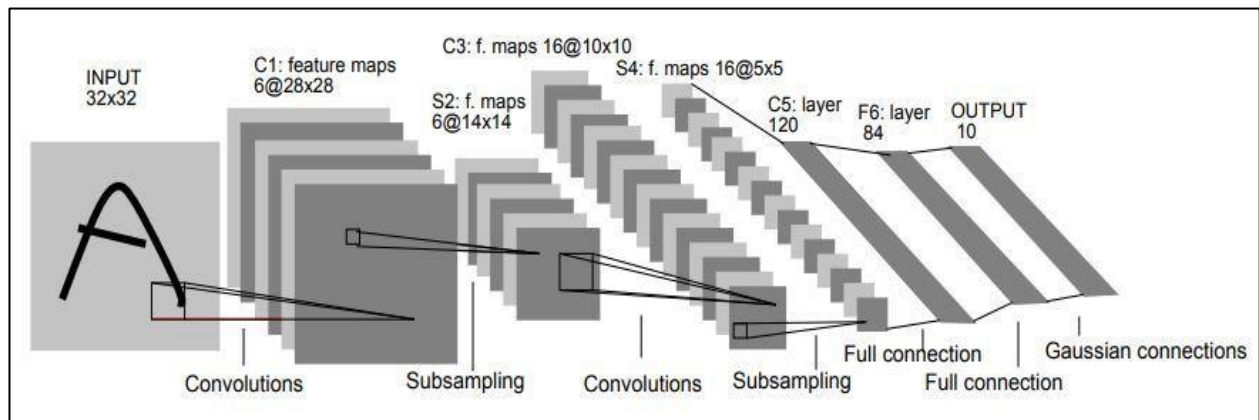
Figure 2.3.2.4 Scale-Invariant Feature Transform (SIFT) Architecture (L Yao, 2018)



2.3.2.5 Convolutional Neural Network (CNN) Architecture.

The Convolutional Neural Network (CNN) using LeNet-5 architecture which is for digit recognition. The figure 2.3.2.5 shows that CNN architecture consists of two sets of average pooling and convolutional. It also followed by a flattening convolutional layer then two full connected layers and finally a softmax classifier. Except for the input layer, the model has seven layers. The first layer is a convolutional layer with a kernel size of 55, a stride of 11, and a total of 6 kernels. As a result, a 32x32x1 input picture yields a 28x28x6 output. Total layer parameters = $5 * 5 * 6 + 6$ (bias terms). The second layer is pooling layer having a size of 22 kernels, a stride of 22 kernels, and a total of 6 kernels. The receptive's input values were mixed together and then multiplied by a trainable parameter (1 per filter), with the result being added to a trainable bias (1 per filter). Layer 3 is a convolutional layer with the same structure as Layer 1, but it includes 16 filters instead of 6. The layer 4 similar to Layer 2, this layer is a pooling layer, but it has 16 filters. The layer 5 is a convolutional layer with a kernel size of 55 and 120 filters. Layer 6 has 84 characteristics and is a thick layer. As a result, the 120-unit input is reduced to 84-unit output. $84 * 120 + 84 = 10164$ total params. The activation function chosen in this case was fairly unusual. Finally, a thick layer of 10 units is employed as the output layer. By using output layer, the face is recognized.

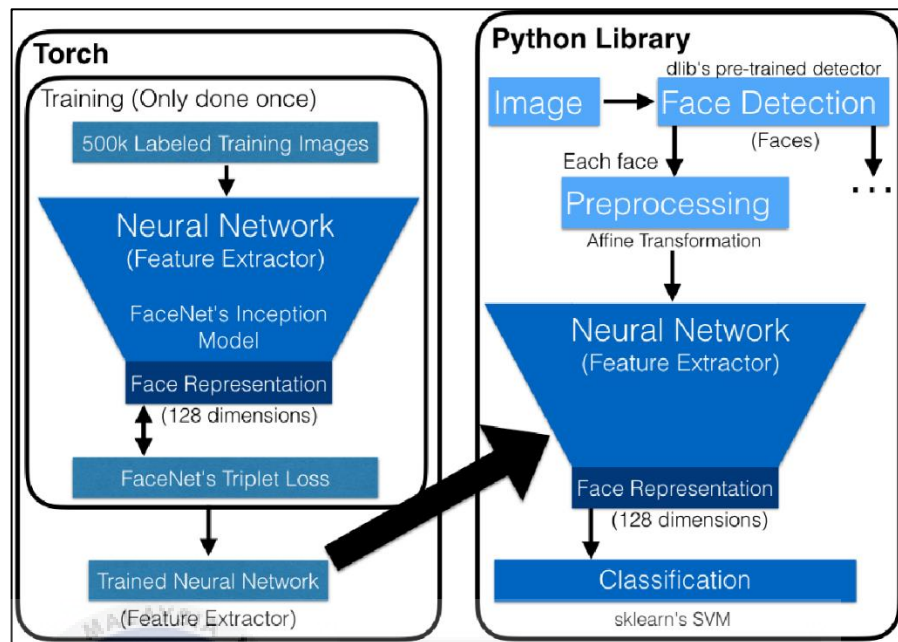
Figure 2.3.2.5 Convolutional Neural Network (CNN) Architecture (N Ma, 2018)



2.3.2.6 OpenFace Architecture.

OpenFace trains offline using Torch, a scientific computing framework, which means it only has to be done once by OpenFace and the user doesn't have to get their hands dirty training hundreds of thousands of images. The images are then fed into Google's FaceNet model, which uses a neural network to extract features. Figure 2.3.2.6 shows the architecture of OpenFace. The major feature extractor tools in OpenFace are neural network architecture adapted from Google's FaceNet. FaceNet is a unified system for face verification, recognition and clustering by find common people among these faces. Using a triplet-based loss function, it trains the outputs to be a compact 128-D embedding. The triplets are made up of two photos from the same source and a non-matching picture from a different source. The goal of the triplet loss function is to separate the positive and negative pairs by a distance margin. Once loss function is separated, the face will be represented to classify the trained face with the preprocessed face.

Figure 2.3.2.6 OpenFace Architecture (Stephanie, 2018)

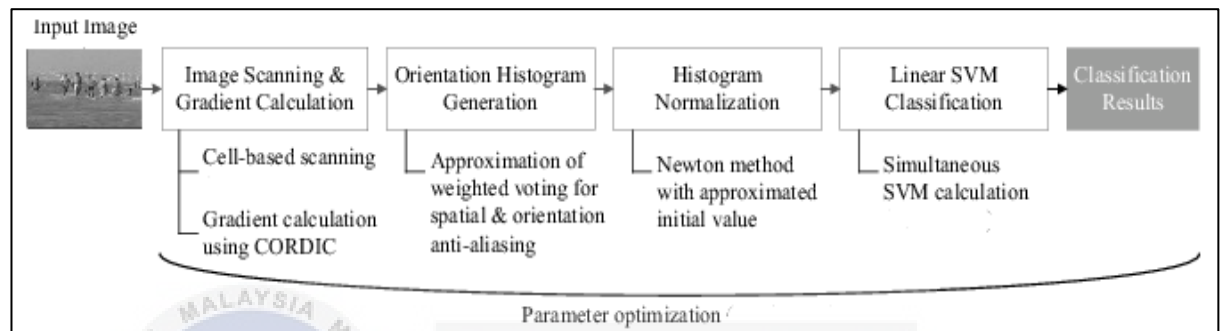


2.3.2.7 Histogram of Oriented Gradients Architecture.

Figure 2.3.2.7 shows the architecture of face recognition using the original Histogram of Oriented Gradients (HOG). First of all, the detection window is used to scan the input image. The scanning of the image done by cell based. The window is split into cells, with each cell's pixels collecting a histogram of gradient orientations. The gradient is calculated by using Coordinate Rotation Digital Computer (CORDIC). After that, the histogram goes through weighted voting approximation for anti-aliasing in space and orientation. Histogram normalization may be used to improve illumination invariance by collecting a measure of local histogram energy over blocks and utilizing the findings to normalize all cells in the block. With an estimated beginning value, Newton's technique is used. Over the detection window, normalized histograms which is HOG features are gathered. For object or non-object classification, the gathered features are input into a linear SVM. The simultaneous

SVM calculation will occur during linear classification stage. Lastly, the input image is recognized once the image pass through the five techniques in the HOG architecture.

Figure 2.3.2.7 Histogram of Oriented Gradients Architecture (Kosuke Mizuno, 2018)



2.4 Related Work

In this section will discuss about the related works with this project. This section only focuses on face recognition using camera.

2.4.1 Approach

The first related work is called “A Face Recognition Method on the Internet of Things for Security Applications in Smart Homes and Cities” by (Nashwan Adnan Othman, 2018). In this project, they focus on protect home, office by recognizing people. In this study, they use PIR sensor to detect the movement in the specific area. After that, the Raspberry Pi will capture the images once sensor detect movement. The person face will be detected and recognized in the captured image. Finally, images and notifications will be sent to smartphone using Telegram application which is based on IoT. The system is real-time and using Local Binary Patterns Histograms (LBPH) algorithms.

The second related work is called “Smart Security System for Sensitive Area Using Face Recognition” by (Danish Ali Chowdhry, 2019). In this project, they focus on implementation of smart security system for restricted area where access is limited to people whose face are available in training database. They detect the faces by using human motions. The authority of the person who enter the sensitive area will be checked by performing face recognition. If it fails to recognize the person face, it will pass the estimated coordinate to anesthetic gun. This project used Principal Component Analysis (PCA) algorithm for face recognition.

The third related work is called “Web and Mobile based Facial Recognition Security System using Eigenfaces algorithm” by (Maria Rosario D.Rodavia, 2017). In this project, they focus on the approach in facial recognition security system, as the main access level for private establishments in order to prevent unauthorized entry of an individual. They applied Eigenfaces algorithm to facial recognition to form a basis set of all images used to build the covariance matrix. This security system is built to unlock door using face recognition, if there is suspicious face detected the system will automatically send SMS to the owner.

2.4.2 Device

In this section, we will discuss about the devices that is used related in this project.

2.4.2.1 Raspberry Pi

Raspberry Pi is a miniature computing gadget the size of a credit card, according to (Dwi Ely Kurniawan et al, 2019). Like ordinary computers on this device, we can install the Linux operating system. Raspberry Pi can be used as a web server, router, media center and others. The Raspberry Pi’s input-output is similar to a USB port, LAN, HDMI and 40 Pin GPIO headers that can be customized as required. Raspberry Pi runs on the Apache, MYSQL, database server and Yowsup Python library. The Raspberry Pi specifications used are equipped with an internal Wi-Fi device that functions to connect and send data between sensors with Raspberry Pi. Raspberry Pi

saves enough electricity, only requires 2.5A and 5V to turn it on. Data storage on Raspberry Pi does not use disks like hard drives on computers. This device uses Micro SD card which is also a data storage area on a smartphone.

2.4.2.2 CCTV

Analog camera CCTV systems have been used for a long time. According to experts, they are now the most popular form of camera mounted in the region. Consider a camera or a group of cameras connected by a dedicated set of wires to a storage system and a series of monitors. On-site video is captured and preserved. According to (Amy Raber) IP-based cameras perform the same functions as analogue cameras, but with a slew of additional features. IP cameras usually have higher-resolution videos and greater versatility, enabling users to e-mail video images for consultation. Insurance providers also choose, and in some cases require, IP systems in large organizations with many locations. IP cameras operate on an IP (internet protocol) network, which is often the same network as network on that specific place.

2.4.3 Platform

In this section, we will discuss about the platforms that is used related in this project.

2.4.3.1 Web-based

Web-based is used to access the CCTV to check the surrounding condition of the house. The user can just enter the IP address of the CCTV or the link of the CCTV to directly access the CCTV to monitor or check the surrounding of the house. In the past related work, we can see that they used a web-based to monitor and display the images from the CCTV. The web base can be access at phone or computer.

2.4.3.2 E-mail

The method of exchanging messages (“mail”) between people using electronic devices is called electronic mail (email or e-mail). Email is a form of communication that takes

place over computer networks, mainly the Internet. The store-and-forward paradigm is used in today's email systems. Messages are accepted, forwarded, sent, and stored on email servers. Users and machines are not expected to be online at the same time; they must connect to a mail server or a webmail interface to send or receive messages. The email also can be used to receive notification from a curtain system.

2.4.4 Extra Function

In this section, we will discuss about the platforms that is used related in this project.

2.4.4.1 Intruder Alert Notification

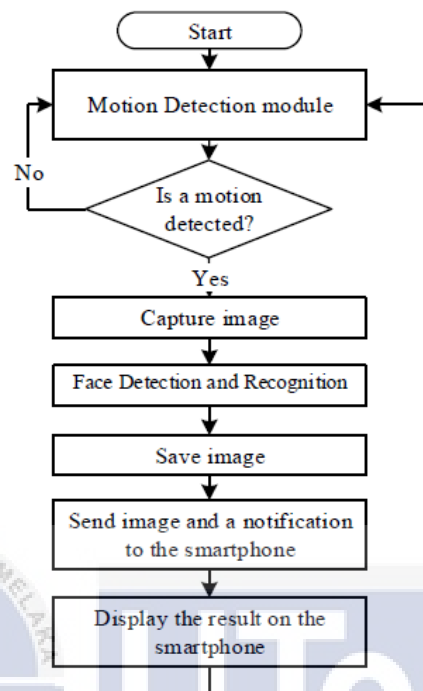
A research done by (Nashwan Adnan Othman, 2018), they created an extra function where they use intruder alert notification. Their system is configured with the CCTV which monitor and provide notifications to users using the telegram application. The user also can control the CCTV to view the surrounding of area if they receive the notification from the system.

2.4.5 Architecture of current related work

2.4.5.1 Architecture of Face recognition Security Applications in Smart Homes and Cities

When the PIR sensor detects movement, the camera is used to create the picture. The computer vision module is then used to detect and distinguish human faces in the collected photos. The information will then be sent to a smartphone. This method is extremely beneficial and necessary for securing a location. The application will not proceed to face detection and identification if there is no movement observed.

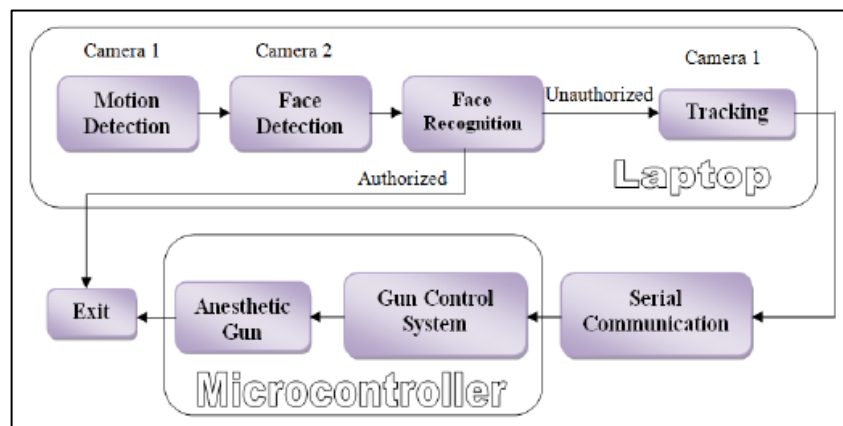
Figure 2.4.5.1 Architecture Smart Homes and Cities (Nashwan Adnan Othman, 2018)



2.4.5.2 Architecture of Smart Security System for Sensitive Area Using Face Recognition.

The security system is powered by two active cameras that are connected to a laptop. Camera 1 detects motion and subsequently monitors the moving target's location coordinates, while camera 2 identifies and extracts the face for identification. The individual is not permitted to access the sensitive area if their face is not recognized in the training database. The current coordinate is subsequently sent to the gun control system through serial connection to target the invader.

Figure 2.4.5.2 Architecture Smart Security System (Danish Ali Chowdhry, 2019)



2.4.5.3 Architecture of Web and Mobile based Facial Recognition Security System using Eigenfaces algorithm.

The architecture in figure 2.4.5.3 shows the prototype's flow or entire process, which includes the prototype's backend and frontend. The camera will begin recognizing faces using facial recognition as people enter the selected room where the software and camera are installed. Given the circumstances, it will check to see if the user's face is registered to get admission to the room. Only registered users or faces are checked, and if they are, the time of admission is recorded in the log reports. If the face spotted is unregistered, the system will send an alarm message to the administrator via the mobile application, and security staff will be able to examine this individual who is attempting to get entry to the area.

Figure 2.4.5.3 Architecture Web and Mobile based Facial Recognition
(Maria, 2017)

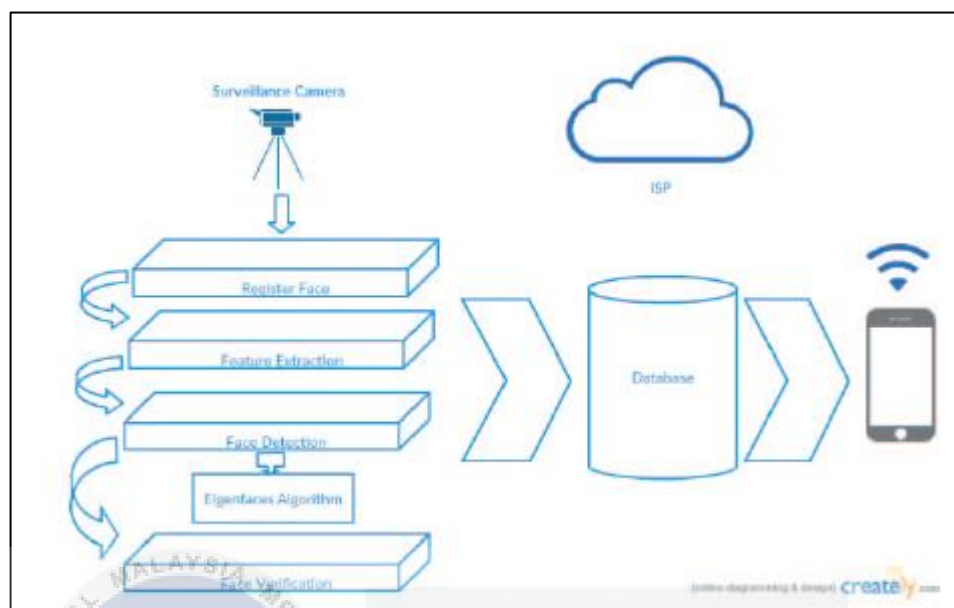


Table 2.4 The summary of related work.

Journal Name/Author	Hardware and Software	Algorithm that used by face recognition	Cost factor for the system	CCTV Specification
A Face Recognition Method on the Internet of Things for Security Applications in Smart Homes and Cities” by (Nashwan Adnan Othman, 2018)	<ul style="list-style-type: none"> • Raspberry Pi • PIR Sensor • Smartphone • Camera • Telegram Application 	Local Binary Patterns Histograms (LBPH) algorithms	Medium	Raspberry Pi 8MP Camera
Smart Security System for Sensitive Area Using Face Recognition” by (Danish Ali Chowdhry, 2019)	<ul style="list-style-type: none"> • Motion Sensor • CCTV camera • Arduino • Anesthetic Gun 	Principal Component Analysis (PCA) algorithm	Hard	1080P Wireless Wifi IP CCTV Camera Mini PTZ Camera 2MP

<p>Web and Mobile based Facial Recognition Security System using Eigenfaces algorithm” by (Maria Rosario D.Rodavia, 2017).</p>	<ul style="list-style-type: none"> • CCTV Camera • Raspberry Pi • Smartphone • Short Message Services (SMS) Application • Web-based 	<p>Eigenfaces algorithm</p>	<p>Medium</p>	<p>720P HDTV 1000tvl dome CCTV camera</p>
--	--	-----------------------------	---------------	---

Based on the table 2.4, the device and software that will be used for this smart CCTV home camera system is:

- Raspberry Pi

To use face recognition algorithm and connect to the camera for record and recognize people face.

- Raspberry Pi Camera

Use to capture or record video in front of house door.

Use to recognize people face that in front of the house door.

- Laptop

To use web for access the camera and view the camera.

- Email Application

To receive notification from the face recognition camera system.

2.5 Proposed Solutions

In this section, we will discuss about the parameters that will use in this project. These parameters are used to measure which solution is the best used for this project.

2.5.1 Parameters

Parameters are an essential component of any analysis. In simple words, a parameter is any numerical quantity that characterizes a given population or some aspect of it. This means that the parameter tells us everything we need to know about the whole society. It represents the true value that would be collected if a census is conducted rather than a questionnaire. In this project the parameter is the face recognition, sensors, notification's alert and online platform. As shown in table 2.5.1 we will review and compare each of the related work and propose a solution from this review.

Table 2.5.1 Compare related work. ✓(Yes), ✗ (No)

No	Journal Name/Author	Face Recognition	Sensors	Notification's Alert	Online Platform
1	A Face Recognition Method on the Internet of Things for Security Applications in Smart Homes and Cities” by (Nashwan Adnan Othman, 2018)	✓	✓	✓	✓
2	Smart Security System for Sensitive Area Using Face Recognition” by (Danish Ali Chowdhry, 2019)	✓	✓	✗	✗
3	Web and Mobile based Facial Recognition Security System using Eigenfaces algorithm” by (Maria Rosario D.Rodavia, 2017).	✓	✗	✓	✓

Based on table 2.5.1, for the proposed solution in this system will have face recognition that is more suitable for the home security. Other than that, the system will not have sensors because to reduce the cost and difficulty of implementing the system. Furthermore, the alert notification is important for the system to notify user as soon as possible and it should be online platform to deliver notification on the online application. So, the system will be implemented with alert notification and will be online platform.

2.5.2 Proposed parameters, algorithm and CCTV

The algorithm that proposed for this project is Histogram of Oriented Gradients (HOG). One of the reasons is HOG descriptor maintain few keys that benefits over other descriptor methods. HOG does not require a lot of GPU memory to train the data compared to the other algorithms, HOG just require CPU to train the data. Convolutional Neural Network (CNN) and OpenFace uses machine learning algorithm which require a lot of GPU memory to train data one by one. CNN and OpenFace algorithm are not suitable to use as the face recognition algorithm for this project. HOG algorithm is easy to use and efficient when compared to other matching systems. HOG is commonly utilized in face recognition tasks in computer vision.

The parameter that will be used by the HOG algorithm in this project is getting locations and outlines of eyes, nose, mouth and chin of a person. HOG will use these features to recognize people face easily. HOG is a feature descriptor that used to extract features from the image data. HOG require this feature to detect the person face more accurate compared to other algorithm. To make sure the system captures the picture of person more clearly, it requires high quality camera to give a clear view. Camera that proposed for this project is Raspberry PI High Quality Camera with 6 mm lens. This camera can clear capture people face and make it easy for recognize face.

2.5.3 Proposed Architecture

In figure 2.5.3 shows the proposed architecture for the smart CCTV home camera using face recognition. The Raspberry Pi high quality camera is used to detect and capture the person face. Once it detects the person face, it will send the picture to the Raspberry Pi to do verification. In the Raspberry Pi receive the picture, it will extract the features from the picture which is eyes, nose, mouth and chin using the HOG algorithm. After that, the picture will be verified with the pictures in the database to identify whether the image is similar to the image in the database. If the image is differed from the image in the database, the Raspberry Pi will send notification to the user through email. The user can access camera using phone or laptop.

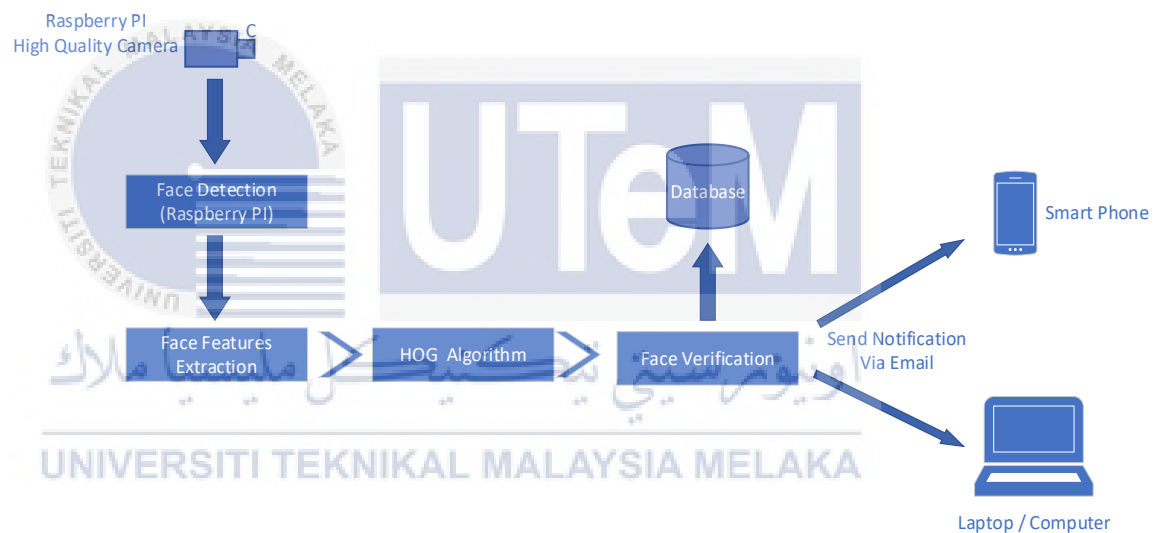


Figure 2.5.3 Proposed solution for smart CCTV home camera.

2.6 Critical Review

Table 2.6 provide a critical review for each one of the related works.

Table 2.6 Critical review of related work.

No	Journal Name/Author	Sensors	Functionality	Face recognition algorithm	Notification's alert
1	A Face Recognition Method on the Internet of Things for Security Applications in Smart Homes and Cities” by (Nashwan Adnan Othman, 2018)	PIR sensor detect the movement of specific area	In their project, they use PIR sensor to detect movement and then the face recognition camera will activate. Their system also include notification to the user.	The face recognition using LBPH algorithm which has accuracy 85.01%	The notification they use are sent via Telegram application on the smartphone
2	Smart Security System for Sensitive Area Using Face Recognition” by (Danish Ali Chowdhry, 2019)	Detect people face by motion sensor.	In their project, the system they created detect people face by motion sensor and then the face recognition activate to detect people, but they do not implement notification to alert user.	The face recognition that they implemented using PCA algorithm which has accuracy of 81.80%	They do not include notifications in their studies
3	Web and Mobile based Facial Recognition Security System using Eigenfaces algorithm” by (Maria Rosario D.Rodavia, 2017).	None	In their project, they did not use sensors. They just implement face recognition to detect people using camera. They also implement notifier	The face recognition that they implemented has accuracy of 86.05% with the low-quality camera.	The system sends notifications through SMS on the smartphone if stranger is detected.

			to notify user when detect stranger.		
--	--	--	--	--	--

Based on the table 2.6 the main significant issues that facing by the projects are algorithms or technique that used for face recognition. The way to face this problem is by implementing compatible algorithm for the face recognition with the high-quality camera to recognize people face more accurately.

2.7 Conclusion

In conclusion, we will propose a system that will recognize people face by using face recognition that more compatible to the system and notifications alert. The algorithm or technique that is proposed in this project is that we will use HOG algorithm which more compatible for home security system. The parameter for the face recognition is the person's eyes, nose, mouth and chin. For example, the system will face recognize the people at front door of house by checking the parameter of that person and compared to image in the system, the system will notify the user if the image is not match with image in system using the platform email provided via notifications.

CHAPTER 3: METHODOLOGY

3.0 Introduction

This chapter will be focusing on the methodology that used to develop the smart CCTV home camera with face recognition. There are numerous types of software development models or methods available and selecting the right one is critical since it will have a significant influence on the development of the smart home camera with face recognition system.

In this project, the goal is to gather all information related to the title of the project, which is the smart CCTV home camera with face recognition through the journal, articles, web search and books. For early stages, this project starts by stating the introduction of the project includes the objectives, problems statements and others. The chapter then progressing by identifying and analysing all information gathered in the literature review. A detail explanation on the software development model chosen will be further elaborate and the project milestones and Gantt chart for this project will be attached in this chapter.

3.1 Methodology

The waterfall software development methodology, which is part of the Software Development Life Cycle (SDLC) method, chosen for this project. This model is appropriate since the criteria are well-documented, precise, and well-defined.

3.1.1 Software Development Life Cycle (SDLC)

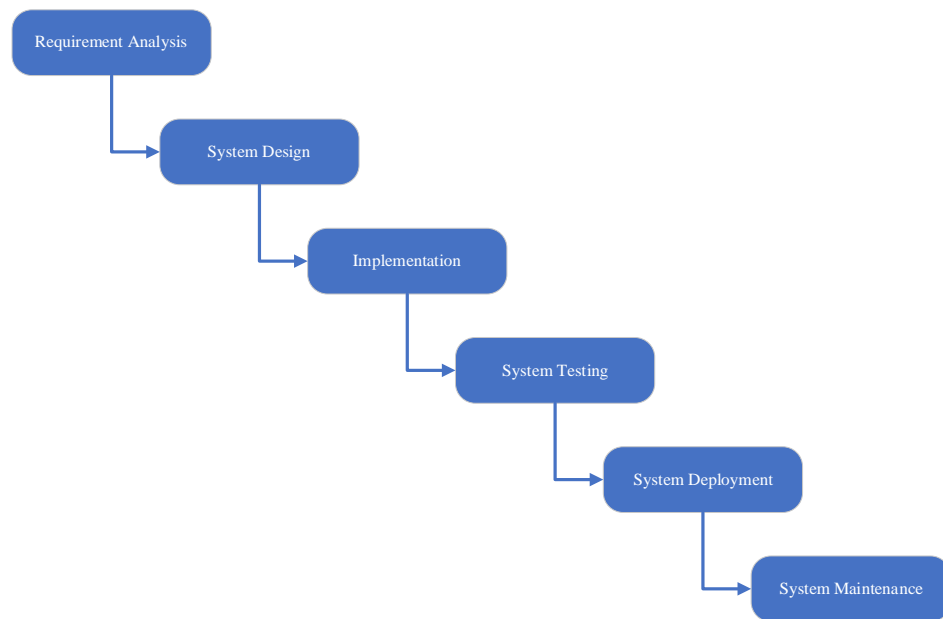
Waterfall is the most widely used, conventional and sequential technique in the System Development Life Cycle (SDLC). Although most reviews of this approach believe it to be an old approach, it is quite useful in assisting the user, author and developer in fully comprehending the system. It necessitates a great deal of structure and documentation up front, which is separated into numerous stages, each with its own set of procedures and data.

This technique is rigid, which means that every piece of information and step in this process need a thorough explanation and justification at the outset. Any modifications made in the midst of the implementation process will necessitate a complete restart of the technique, including re-stating and evaluating the preformation data.

3.1.2 Waterfall Model

The waterfall model is a traditional system development life cycle model that uses a linear and sequential method to design a system. The model is called a waterfall because it progresses methodically from one phase to the next in a downward direction. The output of one phase is utilized as the input of the following phase in this model, which is separated into many phases. Determine the project's needs and scopes, analyze those needs, design, develop, test, deploy, and maintain are the basic phases of this methodology. Each stage is laid out in a waterfall pattern, with the early stages at the top and falling into each step as the project progresses. Figure 3.1.2 shows the flow of waterfall model phases.

Figure 3.1.2 Flow of Waterfall Model Phase.



The Waterfall model was chosen for this project because of the projected system's evolution. In general, each stage must be completed before moving on to the next, which might assist with the project's needs and processes. Because all of the facts and needs are declared at the beginning of the stages, the approach of this model is highly ideal for this suggested system, allowing the progressions to grow more thorough, increasing the other stages development with all of the information declared and obtained.

3.1.2.1 Requirement Analysis Phase

The requirement analysis phase of the prototyping model is the first stage of methodology. This phase is critical because it analyses and collects all project needs, particularly functional needs. The project objectives must be met by all functional and non-functional requirements. Any ambiguous requirements will result in a poorly defined project scope, which may cause issues later in the development process. Before developing a project, it is important to identify and research more about the device. This is to ensure there are a good flow and planning in the project. The project's requirements are outlined below:

- Software Requirements
 - i) Raspberry Pi OS
 - An open-source program for writing and uploading code to a microcontroller.

- Hardware Requirements
 - i) Raspberry Pi 4 Model B
 - A microcontroller kit that can be used to create a software that can interface with camera devices.

 - ii) Raspberry Pi High Quality Camera
 - A camera that offers significantly more pixels than 8MP Pi camera V2 and can be installed in Raspberry Pi.

 - iii) Raspberry Pi 6mm Lens Camera
 - Camera lens that can give clear view when capture image and it installed with Raspberry Pi High Quality Camera.

 - iv) MicroSD (32GB)
 - A removable flash memory card uses for store data in the Raspberry Pi.

- System Requirements
 - i) Notification
 - Phone or Computer will receive notification through email from the system.

3.1.2.2 System Design Phase

The architecture of the project was addressed at this point. The first phase's requirement definition is examined in order to build a system design. Using the requirements specified in the first step, the design will assist in designing the overall system architecture. The prototype design is to use Raspberry Pi camera to detect the people face and send the captured image to Raspberry Pi for further process. The face algorithm will be written for the camera to detect and recognize face. Then, the connection between camera and microcontroller is designed and drafted.

In this design phase, the algorithm that will be applied for the system is Histogram of Oriented Gradients (HOG) algorithm. Feature descriptors called Histogram of Oriented Gradients (HOG) are used in computer vision and image processing to recognize faces. The method counts the number of times a gradient orientation appears in a certain area of an image. Edge orientation histograms, scale-invariant feature transform descriptors, and shape contexts are all comparable methods, but this one differs in that it is computed on a dense grid of evenly spaced cells and employs overlapping local contrast normalization for enhanced accuracy. The basic idea behind the Histogram of Oriented Gradient descriptors is that the distribution of intensity gradients or edge directions may be used to characterize the look and form of local objects within an image. Local histograms can be contrast-normalized for better accuracy by generating an intensity measure across a wider portion of the image, known as a block and then using this value to normalize all cells inside the block. As a result of this normalization, the invariance to changes in lighting or shadowing is improved.

The HOG algorithm require dataset that allow the algorithm to recognize people face. The database of Histogram of Oriented Gradient requires 35 images of same person face but with various facial expressions. The images also contain various distances and lighting conditions. In comparison to other descriptor approaches, the HOG descriptor has a few major advantages. Except for object orientation, the HOG descriptor is invariant to geometric and photometric alterations since it acts on isolated cells. The algorithm will extract the outline of the person face features from the images which is eyes, nose, mouth and chin. The image pixel values will be normalized, and one-hit wonder the category column will be encoded. After that, the HOG algorithm

creates histogram graph based on the geometric. Lastly, the algorithm makes recognition and train the model using the dataset.

3.1.2.3 Implementation Phase

During this phase, using Raspberry Pi OS which manages the coding development environment, all of the program code will be created and tested on the Raspberry Pi. The camera and microcontroller connection are validated using the previously prepared coding to recognize correctly. Then, the system will send notification to email which also required further configuration in order to connect to the email and can be checked in phone or computer.

The algorithm in the Raspberry Pi board will integrate with Raspberry Pi High Quality Camera to capture images. In the Raspberry Pi implemented OpenCV coding to detect people face using the Raspberry Pi camera. OpenCV algorithm use machine learning algorithm for search people faces from the image that captured from the Raspberry Pi camera because faces are complicated. Once the OpenCV detect face from the image, it will pass the following image to the Histogram of Oriented Gradient (HOG) algorithm to recognize the image by doing classification. In the HOG algorithm, the image will pass through HOG computation module in the algorithm.

Once the HOG algorithm receive image, it scans on the input of the image based on the image. The image is partitioned into cells, with each cell's pixels collecting a histogram of gradient orientations. The features in the image will be extracted by the HOG algorithm during the histogram gradient process. After that, histogram normalization will be used to improve illumination invariance by collecting a measure of local histogram energy over blocks and utilizing the findings to normalize all cells in the block. The histogram normalization is a one of process in the HOG computation module. Once the input image is divided into block and it will be plot as histogram graph that will be easy to do classification.

During the classification stage, the normalized histograms (HOG features) are gathered. The face image classification, the obtained features are put into a linear. Every image from the dataset will be classified one by one with the input image that normalized as a histogram. To store the previous descriptors, the HOG algorithm must require a lot of memory. The classification module in this work employs a cell-based

scanning structure to decrease memory use, similar to several earlier states of the art. Finally, it categorizes the output using an activation function and the algorithm will make recognition people face. The system will send notification through email when the algorithm recognizes unknown people face.

3.1.2.4 System Testing Phase

System testing involves putting all of the designed materials, as well as the installed hardware and software, to the test. For this project, the face recognition will be tested, detect if any error occurs during recognition. After all of the face recognition system are tested and fully functional, the code for notification will be tested, whether the user receive email according to the system requirement.

The HOG algorithm will be tested by adding people or user image which is face up to 35 images in the database. Every image that added in the database have various facial expressions, distance and lighting. Once the database added, the algorithm tested by letting the user pass through the front of the camera to let the system recognize. If the algorithm could not recognize the user, more images of the user will be added to train the algorithm to recognize the user. The user also passes in front of camera many times with different facial expressions, so the algorithm can recognise accurately.

3.1.2.5 System Deployment Phase

After all of the functional and non-functional aspects of the hardware and system have been thoroughly tested, the project is now ready for deployment and usage in a real circuit design prototype. This phase begins after the testing phase is free of bugs, errors, or other difficulties, and it is ready to be deployed in production and utilized by end users.

3.1.2.6 System Maintenance Phase

The goal of the maintenance phase is to ensure that the system and hardware are in good working order. This phase requires the author to monitor the product that has already been produced and give bug fixes, as well as be prepared to accept feedback from the end user on the product. When the product is available to the end user, it is required to supply fresh updates and fixes.

3.1.3 Project Milestones

Project milestones will illustrate how far the project has progressed based on a defined timetable. Each work must be completed within the specified time frame in order for the project to be completed on time. The milestones might help the developer determine whether the project is being carried out properly. Table 3.1.3 is the milestone of the project.

Table 3.1.3 Milestone of Project

WEEK	ACTIVITY	NOTE / ACTION
< W0 (< 21/3)	Select a suitable project topic and potential Supervisor	• Action – Student
W 1 (15/3 → 21/3) Meeting 1	Proposal PSM: Discussion with Supervisor	Deliverable – Proposal Action – Student
	Proposal assessment & verification	• Action – Supervisor
W2 (22/3 → 28/3)	Proposal Correction/Improvement	• Action – Student
	Proposal submission to Committee via email	
	Proposal Approval List of Supervisor/Title	• Action – PSM/PD Committee
W3 (29/3 → 4/4) Meeting 2	Proposal Presentation & Submission via PSM ULearn	Deliverable – Proposal Presentation (PP) and Completed Proposal Form Action – Student
	Chapter 1 (System Development Begins)	• Action – Student
W4 (5/4 → 11/4)	Chapter 1	Deliverable – Chapter 1 Action – Student, Supervisor
W5 (12/4 → 18/4)	Chapter 2	• Action – Student
W6	Chapter 2	• Deliverable – Chapter 2

(19/4 → 25/4) Meeting 3	Project Progress	<ul style="list-style-type: none"> Progress Presentation 1 (PK 1) Action – Student, Supervisor
W7 (26/4 → 2/5)	Chapter 3	<ul style="list-style-type: none"> Action – Student
W8 (3/5 → 9/5)	Chapter 3	<ul style="list-style-type: none"> Deliverable: Chapter 3 Action – Student, Supervisor
W9 (10/5 → 16/5)	MID SEMESTER BREAK	
W10 (17/5 → 23/5) Meeting 4	Chapter 4	<ul style="list-style-type: none"> Action – Student
	Project Progress	<ul style="list-style-type: none"> Progress Presentation 2 (PK 2) Action – Student, Supervisor
W11 (24/5 → 30/5)	Project Demo	<ul style="list-style-type: none"> Action – Student, Supervisor
W12 (31/5 → 6/6)	Project Demo PSM1 Report	<ul style="list-style-type: none"> Action – Student, Supervisor
W13 (7/6 → 13/6) Meeting 5	Project Demo PSM1 Report Schedule the Presentation	<ul style="list-style-type: none"> Action – Student, Supervisor Action – PSM/PD Committee Presentation Schedule
W14 (14/6 → 20/6)	Project Demo	<ul style="list-style-type: none"> Deliverable – Complete PSM1 Draft Report Action – Student, Supervisor
W15 (21/6 → 27/6) Final Presentation	FINAL PRESENTATION Submission of the PSM1 Report onto the PSM ULearn.	<ul style="list-style-type: none"> Action – Student, Supervisor, Evaluator, PSM/PD Committee
W16 (28/6 → 4/7)	REVISION WEEK Correction on the draft report. Submit PSM1 Logbooks to PSM ULearn. Submit an EoS Survey form.	<ul style="list-style-type: none"> Deliverable – Complete PSM1 Logbooks Action – Student, Supervisor
		<ul style="list-style-type: none"> EOS Survey Action – Student
W17 & W18 (5/7 → 18/7)	FINAL EXAMINATION WEEKS	

3.1.4 Project Gantt Chart

Gantt Chart is a simplified form of project milestones that shows a graphical representation of the project's reference point. The Gantt Chart serves as a graphical roadmap for project completion each week. Figure 3.1.4 Gantt chart for project

Table 3.1.4 Gantt chart for project.

ACTIVITY	PERIODS (WEEKS)													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Idea Discussion	■	■												
PSM Proposal Submission and Correction			■	■										
Gathering all the requirements					■	■								
Design the system							■	■						
Apply code in the system									■					
Testing the product									■	■				
Evaluate the product											■	■		
PSM Report													■	
Final Presentation														■

UNIVERSITI TEKNIKAL MALAYSIA MELAKA
 اونیورسیتی تکنیکل ملیسیا ملاک

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

3.2 Conclusion

In conclusion, this chapter described the development project process, including the chosen SDLC strategy. Furthermore, each step expounded on the selected method in relation to the present development project on what and how the phases would be established. This project also examines the significance of project milestones and a project Gantt chart, both of which are essential components in ensuring that a project is developed in a disciplined and orderly way. The analysis and design for the project will be discussed in the following chapter, which will contain a full problem analysis, hardware and software requirements for the project, and project design.

CHAPTER 4: DESIGN

4.0 Introduction

The analysis and design for this project will be discussed in this chapter. The technique employed, the necessity, and the distribution platform all required a problem for analysis and design. The issue will be examined during the analysis and design phase, and the project will be evaluated. The term "analysis" refers to a study of the entire element as well as the relationships that make up the whole. Most of current existing systems for products using smart CCTV home camera with face recognition have the same thing or concept to achieve. The second step of a project's development is design. One or more designs are generated in the design stage to provide a pleasing design. The type of project being produced determines the product design stage.

4.1 Problem Analysis

An investigation of the causes of an occurrence, issue, or failure is known as a problem analysis. This is done to find system, process, method, design, and cultural improvements. This project is a smart CCTV home camera with face recognition that is being programmed in a Microcontroller (Raspberry Pi). Since the smart system is a simple and convenient tool, the user can use their phone or computer to access camera anytime.

4.2 Requirement Analysis

This section will go through the functional requirements that are part of the project. The process of discovering, obtaining, and comprehending system requirements and their properties is known as requirement analysis. The only components of this section

that we will examine are data requirements, software requirements, and hardware requirements in order to create the project and better understand what equipment we will utilize in this project.

4.2.1 Data Requirement

The data that receiving from the camera and that will be send to the microcontroller (Raspberry Pi) is the image of valid user face. First of all, the data is added in the algorithm which is some images of the celebrity for the testing purposes. The data or the images are taken from the following website <https://www.kaggle.com> which provide a lot of images that have same person face with a different expression. The website provides up to 50 to 100 images of the person face which suitable to use as dataset for this system. The dataset is constructed by collecting the images from Kaggle website and the images will be added in the training phase in the HOG algorithm to train the images and prepare the model for the recognition. According to the (O.Deniz, 2018) the quantity of images or data that require for the HOG algorithm is minimum 30 images. The quantity of images that will be used for this image is 35 for training module.

4.2.1.1 Images of valid user face.

Images of valid user faces is the data that will be used by HOG algorithm for face recognition. The database requires 35 images of same person face but with various facial expressions. By collecting the images from Kaggle website and the images will be added in the training phase in the HOG algorithm to train the images and prepare the model for the recognition. The images also contain various distances and lighting conditions. The images should be clear enough, so the training process will go smoothly. The more the valid user, the more images should be added in the database, to ease the algorithm to do face recognition.

4.2.2 Software Requirement

The software requirements are a list of the target system's characteristics and functions. Users' expectations of the software product are expressed in requirements.

4.2.2.1 Raspberry Pi Operating System



Figure 4.2.2.1 Raspberry Pi OS Logo

Raspberry Pi OS is a Debian-based free operating system that is optimized for the Raspberry Pi device. Over 35,000 packages are included with Raspberry Pi OS, which are precompiled software bundles in a tidy format for simple installation on your Raspberry Pi. The Raspberry Pi microcontroller require Raspberry Pi OS, it is necessary to install the operating system into the device.

4.2.3 Hardware Requirement

4.2.3.1 Raspberry Pi 4 Model B

The Raspberry Pi is a small, low-cost computer the size of a credit card that connects to a computer display or television and utilizes a conventional keyboard and mouse. It's a powerful small gadget that allows individuals of all ages to learn about computers and programming languages like Scratch and Python. The Raspberry Pi's hard drive is an SD card placed into a slot on the board to store data and install operating system. The model Raspberry Pi 4 has extra specification than other model Raspberry Pi which is the RAM up to 4GB. It is powered by using USB power supply and have camera slot, even HDMI port for monitor or TV.

Features:

- Broadcom BCM2711 Quad-core A72 64-bit SoC
- 4GB LPDDR4 SDRAM memory
- Bluetooth 5.0, WLAN 2.4GHz/5.0GHz and Gigabit Ethernet connectivity.
- 2 x USB 3.0 ports and 2 x USB 2.0 ports
- 40-pin GPIO Header connection

- Dual micro-HDMI ports (up to 4Kp60 UHD supported)
- H.265 decode (4kp60) H.264 decode (1080p60)
- MIPI DSI display port, CSI camera port and 4 pole stereo output and composite video port.
- microSD card slot for loading operating system and data storage
- PoE enabled (PoE HAT required, sold separately)
- 5V3A USB-C power supply required.



Figure 4.2.3.1 Raspberry Pi 4

4.2.3.2 Raspberry Pi High Quality Camera

For industrial and consumer applications, such as security cameras, that demand the greatest levels of visual fidelity and integration with specialized optics, the Raspberry Pi High Quality Camera is an alternative to the Camera Module v2. It is compatible with all Raspberry Pi models. The Raspberry Pi High Quality Camera has a higher resolution (12 megapixels compared to 8 megapixels) and sensitivity (approximately 50 percent more area per pixel for improved low-light performance) than the previous Camera Module v2, and it's compatible with interchangeable lenses in both C- and CS-mount form factors. Third-party lens adapters can accommodate other lens form factors.

Features:

- Sensor: Sony IMX477R stacked, back-illuminated sensor
12.3 megapixels

7.9 mm sensor diagonal

1.55 μm \times 1.55 μm pixel size

- Output: RAW12/10/8, COMP8
- Back focus: Adjustable (12.5 mm–22.4 mm)
- Lens standards: CS-mount
C-mount (contain C-CS adapter)
- IR cut filter: Integrated.
- Ribbon cable length: 200 mm
- Tripod mount: 1/4"-20



Figure 4.2.3.2 Raspberry Pi High Quality Camera.

4.2.3.3 Raspberry Pi 6mm Lens Camera

This Raspberry Pi 6mm lens is good for simple photography. Because it can concentrate on things at such close distances, it may also be utilised for macro photography. The broad field of vision, the 6mm 3MP lens is ideal for CCTV-style applications, allowing us to see more of the total target area. The Raspberry Pi 6mm lens also compatible to mount in Raspberry Pi High Quality Camera.

Features:

- Image format: 1/2"

- Focal length: 6 mm
- Resolution: 3 MegaPixel
- Aperture: F1.2
- Mount: CS
- Field Angle: 63°
- M.O.D.: 0.2 m
- Back Focal Length: 7.53 mm
- Dimension: 30 x 34 mm
- Weight: 53 g



Figure 4.2.3.3 Raspberry Pi 6mm Lens Camera

4.2.3.4 MicroSD (32GB)

A detachable flash memory card specially developed for mobile phones. It may be used to store a variety of things, including images, movies, music, and software, just like any other flash memory card. A microSD card is roughly the size of a fingernail and is one of the smallest memory card types available.

Features:

- Capacity: 32 GB
- Compatibility: Compatible with microSDHC and microSDXC supporting host devices
- Dimensions (L x W x H): 0.04" x 0.59" x 0.43"
- Sequential Read Performance: Up to 120MB/s



Figure 4.2.3.4 MicroSD (32GB)

4.3 Detailed Design

In this part we will discuss about the project detailed design. The implementation component of what is regarded as a system and its sub-systems in architectural designs is dealt with in detailed design. It goes into further into modules and their implementations. It specifies the logical structure of each module as well as their communication interfaces with other modules.

4.3.1 Circuit Design

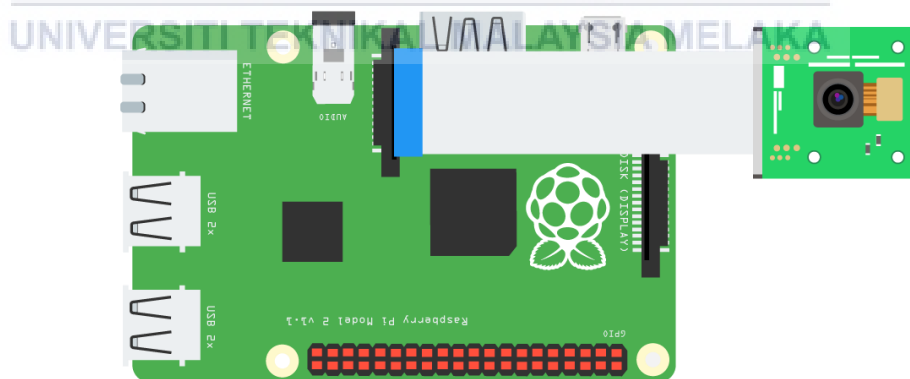


Figure 4.3.1 Circuit Design of Raspberry Pi Camera

In figure 4.3.1 it shows that the Raspberry Pi High Quality Camera connected to the Raspberry Pi. The camera is mounted using the ribbon cable from Raspberry Pi board

module to the Raspberry Pi High Quality Camera module. The ribbon cable connected to the camera slot that designed in the Raspberry Pi board. The Raspberry Pi board is connected to the USB-C power supply, so it can run the whole circuit.

4.3.2 Architecture Design

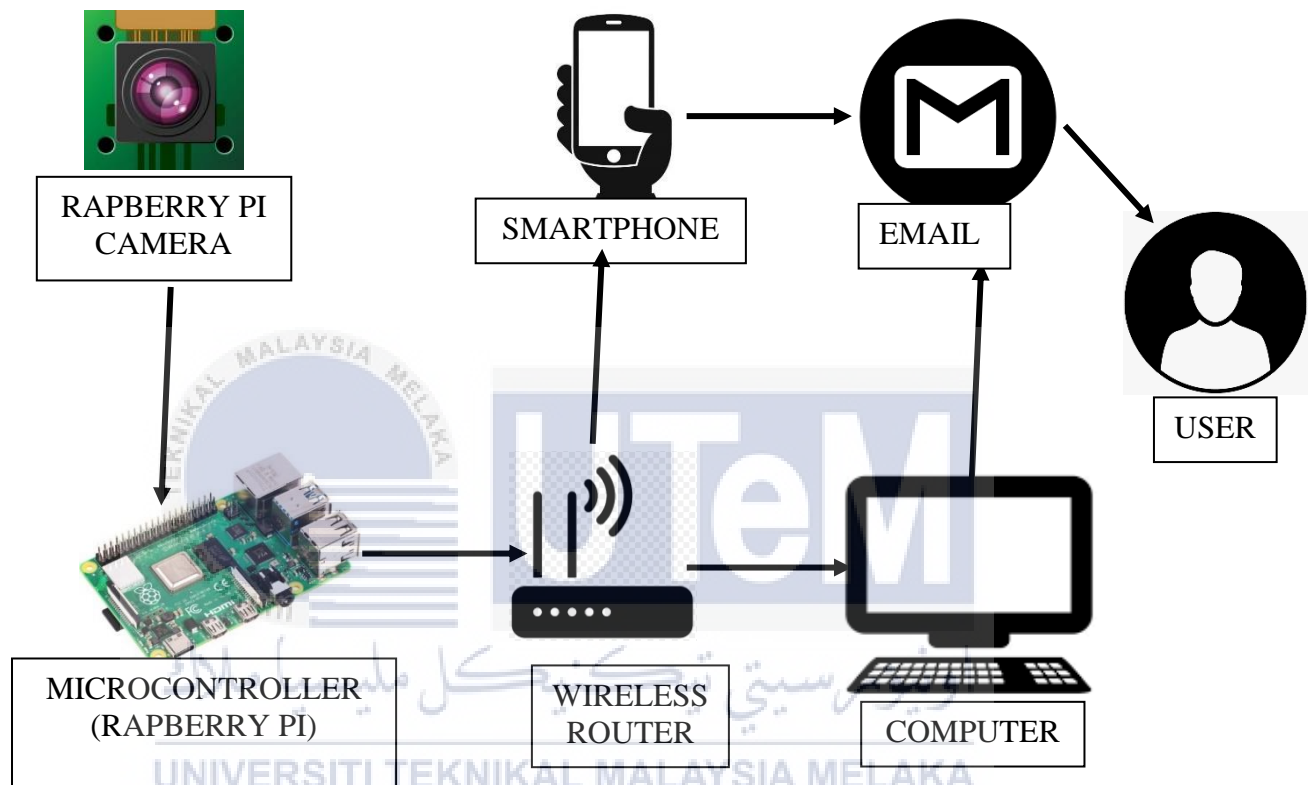


Figure 4.3.2 Architecture of System

The general flow of the system architecture of this advanced project will be discussed in this portion. The figure 4.3.2 shows the system design process consists of two parts, the first is the design of hardware capture and recognize the image using the camera. The second is sending notification to the user. According to the architecture, the Raspberry Pi camera will capture the image of people face and send it to the Raspberry Pi board. In the Raspberry Pi board contain HOG algorithm. The algorithm will classify the images that send by Raspberry Pi camera. Once the algorithm recognizes people face in the image, if the person is unknown the system will send notification to the user through the email. So, the user will be alert regarding to the situation of surrounding the house.

4.3.3 General Flow Chart

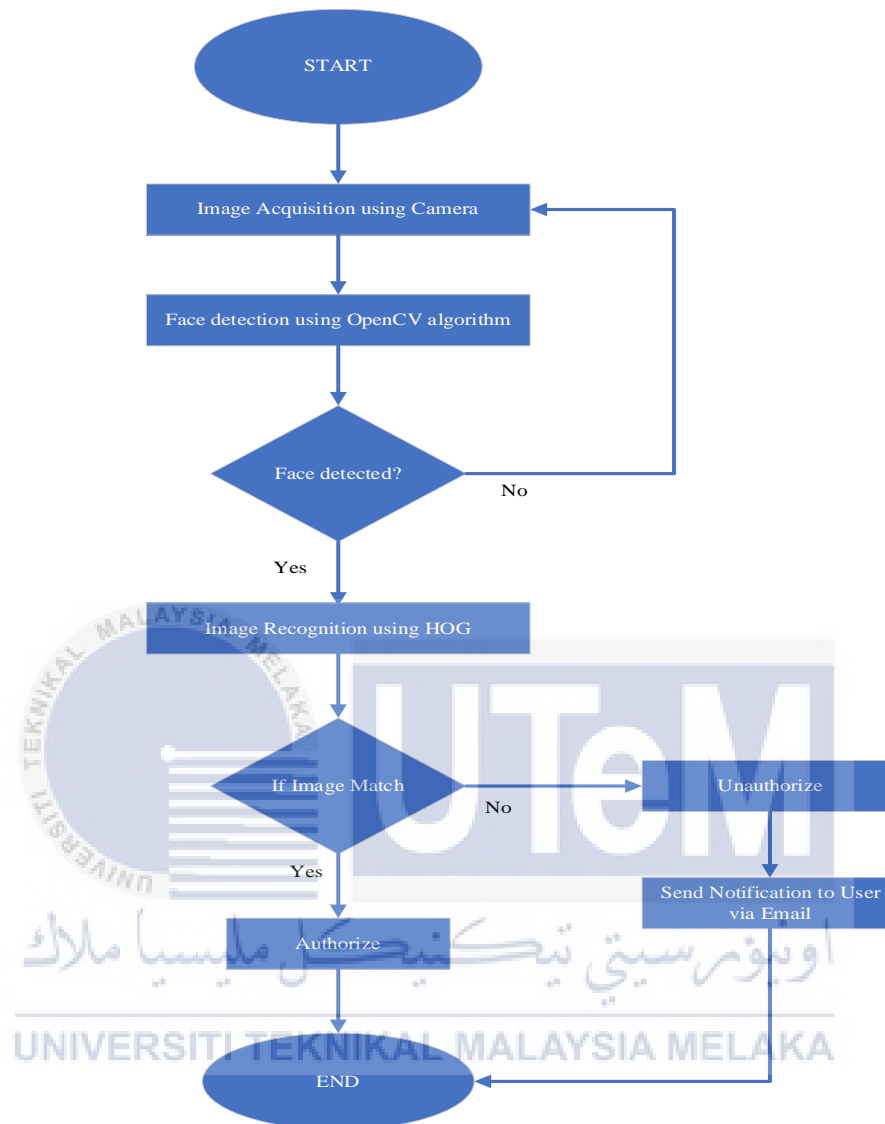


Figure 4.3.3 Flow Chart of system

In figure 4.3.3 shows, the flowchart of smart CCTV home camera with face recognition system. According to the flow chart, the system will start by image acquisition using the Raspberry Pi camera. The images will be sent to the OpenCV algorithm that coded in the Raspberry Pi board, to detect faces from the image. If there are no faces detected from the image, the process will be return back to the image acquisition until the OpenCV algorithm detect face in the images. If the algorithm detect face in the image, the image will be pass to the next stage which is extract features. The person face that found in the image will be extracted by the HOG algorithm. Once the feature from the face is extracted, the image will pass through

every stage in the HOG algorithm to classify the image. The HOG algorithm uses database that already added in the system which is valid user images. The HOG algorithm will predict the image and identify matches between database images. If the image match with database images, the process will be end. If the image is not match with database and the person is unknown, the system will send notification to the user through email and notify the user.

4.3.3.1 Flow Chart for OpenCV module

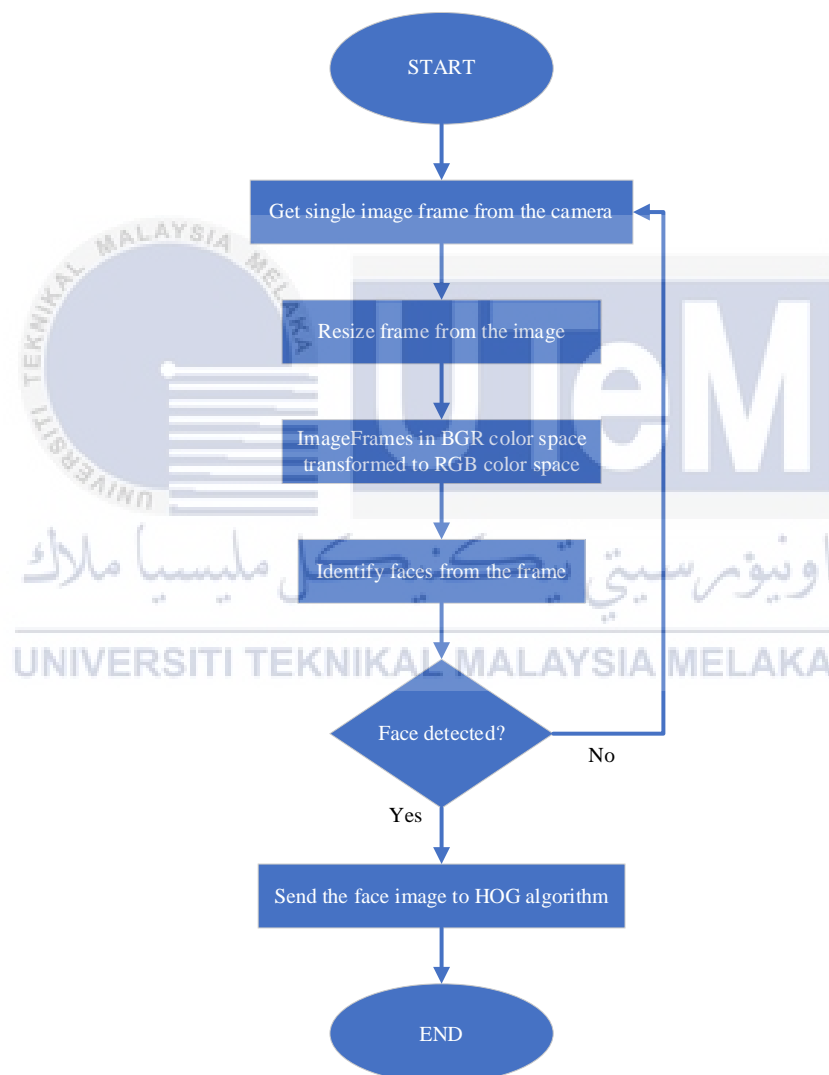


Figure 4.3.3.1 Flow Chart of OpenCV module

In figure 4.3.3.1 shows, the flow chart of OpenCV module which is part of flowchart of smart CCTV home camera with face recognition system. According to the flow chart, the OpenCV will get single image frame from the Raspberry Pi camera. Once OpenCV get the image frame, it will resize frame from the image into 1/4 size for

fasten the process of face recognition. After the resize of frame, the OpenCV will convert the image from BGR colour space into RGB colour space which the face recognition is required. Once the colour change is done, the OpenCV will identify faces from the frame. If the face is detected by the OpenCV from the image frame, the face image will be sent to the HOG algorithm to continue the face recognition process. Meanwhile, if there is no face found in the image frame, the OpenCV will redirect the process to the early stage of OpenCV module which is the OpenCV will get back image frame from the Raspberry Pi camera and rerun the process.

4.3.3.2 Flow Chart for HOG algorithm module

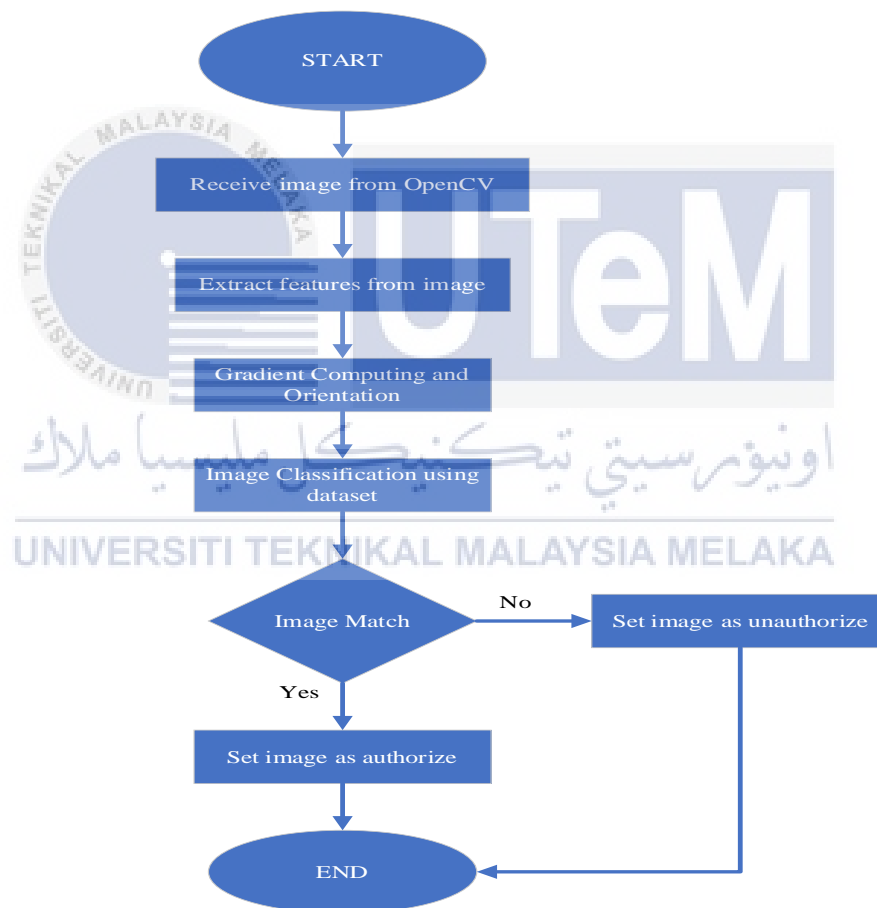


Figure 4.3.3.2 Flow Chart of HOG algorithm module

In figure 4.3.3.2 shows, the flow chart of HOG algorithm module which is part of flowchart of smart CCTV home camera with face recognition system. According to the flow chart, the HOG algorithm receive image from the OpenCV to process face

recognition. Once receive the face image, the HOG algorithm extracts the features from the face which is mouth, eyes, nose and chin. For every feature, the corresponding module of gradient is created and calculate horizontal and vertical convolution. Every gradient is stored as an appropriate orientation image. Once the gradient is calculated for each square-blocks in the image, the image is classified with the dataset that trained in the algorithm. If the image is recognized, the image will be set as authorize image and the process will be end. If the image is could not recognize, the image will be set as unauthorize and the process will be end.

4.3.3.3 Flow Chart for Modified HOG algorithm module

In figure 4.3.3.3 shows, the flow chart of modified HOG algorithm module which is this module are applied in the smart CCTV home camera system. According to the flow chart, the process of HOG algorithm bit similar to the original HOG algorithm but it has been modified during the training process. When the HOG algorithm receive image from OpenCV to process the face recognition. The HOG algorithm extracts the features from the image which is mouth, eyes, nose and chin. Once extraction is done, the data of each user will be stored in the face encodings. Compare to the modified HOG algorithm the original HOG algorithm only consists of one face encodings which will store all data of all users in one face encodings, that will make the HOG algorithm more difficult to do recognition with high accuracy. Therefore, the modified HOG algorithm has separated face encodings for each user to store their face data that trained. During this process HOG algorithm can start to recognize by using 80% of the data from the dataset. The next process will be the image classification using the image data with the image that captured from the camera. If the image is recognized, the image will be set as authorize image and the process will be end. If the image is could not recognize, the image will be set as unauthorize and the process will be end.

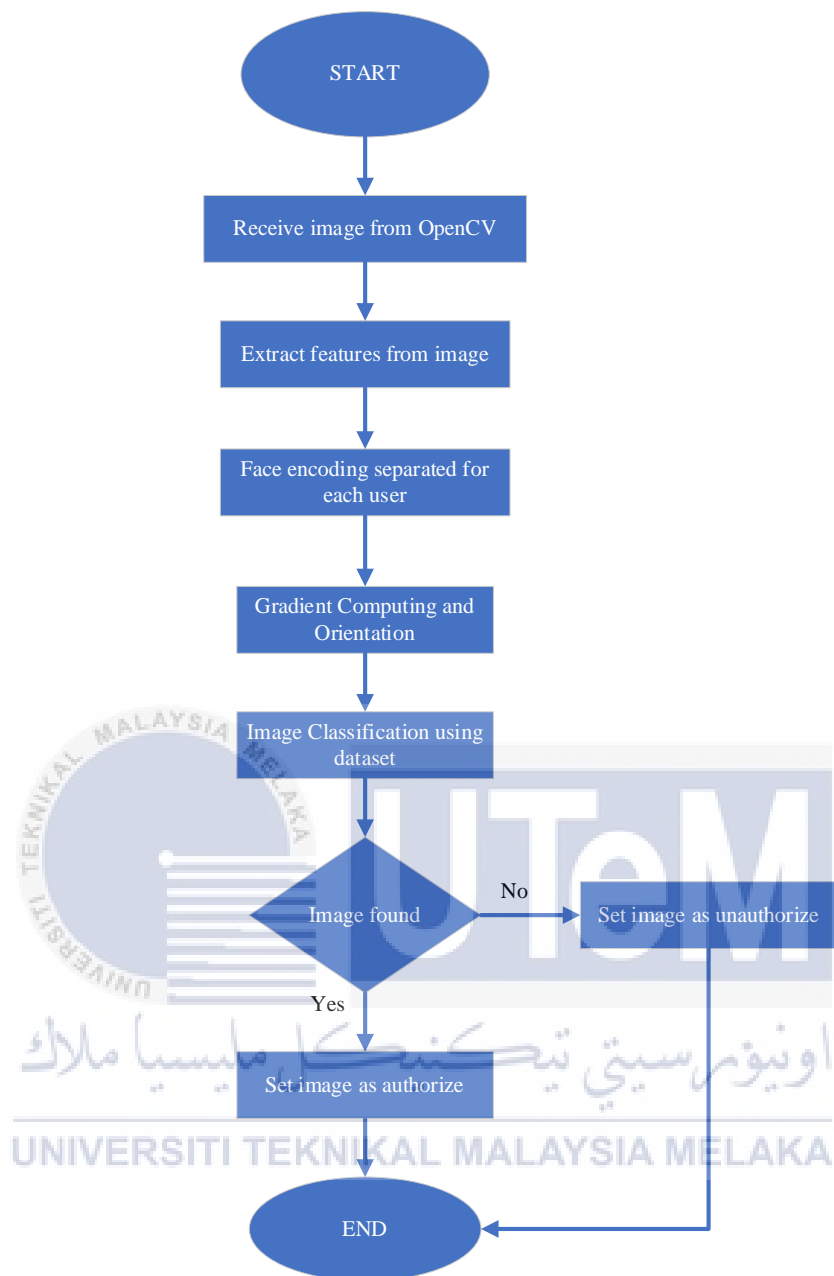


Figure 4.3.3.3 Flow Chart of HOG algorithm module

4.4 Conclusion

In conclusion, this chapter explained the importance of the requirement analysis and how it is essential to know the designs of every aspect for this project. Moreover, this chapter explained the importance of the flowchart and how the workflow of this project that will take into place. This chapter tackle about the analysis and design for the project, which includes the detailed problem analysis, hardware and software requirements that will be used in the project and the project design.



CHAPTER 5: IMPLEMENTATION

5.0 Introduction

This chapter discusses the activity involved during implementation phase of developing smart CCTV home camera using face recognition. This phase focuses on how face is recognized using Raspberry Pi camera and applied using several setups. Coding for the hardware and face recognition are applied in this chapter. Every detail on how the implementation and configuration are also established in this chapter.

5.1 Software Development Environment Setup

The Raspberry Pi board and Raspberry Pi high quality camera coding must be configured before linking it to the Histogram of Oriented Gradient (HOG) algorithm. HOG algorithm is chosen because it does not require GPU memory to train data, just require CPU to train data. HOG algorithm is easy to use and efficient when compared to other algorithms. This project requires Raspberry Pi 4 board to store the data, which is the images of valid user, that will be trained and used by the HOG algorithm. The Raspberry Pi has at least 4 GB memory to avoid the system from lagging and also it requires 32 GB MicroSD to store a lot of data. Finally, the installation take place after everything have been setup according to the project.

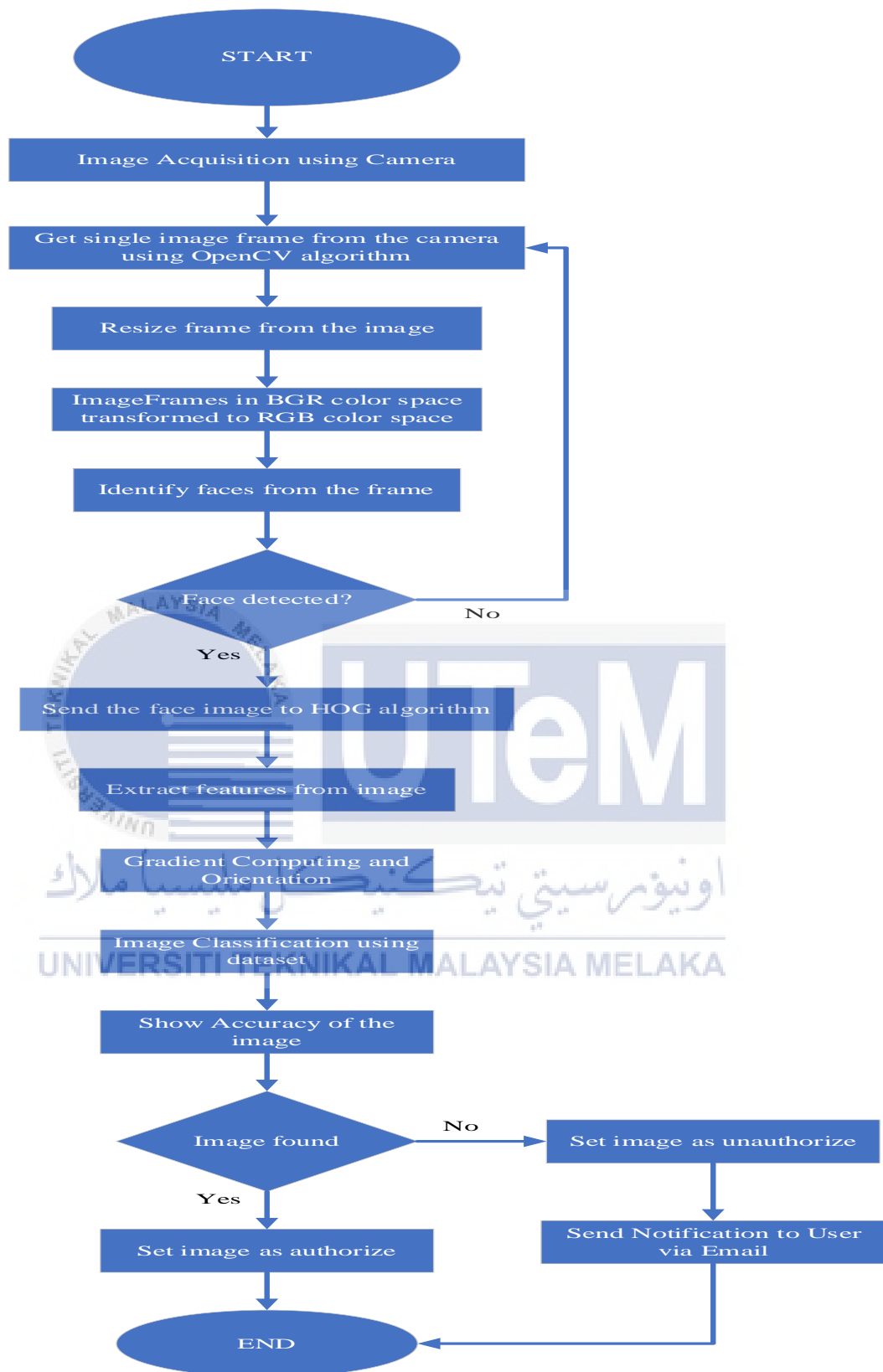


Figure 5.1 Flowchart of system

In figure 5.1 shows, the flowchart of smart CCTV home camera with face recognition. According to the flow chart the system will start to capture image using the Raspberry Pi camera. The OpenCV will get single image frame from the Raspberry Pi camera. Once OpenCV get the image frame, it will resize frame from the image into 1/4 size for fasten the process of face recognition. After the resize of frame, the OpenCV will convert the image from BGR color space into RGB color space which the face recognition is required. Once the color change is done, the OpenCV will identify faces from the frame. If the face is detected by the OpenCV from the image frame, the face image will be sent to the HOG algorithm to continue the face recognition process. Meanwhile, if there is no face found in the image frame, the OpenCV will redirect the process to the early stage of OpenCV module which is the OpenCV will get back image frame from the Raspberry Pi camera and rerun the process.

The HOG algorithm receive image from the OpenCV to process face recognition. Once receive the face image, the HOG algorithm extracts the features from the face which is mouth, eyes, nose and chin. For every feature, the corresponding module of gradient is created and calculate horizontal and vertical convolution. Every gradient is stored as an appropriate orientation image. Once the gradient is calculated for each square-blocks in the image, the image is classified with the dataset that trained in the algorithm. The accuracy of the image will be shown on the frame. If the image is recognized, the image will be set as authorize image and the process will be end. If the image is could not recognize, the image will be set as unauthorize and the system will send notification to the user through email and notify the user.

5.2 Software Configuration Management

5.2.1 Configuration environment setup

In this part, figure 5.2.1 figuratively explains the deployment of the system.

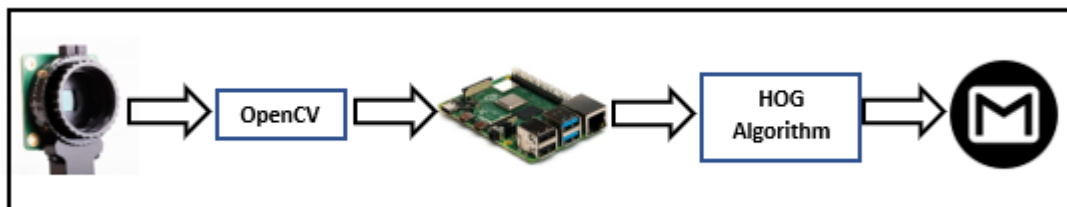


Figure 5.2.1 Deployment of the face recognition

Figure 5.2.1 shows the system deployment of face recognition using Raspberry Pi board and Raspberry Pi High Quality Camera. Raspberry Pi board is a platform to write the code that will allow hardware to work. The Raspberry Pi camera will transfer the picture that captured to the Raspberry Pi board after the OpenCV detect face of a person in the video frame. The command received by Raspberry Pi board from Raspberry Pi High Quality Camera is sent through FFC cable.

5.2.2 OpenCV Coding in Raspberry Pi Board

Code snippets in figure 5.2.2 shows how the OpenCV detect the faces of the people from the video frame of the Raspberry Pi Camera. The figure 5.2.2 shows pseducode where the coding will detect people face from video frame. The line 300 shows that OpenCv coding will grab a single frame from the video that being captured by the Raspberry Pi High Quality camera. Once captured the frame of the video in line 303 shows, it resize the frame of the video into $\frac{1}{4}$ size for faster up the face recognition process. In line 306 the image will covert from BGR color to the to RGB color where its easy for the face recognition to detect faces in the frame.

```

facerec_pi_test.py
...
298 while True:
299     # Grab a single frame of video
300     ret, frame = video_capture.read()
301
302     # Resize frame of video to 1/4 size for faster face recognition processing
303     small_frame = cv2.resize(frame, (0, 0), fx=0.25, fy=0.25)
304
305     # Convert the image from BGR color (which OpenCV uses) to RGB color (which face_recogniti
306     rgb_small_frame = small_frame[:, :, ::-1]
307

```

Figure 5.2.2 OpenCV coding for detecting people face from the video frame.

Figure 5.2.2.1 shows that pseudocode for OpenCV to show people names in the camera frame. The line from 390 until 413 shows that the OpenCV will make rectangle frame around the faces that detect in the video frame. The size of the frame is assigned in the coding to fit the person face in the frame that will be created. Once the frame is created, OpenCV will shows the person's name and the accuracy of the face according to the data that send by HOG algorithm. The frame will be created in red colour around the face, which can be seen by the user. In line 418 shows the process can be quit by pressing 'q' in the command prompt.

```

facerec_pi_test.py
...
384 face_names.append(name)
385
386 process_this_frame = not process_this_frame
387
388
389
390 # Display the results
391 for (top, right, bottom, left), name in zip(face_locations, face_names):
392     # Scale back up face locations since the frame we detected in was scaled to 1/4 size
393     top *= 4
394     right *= 4
395     bottom *= 4
396     left *= 4
397
398     # Draw a box around the face
399     cv2.rectangle(frame, (left, top), (right, bottom), (0, 0, 255), 2)
400
401     #accuracy
402     text = "{:.2f}%".format(linear_val * 100)
403

```

```

404 # Draw a label with a name below the face
405 cv2.rectangle(frame, (left, bottom - 35), (right, bottom), (0, 0, 255), cv2.FILLED)
406 cv2.rectangle(frame, (left, bottom + 25), (right, bottom), (0, 0, 255), cv2.FILLED)
407 font = cv2.FONT_HERSHEY_DUPLEX
408 fonts = cv2.FONT_HERSHEY_SIMPLEX
409 cv2.putText(frame, name, (left + 6, bottom - 6), font, 1.0, (255, 255, 255), 1)
410 cv2.putText(frame, text, (left + 6, bottom + 25), font, 1.0, (255, 255, 255), 1)
411
412 # Display the resulting image
413 cv2.imshow('Video', frame)
414
415
416
417 # Hit 'q' on the keyboard to quit!
418 if cv2.waitKey(1) & 0xFF == ord('q'):
419     break
420
421 # Release handle to the webcam
422 video_capture.release()
423 cv2.destroyAllWindows()
424

```

Figure 5.2.2.1 OpenCV coding for showing known person name with frame on face.

5.2.3 Histogram of Oriented Gradients (HOG) algorithm in Raspberry Pi Board

Code snippets below shows how the HOG algorithm train the data and recognize people face by comparing face that captured with the face that trained in the Raspberry Pi Board. In figure 5.2.3 at line 25 shows that HOG algorithm get images data from the profile where all the valid users images are stored to let the HOG algorithm train to learn their face features. In line 26 shows that the data of the images that get from the profile will be stored as an encoding to do recognition easily. This code applied to every images that uploaded to let the HOG train for recognition.

```

facerec_pi_test.py x
23
24 # Load a sample picture and learn how to recognize it.
25 tony_image = face_recognition.load_image_file("profiles/Tony.jpg")
26 tony_face_encoding = face_recognition.face_encodings(tony_image)[0]
27 tony_image = face_recognition.load_image_file("profiles/Tony1.jpg")
28 tony_face_encoding = face_recognition.face_encodings(tony_image)[0]
29 tony_image = face_recognition.load_image_file("profiles/Tony2.jpg")
30 tony_face_encoding = face_recognition.face_encodings(tony_image)[0]
31 tony_image = face_recognition.load_image_file("profiles/Tony3.jpg")
32 tony_face_encoding = face_recognition.face_encodings(tony_image)[0]
33 tony_image = face_recognition.load_image_file("profiles/Tony4.jpg")
34 tony_face_encoding = face_recognition.face_encodings(tony_image)[0]
35 tony_image = face_recognition.load_image_file("profiles/Tony5.jpg")
36 tony_face_encoding = face_recognition.face_encodings(tony_image)[0]
37 tony_image = face_recognition.load_image_file("profiles/Tony6.jpg")
38 tony_face_encoding = face_recognition.face_encodings(tony_image)[0]
39 tony_image = face_recognition.load_image_file("profiles/Tony7.jpg")
40 tony_face_encoding = face_recognition.face_encodings(tony_image)[0]
41 tony_image = face_recognition.load_image_file("profiles/Tony8.jpg")
42 tony_face_encoding = face_recognition.face_encodings(tony_image)[0]
43 tony_image = face_recognition.load_image_file("profiles/Tony9.jpg")
44 tony_face_encoding = face_recognition.face_encodings(tony_image)[0]

```

Figure 5.2.3 HOG algorithm coding to get known images to train for recognition.

The source of the HOG algorithm code has been taken from the article HOG face recognition algorithm (Kosuke Mizuno, 2018), The original pseudocode for the HOG algorithm is shown in figure 5.2.3.0. The original HOG algorithm having problem in term of the accuracy in the recognizing people face. The algorithm having difficulties in term of recognizing people face correctly and the accuracy is lower compared to the SIFT algorithm that used to compare with HOG algorithm. Figure 5.2.3.0 shows the original coding of the HOG algorithm where in line 34 until line35 shows the algorithm getting images from the profile folder to train it. In line 38 shows every data of images that taken is added in the face encoding, to use it for comparison with images from camera and recognize the faces.



```

facerec_pi_test_profiles.py
30 #Loop to add images in friends folder
31 for file in os.listdir("profiles"):
32     try:
33         #Extracting person name from the image filename eg: david.jpg
34         known_person.append(file.replace(".jpg", ""))
35         file=os.path.join("profiles/", file)
36         known_image = face_recognition.load_image_file(file)
37         #print(face_recognition.face_encodings(known_image)[0])
38         known_face_encodings.append(face_recognition.face_encodings(known_image)[0])
39         #print(known_face_encodings)
40
41     except Exception as e:
42         pass
43
44 #print(len(known_face_encodings))
45 #print(known_person)

```

Figure 5.2.3.0 Original HOG algorithm face encoding.

To avoid the low accuracy of the HOG algorithm, the code has been modified to increase the accuracy and recognizing people easily. The face encoding plays the important role in recognizing people face. In figure 5.2.3.0 all images' data are added in the one face encoding which made the HOG algorithm having difficulty in recognizing people face and reduce the accuracy. To avoid this problem the face encoding is modified where every valid user has their own face encoding that separated from the other valid user, and it also assigned own names. By modifying the face encoding code, the HOG algorithm can recognize people face easily and the accuracy also high compared to the original HOG algorithm. Figure 5.2.3.1 shows the HOG algorithm that have been modified. In line 274 until 278 shows every encoding of the

specific person images is separated into own face encoding by creating arrays of known face encodings. In line 280 until 284 shows that the known person names assigned according to the arrays of face encoding where the algorithm will show the person's name if the known person face recognized. The value of variables is passed to other coding to do further recognitions.

```

facerec_pi_test.py
273 # Create arrays of known face encodings and their names
274 known_face_encodings = [
275     tony_face_encoding,
276     elon_face_encoding,
277     bella_face_encoding,
278     arasu_face_encoding
279 ]
280 known_face_names = [
281     "Tony",
282     "Elon",
283     "Bella",
284     "Arasu"
285 ]
286
287 print("[Info] Done Training Module")
288 # Initialize some variables
289 face_locations = []
290 face_encodings = []
291 face_names = []
292 process_this_frame = True

```

Figure 5.2.3.1 Coding of HOG algorithm that modified

Figure 5.2.3.2 shows that HOG algorithm compares images and do recognition while checking the accuracy of the recognition. In line 309 until 323 shows that the HOG coding finds the faces and face encoding in the video frame. After that, the algorithm does comparison between the faces from the frame with the faces that already in profile, to check the matches. If the face does not match it set as unknown variable but if the face matches the algorithm will set the name of the image according to the name variable that assigned and pass to the OpenCV to show on the frame. In line 327 until 340 shows that the accuracy of the image recognition calculated using threshold of 0.6. The face encoding of known and face encoding of frame is differentiated by checking the value of threshold of the images. Once the accuracy is calculated, the value will be transferred to OpenCV to show in the video frame.

```

facerec_pi_test.py ✖
307
308 # Only process every other frame of video to save time
309 if process_this_frame:
310     # Find all the faces and face encodings in the current frame of video
311     face_locations = face_recognition.face_locations(rgb_small_frame)
312     face_encodings = face_recognition.face_encodings(rgb_small_frame, face_locations)
313
314     face_names = []
315     for face_encoding in face_encodings:
316         # See if the face is a match for the known face(s)
317         matches = face_recognition.compare_faces(known_face_encodings, face_encoding)
318         name = "Unknown"
319
320         face_distances = face_recognition.face_distance(known_face_encodings, face_encoding)
321         best_match_index = np.argmin(face_distances)
322         if matches[best_match_index]:
323             name = known_face_names[best_match_index]
324
325 #calling accuracy
326 #face_distance_to_conf(face_distances, face_match_threshold=0.6)
327 face_match_threshold=0.6
328
329 if face_encoding[1] > face_match_threshold:
330     range = (1.0 - face_match_threshold)
331     linear_val = (1.0 - face_encoding[1]) / (range * 2.0)
332 #return linear_val
333 print(linear_val)
334
335 else:
336     range = face_match_threshold
337     linear_val = 1.0 - (face_encoding[1] / (range * 2.0))
338     linear_val = linear_val + ((1.0 - linear_val) * math.pow((linear_val - 0.5) * 2, 0
339 #return linear_val + ((1.0 - linear_val) * math.pow((linear_val - 0.5) * 2, 0.2))
340 print(linear_val)
341

```

Figure 5.2.3.2 HOG coding for compare images and show accuracy of recognition

Figure 5.2.3.3 shows HOG algorithm for getting parameters of face from the image. The algorithm gets the parameters according to the face size in the image or in the video frame. The size of the eyebrows, nose, eyes and lips will be calculated and stored in the face encoding. This process is done to the image that in the profile data and also to the face that detected in the video frame. The parameter points will be added according to the size of the parameters. If the faces are not support the model type that assigned which is small and large, the HOG algorithm will show error messages during the recognition process.

```

api.py x
179
180 # For a definition of each point index, see https://cdn-images-1.medium.com/max/1600/1*AbEg3
181 if model == 'large':
182     return {
183         "chin": points[0:17],
184         "left_eyebrow": points[17:22],
185         "right_eyebrow": points[22:27],
186         "nose_bridge": points[27:31],
187         "nose_tip": points[31:36],
188         "left_eye": points[36:42],
189         "right_eye": points[42:48],
190         "top_lip": points[48:55] + [points[64]] + [points[63]] + [points[62]] + [points[61]]
191         "bottom_lip": points[54:60] + [points[48]] + [points[60]] + [points[67]] + [points[6
192     } for points in landmarks_as_tuples]
193 elif model == 'small':
194     return {
195         "nose_tip": [points[4]],
196         "left_eye": points[2:4],
197         "right_eye": points[0:2],
198     } for points in landmarks_as_tuples]
199 else:
200     raise ValueError("Invalid landmarks model type. Supported models are ['small', 'large']".
201

```

Figure 5.2.3.3 HOG coding for getting face parameters in image frame

5.2.4 Gmail notification coding in Raspberry Pi Board

Code snippets below shows how the notification is send to the user email when there is unknown person is detected. Figure 5.2.4 shows the pseducode for the Gmail notification if unknown people face detected. In line 349 until 367 shows that email requiremnt which is the subject, To, From information are added to allow the coding add it in email automatically once its run. The image of unknown person is captured and stored in unknown file. The path of the file is inserted in the coding to allow it take the image and insert into gmail to send it to the user. The protocol smtp.gmai.com and 587 is added in the coding to allow the gmail to send email from the Raspberry Pi. The information of the user email is added to allow to use the user account to send the email. The information of such as gmail id and password are added in the coding.


```

facerec_pi_test.py ✖
348         else:
349             cv2.imwrite(filename='/home/pi/Desktop/Arasu/unknown.jpg',img=frame)
350
351             linear_val = 0.0
352
353             |
354             toaddr = 'psmarasu12@gmail.com'           # To id
355             me = 'psmarasu12@gmail.com'             # your id
356             subject = "ALERT Intruder"              # Subject
357
358             msg = MIMEMultipart()
359             msg['Subject'] = subject
360             msg['From'] = me
361             msg['To'] = toaddr
362             msg.preamble = "test "
363
364             part = MIMEBase('application', "octet-stream")
365             part.set_payload(open("/home/pi/Desktop/Arasu/unknown.jpg", "rb").read())
366             encoders.encode_base64(part)
367             part.add_header('Content-Disposition', 'attachment; filename="unknown.jpg"')
368             msg.attach(part)
369
370         try:
371             s = smtplib.SMTP('smtp.gmail.com', 587) # Protocol
372             s.ehlo()
373             s.starttls()
374             s.ehlo()
375             s.login(user = 'psmarasu12@gmail.com', password = 'thennarasu555') # User id & password
376             #s.send_message(msg)
377             s.sendmail(me, toaddr, msg.as_string())
378             s.quit()
379         except SMTPException as error:
380             print ("Error") # Exception
381

```

Figure 5.2.4 Gmail notification coding when unknown person detected

5.3 Hardware Configuration Management

The hardware used in this project are stated in the design phase. Raspberry Pi High Quality camera and FFC cable are attached to the Raspberry Pi board. Raspberry Pi High Quality camera receive command from HOG algorithm to control the hardware. Below shows the details of each pin that are inserted to Raspberry Pi board in figure 5.3.



Figure 5.3 Raspberry Pi 4 board

RASPBERRY PI HIGH QUALITY CAMERA + 6MM LENS	RASPBERRY PI MODULE
15-pin FFC slot	15-pin FFC camera slot

35-5v RPI COOLING FAN	RASPBERRY PI MODULE
Red Cable	Pin 4
Black Cable	Pin 6

Table 5.3 Details of pin number and slot

Figure 5.3.1 shows the prototype to demonstrate Raspberry Pi board. This prototype set up use all hardware needed as been mentioned in the hardware requirements. Hardware connection such as Raspberry Pi board is connected to power source that is 15W (5V/3A) PSU USB C plug via USB C port.



Figure 5.3.1 Prototype of Smart CCTV camera with face recognition.

5.3.1 Connection between Raspberry Pi board and Raspberry Pi High Quality Camera.

Code snippets below shows how the Raspberry Pi board connect to the Raspberry Pi High Quality camera to receive video frame. The figure 5.3.2 shows that the pseudocode to allow the Raspberry Pi board access the the Raspberry Pi camera. In line 16 shows that the coding try to access the camera for get the video that being recorded by the camera.

```

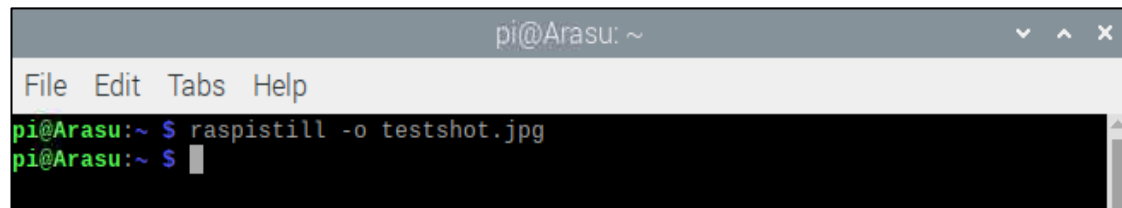
facerec_pi_test.py ✕
3 import face_recognition
4 import cv2
5 import numpy as np
6 import math
7 import smtplib,ssl
8 from email.mime.multipart import MIMEMultipart
9 from email.mime.base import MIMEBase
10 from email.mime.text import MIMEText
11 from email.utils import formatdate
12 from email import encoders
13
14
15 # Get a reference to webcam #0 (the default one)
16 video_capture = cv2.VideoCapture(0)

```

Figure 5.3.2 Raspberry Pi Camera and Board Connection coding

5.3.2 Connection successful between Raspberry Pi and Raspberry Pi camera.

Figure 5.3.2.1 shows the way to determine whether the Raspberry Pi board is successfully connected to the Raspberry Pi High Quality Camera or not. The command “raspistill -o testshot.jpg” is used to know the Raspberry Pi camera successfully connected or not to the Raspberry Pi board. Once the command is run, the Raspberry Pi will check the connection of the camera by opening camera to capture image.



```

pi@Arasu: ~
File Edit Tabs Help
pi@Arasu:~ $ raspistill -o testshot.jpg
pi@Arasu:~ $

```

Figure 5.3.2.1 Command to check Raspberry Pi camera connection.

Figure 5.3.2.2 shows that the Raspberry Pi camera is successfully connected and able to capture picture when the command is run in the command prompt. The camera will open lens to capture image only if run the command.

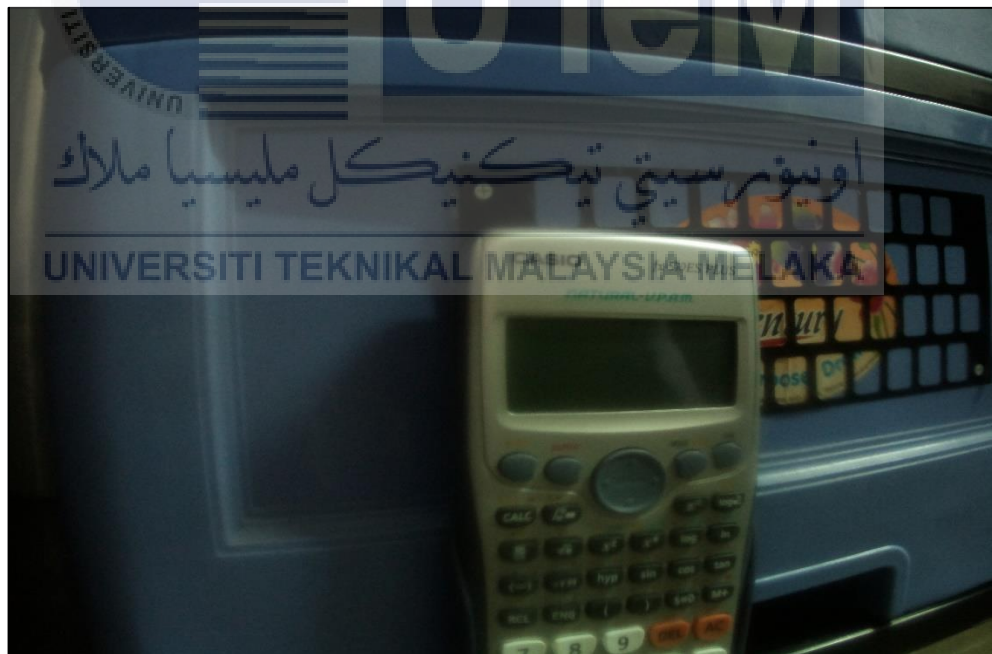


Figure 5.3.2.2 Raspberry Pi Camera successfully connected and capture image.

5.4 Implementation Status

No	Component	Description	Duration to complete
1	Assembling Hardware	Process of connecting FFC cable, RPI Fan, Raspberry Pi High Quality Camera to the Raspberry Pi board by assigning into pins and slots available.	14 days
2	Building Prototype	This process is to setup the prototype of the project	20 days
3	Upload Raspberry Pi Camera OS and Raspberry Pi Camera coding	This process is to ensure the Raspberry Pi board can control the camera that attached to it.	28 days
4	Developing Histogram of Oriented Gradients algorithm	All the HOG algorithm coding will be applied and arranged in the Raspberry Pi board.	30 days

Table 5.4 Status implementation

5.5 Conclusion

In conclusion, for this chapter the prototype has been implemented according to the designs, problem statement and objectives. The HOG algorithm environment shows how the face recognition is working, configuration environment for hardware and software classifies the both of them needed to be installed in line so the function will be stable. It shows a clear picture and vision of developing the project to accomplish the purpose of this project. The implementation status is the duration to complete the task of the job. For the next chapter is to test the product and to know whether the product is a success or fail. The stage will also include the accuracy of the face recognition.



CHAPTER 6: TESTING

6.0 Introduction

In this chapter, the testing of the prototype is carried out to know whether it is successful or not. The prototype is made by using Raspberry Pi board, Raspberry Pi High Quality Camera and HOG algorithm. So, all of this hardware and the software need to be tested. If the testing is not being carry out, the prototype cannot be justified to be success and need some feedback from the users.

6.1 Test Plan

Testing plans describe the activities, scope and basic testing for the application. It is important to ensure all the objective achieved and overall, of the application is running smoothly. Any bugs or errors can be detected and fixed it before sending to the customer.

6.1.1 Test Environment

This project will remotely access through laptop by connecting wirelessly to establish the connection for Raspberry Pi board before carrying out any testing process. The Raspberry Pi will be running with the Raspberry Pi High Quality camera and the Raspberry Pi board will be connected to the Wi-Fi connection. This is ensuring the real-time capture image can be test out successfully.

6.1.2 Test Schedule

The testing processes for the system will take period of time. Any bugs or error being found will return be back to the implementation phase to check again. This is the continuation process until the system is error-free and able to execute successfully.

Module Name	Duration	Test Start Date	Test Date Completed
Communication between Raspberry Pi board and Raspberry Pi High Quality Camera	2 weeks	5 July 2021	18 July 2021
Communication between Raspberry Pi Camera and OpenCV coding.	2 weeks	19 July 2021	1 August 2021
Accuracy of the HOG algorithm in recognizing face.	2 weeks	2 August 2021	15 August 2021
Gmail real time notification	2 weeks	16 August 2021	29 August 2021

Table 6.1.3 Table of Test Schedule

6.2 Test Strategy

Test strategy is guiding plan for the software development cycle of this project. It will explain the test method where the phase is a breakthrough in deciding who, what, when and how to test. The project will be tested by project developer.

6.2.1 Classes of Tests

The classes of tests have four types of tests for this project to ensure the algorithm and hardware does not have any errors. The specification type of testing during the testing are software test, hardware test, usability test and functionality test. The classes of tests will be mentioned in the table 6.2.1.

Test	Description
Software Test	Any errors or problem in the code will be identified and repaired
Hardware Test	Any incorrect hardware and connections will be determined to ensure that all hardware involved in the project are in good condition
Functionality Test	To ensure that the functionality of the project as planned
Accuracy Test	To ensure that accuracy of the project is good.

Table 6.2.1 Table of classes of test

6.3 Test Design

Test design discusses about the test case identification, test cases and expected result for each scenario which are designed and documented. The test description discusses integration test and functionality test.

6.3.1 Test Description

6.3.1.1 Communication between Raspberry Pi board and Raspberry Pi High Quality Camera test

Test Case ID	TC_1
Test Functionality	To test Raspberry Pi board and Raspberry Pi High Quality Camera functionality and communication.
Test Case Summary	To upload coding to the Raspberry Pi board
Precondition	Established the connection of Raspberry Pi board and Raspberry Pi High Quality Camera
Execution Steps	<ol style="list-style-type: none"> 1. Run the Raspberry Pi board and High-Quality Camera 2. Write coding to upload into the Raspberry Pi board 3. Upload
Expected Result	The code is successfully uploaded to the Raspberry Pi board
Error Message	If the coding is not correct or error, the coding will show error alert and not able to run successfully.

Table 6.3.1.1 Table of testing communication Raspberry Pi and Raspberry Pi camera

6.3.1.2 Communication between Raspberry Pi High Quality Camera and OpenCV coding

Test Case ID	TC_2
Test Functionality	To test OpenCV coding functionality
Test Case Summary	To let the OpenCV coding communicate to the Raspberry Pi High Quality Camera
Precondition	Established the OpenCV coding and Raspberry Pi High Quality Camera.

Execution Steps	<ol style="list-style-type: none"> 1. Write OpenCV coding in Raspberry Pi board 2. Setup Raspberry Pi camera with OpenCV coding 3. Switch on camera for testing
Expected Result	The OpenCV could be able to communicate or make connection with the Raspberry Pi High Quality camera
Error Message	If the connection cannot be reaching the Raspberry Pi High Quality cannot detect the Raspberry Pi camera.

Table 6.3.1.2 Table of testing communication OpenCV coding and Raspberry Pi camera

6.3.1.3 Accuracy of HOG algorithm in recognizing face test

Test Case ID	TC_3
Test Functionality	To test difference of image in data and image that captured by the camera by using accuracy.
Test Case Summary	To shows the accuracy capacity of the HOG algorithm in recognizing valid user face.
Precondition	Established the HOG algorithm in the Raspberry Pi board with the accuracy coding.
Execution Steps	<ol style="list-style-type: none"> 1. Upload and write HOG algorithm code in the Raspberry Pi board 2. Run the HOG algorithm 3. View the result in the camera video frame
Expected Result	Video frame in camera expected to show the accuracy of the face that being recognized by the HOG algorithm
Error Message	If the accuracy calculation coding having error, the accuracy won't be showed in the video frame.

Table 6.3.1.3 Table of testing accuracy of HOG algorithm in face recognition

6.3.1.4 Testing email notifications

Test Case ID	TC_4
Test Functionality	To test notification from Raspberry Pi through email functionality.
Test Case Summary	To receive the image of unknown or stranger that captured by Raspberry Pi through email.
Precondition	Established the people face not in the valid user data images.
Execution Steps	<ol style="list-style-type: none"> 1. Open Raspberry Pi board 2. Write coding for email notification 3. Show the image of stranger in front of the camera
Expected Result	The Raspberry Pi board able to send the image of the unknown face to user via email using the internet connection in the Raspberry Pi.
Error Message	If the camera could not capture stranger image, notification will not be sent to the email

Table 6.3.1.4 Table of testing email

6.3.2 Test Data

The test data is the data used to verify the data on the project. Therefore, the test data is to verify whether a set of input data is provided to a set of functions, which will produce some expected data output, such as accuracy of the face recognition in the camera video frame and send the image of strangers through the email.

6.4 Test Result and Analysis

Test results consists of expected output and description of feedback.

6.4.1 Test of TC_1

Module: Communication between Raspberry Pi board and Raspberry Pi High Quality Camera.

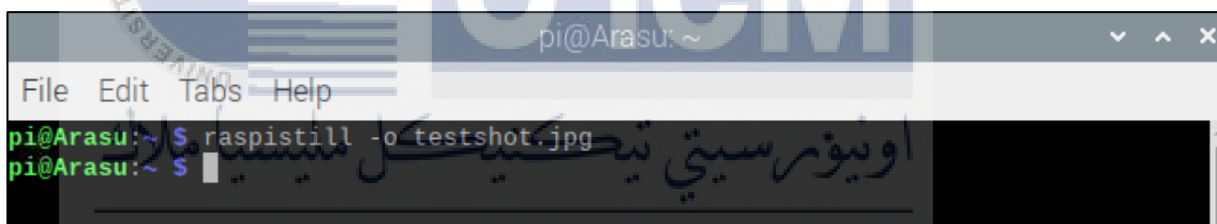
Test case: Functionality

Test Start Date: 5 July 2021

Test End Date 18 July 2021

Duration Cycle: 14 days

Test of TC_1 required Raspberry Pi board, Raspberry Pi High Quality Camera and personal laptop for this process, coding need to be written in the Raspberry Pi board. The Raspberry Pi camera could not capture picture if there is error in coding or the camera not connected and not enabled in the Raspberry Pi board. Developer needs to check whether the camera is connected properly, and camera is enabled in Raspberry Pi board. Figure 6.4.1 shows the way to test the connection of camera in Raspberry Pi. The figure 6.4.1.1 shows the path where the image that took during the testing process will be stored and can be accessed. The figure 6.4.1.2 shows the output of image after testing the communication between Raspberry Pi board and Raspberry Pi camera.



```
pi@Arasu: ~  
File Edit Tabs Help  
pi@Arasu: ~ $ raspistill -o testshot.jpg  
pi@Arasu: ~ $
```

Figure 6.4.1 Command to test TC_1

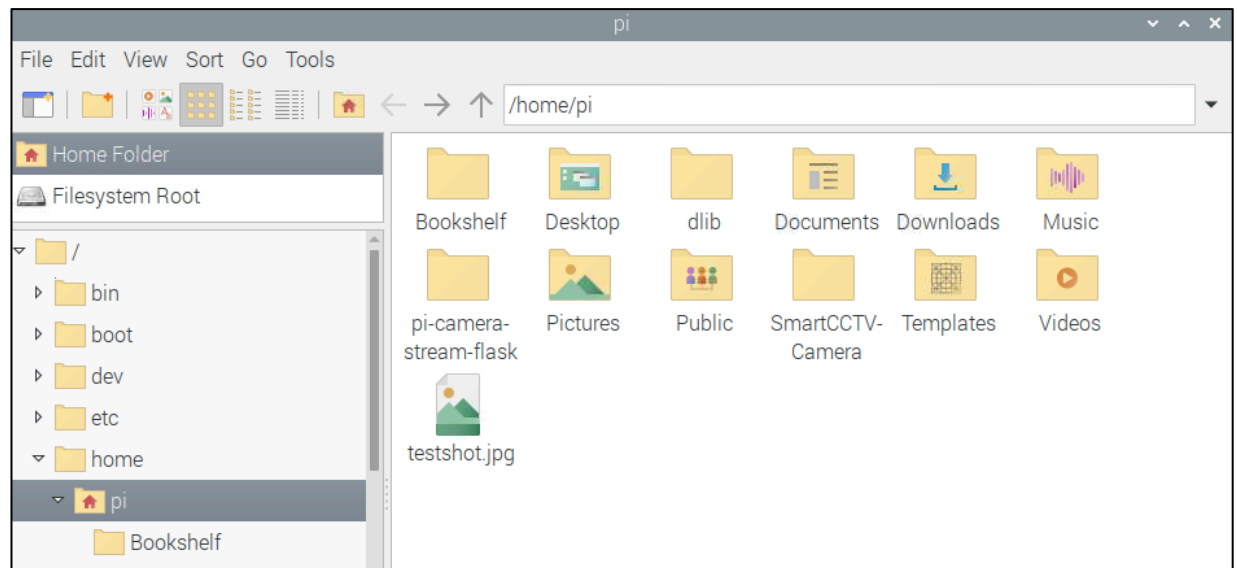


Figure 6.4.1.1 Path of testing images saved.



Figure 6.4.1.2 Success output from the communication.

Test ID	Test Identification	Result (Pass/Failed)
TC_1	OK	PASS

Figure 6.4.1.3 Result of Output

6.4.2 Test of TC_2

Module: Communication between Raspberry Pi High Quality Camera and OpenCV coding

Test case: Functionality

Test Start Date: 19 July 2021

Test End Date: 1 August 2021

Duration Cycle: 14 days

Test of TC_2 required Raspberry Pi High Quality Camera, personal laptop and OpenCV coding for this process, OpenCV coding need to be python3 coding that could read by the Raspberry Pi board. After the connection between Raspberry Pi board and camera is success then can view video frame from the camera through the personal laptop. Once the OpenCV coding uploaded in the Raspberry Pi board, run the code and view the camera whether it could detect people face by making red box around the people face. Figure 6.4.2 shows the OpenCV coding that successfully uploaded and runed in the Raspberry Pi. Figure 6.4.2.1 shows result of coding where it detects people face by making red framed around the face.

اونيور سیتی تکنیکل ملیسیا ملاک

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

The screenshot shows the Thonny IDE interface. The top toolbar includes icons for New, Load, Save, Run, Debug, Over, Into, Out, Stop, Zoom, and Quit. The main editor window displays the following Python code:

```

394
395 #accuracy
396 text = "{:.2f}%".format(linear_val * 100)
397
398 # Draw a label with a name below the face
399 cv2.rectangle(frame, (left, bottom - 35), (right, bottom), (0, 0, 255), cv2.FIL
400 cv2.rectangle(frame, (left, bottom + 25), (right, bottom), (0, 0, 255), cv2.FIL
401 font = cv2.FONT_HERSHEY_DUPLEX
402 fonts = cv2.FONT_HERSHEY_SIMPLEX
403 #cv2.putText(frame, name, (left + 6, bottom - 6), font, 1.0, (255, 255, 255), 1
404 #cv2.putText(frame, text, (left + 6, bottom + 25), font, 1.0, (255, 255, 255),
405
406 # Display the resulting image
407 cv2.imshow('Video', frame)
408

```

The terminal window below shows the execution output:

```

>>> %Run facerec_pi_test.py
[Info] Training Module...
[Info] Done Training Module
[Info] Initiating Camera...

```

Figure 6.4.2 Test TC_2 by running the coding in Raspberry Pi

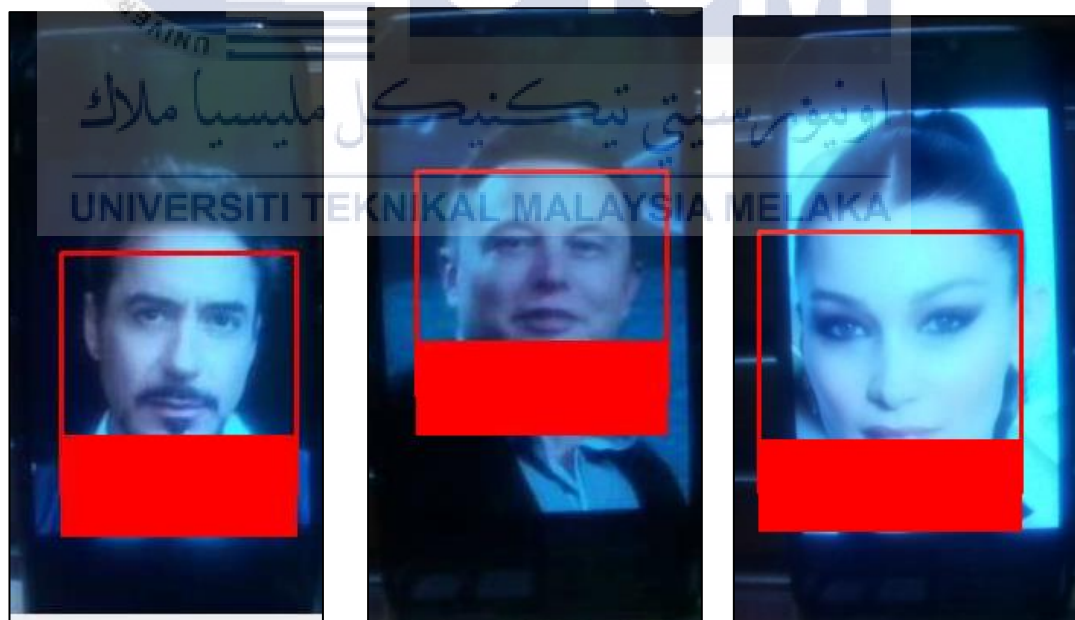


Figure 6.4.2.1 Output showing face successfully detected.

Test ID	Test Identification	Result (Pass/Failed)
TC_2	OK	PASS

Figure 6.4.2.2 Result of Output

6.4.3 Test of TC_3

Module: Accuracy of HOG algorithm in recognizing face

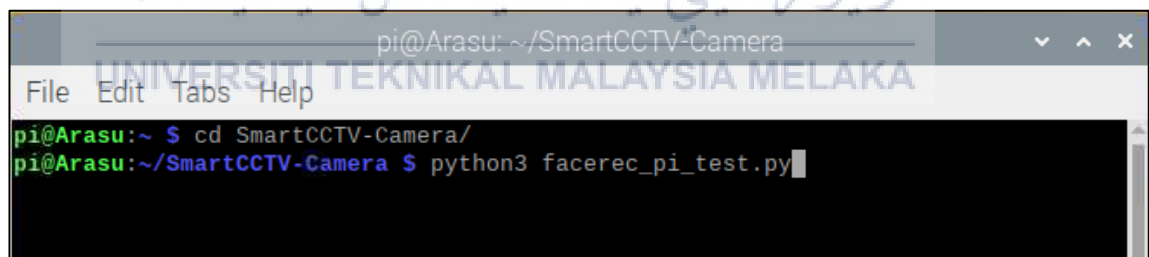
Test case: Accuracy

Test Start Date: 2 August 2021

Test End Date: 15 August 2021

Duration Cycle: 14 days

Test of TC_3 required Raspberry Pi board, Raspberry Pi High Quality Camera, HOG algorithm and personal laptop for this process, the HOG algorithm need to test by using 4 sample of valid user images. 30 different types of expression images are added in database for each valid user. The testing is done by comparing Histogram of Oriented Gradients (HOG) algorithm accuracy of face recognition with the Scale-invariant feature transform (SIFT) algorithm to make sure the HOG algorithm has high accuracy compared to the SIFT algorithm that currently used by people. The technique that used to compare the accuracy of the algorithm is by using amount of dataset or image of the valid users. (T.Lindeberg, 2018). The number of images used to both algorithms are same which is 30 images for each user. The algorithm tested by using same picture of valid user 10 times and getting the accuracy for every tests. Figure 6.4.3 shows command to run HOG algorithm for testing face recognition.

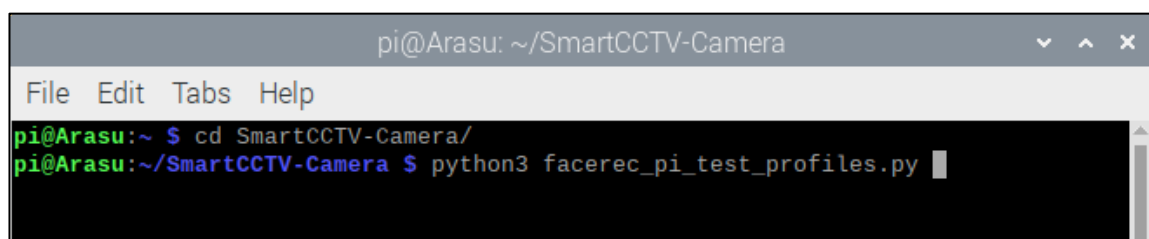


```

pi@Arasu: ~/SmartCCTV-Camera
File Edit Tabs Help
pi@Arasu:~ $ cd SmartCCTV-Camera/
pi@Arasu:~/SmartCCTV-Camera $ python3 facerec_pi_test.py

```

Figure 6.4.3 Test TC_3 by running the HOG command in Raspberry Pi board



```

pi@Arasu: ~/SmartCCTV-Camera
File Edit Tabs Help
pi@Arasu:~ $ cd SmartCCTV-Camera/
pi@Arasu:~/SmartCCTV-Camera $ python3 facerec_pi_test_profiles.py

```

Figure 6.4.3.1 Test TC_3 by running the SIFT command in Raspberry Pi board

The HOG algorithm tested with the 4 valid user images and the result have been noted to compare between SIFT algorithm. Table 6.4.3.1 shows the first valid user “Tony” picture used to do face recognition and check accuracy of the recognition. 10 test is done on the HOG and SIFT algorithm to compare the accuracy. Table 6.4.3.2 and table 6.4.3.3 shows the valid user “Elon” and “Bella” tested for face recognition and also the accuracy of recognition.



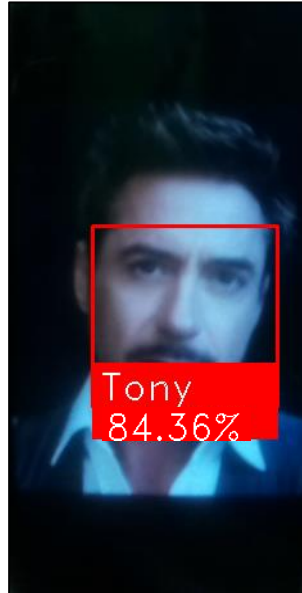
Algorithm	HOG (Modified)	SIFT	HOG (Original)
Valid user 1 name “Tony”			
1 Test of Accuracy	99.76%	88.69%	84.36%
2 Test of Accuracy	99.56%	88.98%	83.72%
3 Test of Accuracy	99.55%	88.58%	82.38%
4 Test of Accuracy	99.59%	87.79%	86.33%
5 Test of Accuracy	99.52%	87.95%	80.81%
6 Test of Accuracy	99.79%	89.06%	85.39%
7 Test of Accuracy	99.85%	88.43%	82.64%
8 Test of Accuracy	99.76%	88.35%	85.60%
9 Test of Accuracy	99.80%	87.98%	88.83%
10 Test of Accuracy	99.84%	88.98%	86.86%

Table 6.4.3.1 Comparison output of algorithm accuracy on valid user “Tony”


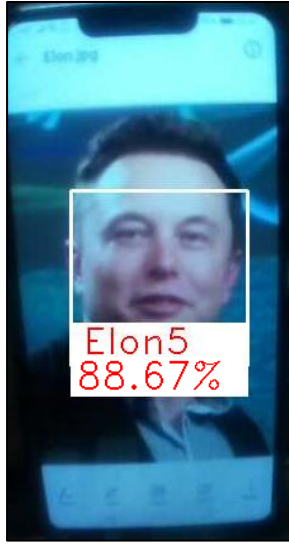

Algorithm	HOG (Modified)	SIFT	HOG (Original)
Valid user 2 name "Elon"			
1 Test of Accuracy	99.98%	88.67%	93.87%
2 Test of Accuracy	99.97%	85.13%	93.79%
3 Test of Accuracy	99.93%	87.28%	93.57%
4 Test of Accuracy	99.94%	88.09%	93.89%
5 Test of Accuracy	99.95%	86.52%	93.97%
6 Test of Accuracy	99.83%	87.99%	93.87%
7 Test of Accuracy	99.95%	84.09%	93.27%
8 Test of Accuracy	99.97%	85.57%	93.77%
9 Test of Accuracy	99.99%	88.56%	92.67%
10 Test of Accuracy	99.98%	88.44%	92.91%

Table 6.4.3.2 Comparison output of algorithm accuracy on valid user "Elon"

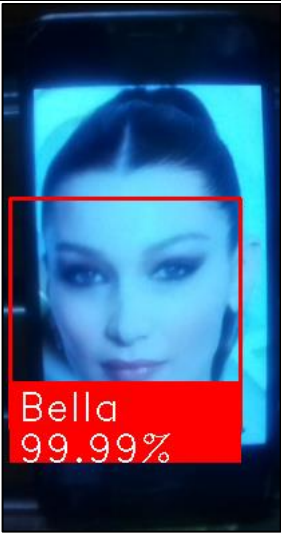


Algorithm	HOG (Modified)	SIFT	HOG (Original)
Valid user 3 name "Bella"			
1 Test of Accuracy	99.99%	87.01%	88.74%
2 Test of Accuracy	99.98%	84.78%	85.77%
3 Test of Accuracy	99.97%	85.35%	87.40%
4 Test of Accuracy	99.98%	85.47%	88.48%
5 Test of Accuracy	99.99%	86.57%	85.84%
6 Test of Accuracy	99.98%	86.50%	89.34%
7 Test of Accuracy	99.97%	85.98%	12.44%
8 Test of Accuracy	99.99%	85.62%	83.22%
9 Test of Accuracy	99.98%	86.37%	88.49%
10 Test of Accuracy	99.99%	86.62%	80.49%

Table 6.4.3.3 Comparison output of algorithm accuracy on valid user "Bella"


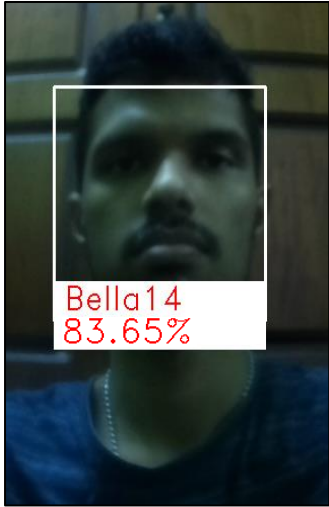

Algorithm	HOG (Modified)	SIFT	HOG (Original)
Valid user 4 name "Arasu"			
1 Test of Accuracy	99.77%	83.65%	89.37%
2 Test of Accuracy	99.72%	84.84%	88.39%
3 Test of Accuracy	99.68%	85.98%	89.03%
4 Test of Accuracy	99.74%	84.28%	85.85%
5 Test of Accuracy	99.80%	89.76%	88.46%
6 Test of Accuracy	99.83%	80.59%	87.32%
7 Test of Accuracy	99.76%	80.00%	86.15%
8 Test of Accuracy	99.71%	81.64%	87.14%
9 Test of Accuracy	99.64%	82.32%	88.49%
10 Test of Accuracy	99.78%	85.47%	89.04%

Table 6.4.3.4 Comparison output of algorithm accuracy on valid user "Arasu"

Test ID	Test Identification	Result (Pass/Failed)
TC_3	OK	PASS

Table 6.4.3.5 Result of output

6.4.4 Test of TC_4

Module: Email notifications

Test case: Functionality

Test Start Date: 16 August 2021

Test End Date: 29 August 2021

Duration Cycle: 14 days

Test of TC_4 required Raspberry Pi board, Raspberry High-Quality Camera, Unknown face and personal laptop for this process. The notification needs to be test with showing the unknown or stranger face that does not consider as valid user in HOG algorithm. The camera should be able to capture picture once unknown face is in front of the camera and able to send notification to the user with the image of unknown people that have been captured by the camera. Figure 6.4.4 shows two unknow faces are used to test the notification process. When the two unknow faces are shown in front of the camera, the algorithm successfully recognize that following face are not invalid users and shows unknown by making frame around their faces. Meanwhile in figure 6.4.4.1 shows the user receive email notification when the unknown faces are shown in front of camera. The system keeps sending email notification when the unknown faces remain in front of camera and also to alert the user. In figure 6.4.4.2 shows email notification the receive by the user is attached with the picture of the unknown people faces that captured by the camera. It shows that the email notification successfully sends to user when intruder detected in the camera.

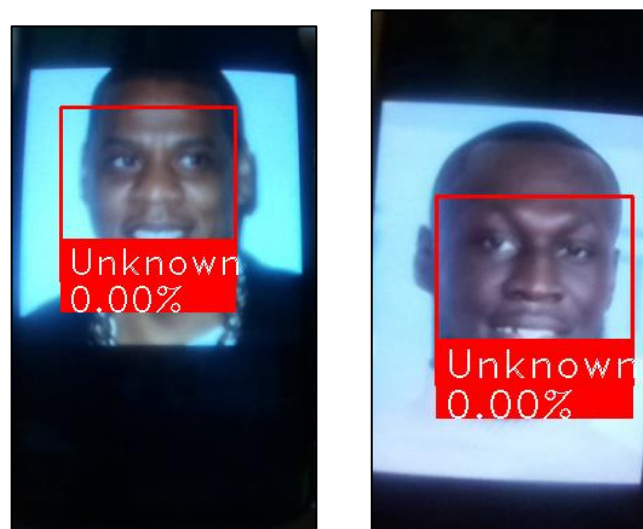


Figure 6.4.4 Invalid user faces that used to test notifications

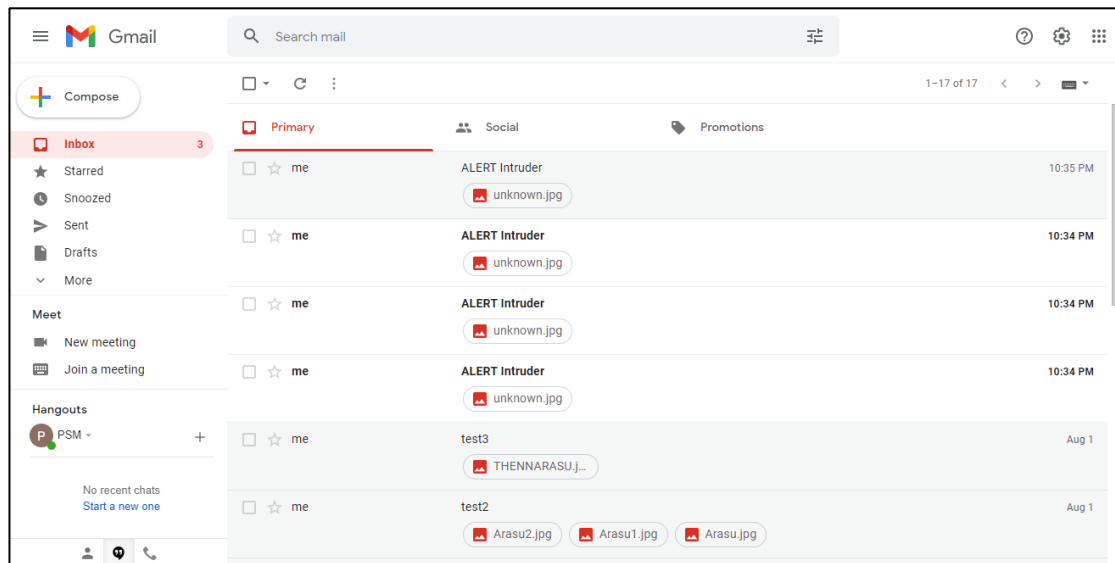


Figure 6.4.4.1 User receive email notification named “Alert Intruder”

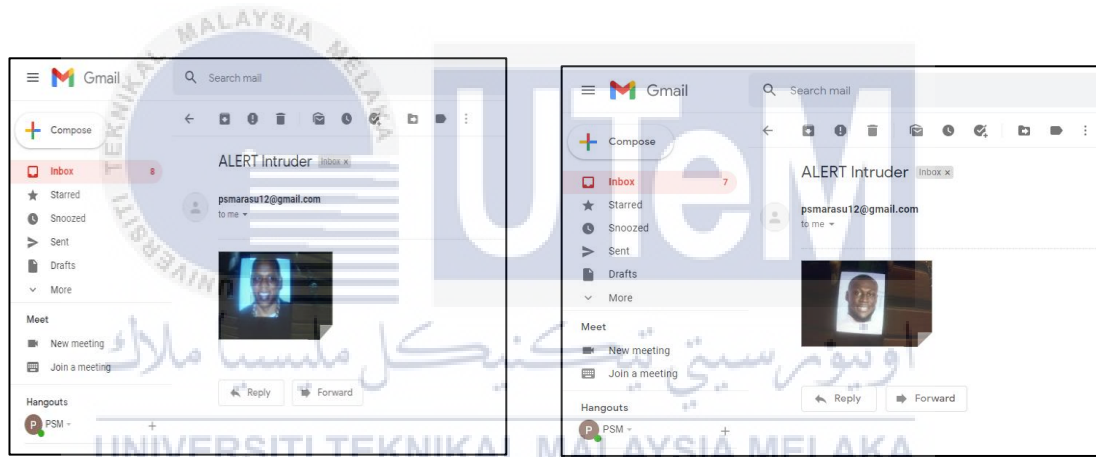


Figure 6.4.4.2 Raspberry Pi successfully send notification with the image of intruder to user

Test ID	Test Identification	Result (Pass/Failed)
TC_3	OK	PASS

Table 6.4.4 Result of output

6.4.5 Analysis Summary

There are 4 test cases has been tested for this project. All of these cases result is success. Test case 1, 2 and 4 is a testing functionality. The Raspberry Pi board, Raspberry Pi High Quality camera connection is fully function for this prototype. Developer can easily modify if there is addition in this project in the future. Test case 3 is accuracy testing for valid user images. With the HOG algorithm the valid user face can be recognized easily and also the accuracy of the HOG algorithm is high. Table 6.4.3.1, 6.4.3.2, 6.4.3.3 and 6.4.3.4 shows the result of TC_3 where the result proves the accurate of HOG algorithm compared to the SSIFT algorithm by using same amount of dataset.

All of the project objectives and problem statement has been achieved. The valid user can easily recognize, and the dataset also can update by developer time to time. Before the dataset being upload to the system, the HOG algorithm needs to be turn off during upload the dataset to avoid any errors. To delete the dataset in the HOG algorithm, the HOG algorithm should stop running to avoid errors. According to the accuracy, the top accuracy HOG can get for valid user recognition is up to 99% accurate. The accuracy of the data is proven with the dataset of TC_3.

6.5 Conclusion

In conclusion, for this chapter is crucial and need a lot of observation to make sure the testing is success. However, planning and timing need to be parallel to complete the testing. The prototype that has been developed need to do a lot of testing until the results satisfied developer. All the testing that has been done is to verify the products is a success or fail. If success, the testing and developing of the project is doing a good job and if fail, justification is a must. The next chapter is chapter seven which is project conclusion.

CHAPTER 7: PROJECT CONCLUSION

7.0 Introduction

In this chapter, there will be a conclusion and summarization the overall project from the beginning until the completion of the project. The limitation and future work of the project also will be stated for any further improvement in the future. This will effectively improve the efficiency of the system and make it more efficient also comprehensive.

7.1 Project Summarization

In the beginning, the project objective has been stated which are to identify features for face detection in smart CCTV Home Camera, to develop face recognition in smart CCTV Home Camera and to test and verify face recognition in smart CCTV Home Camera.

The definition of identify features for face detection in smart CCTV Home Camera for assemble all face recognition algorithms and find the suitable algorithm to make this project happens. As literature review has been done, a few algorithms have been shortlisted but the most suitable algorithm for the smart CCTV Home Camera is Histogram of Oriented Gradients algorithm has been chosen to make this project done successfully. This objective also can be link with second objective which is to develop face recognition in smart CCTV Home Camera. After the algorithm is chosen for this project, it is implemented in the system which is Raspberry Pi board that is connected to the Raspberry Pi High Quality Camera that can be used to achieve the second objective where the HOG algorithm coding added in Raspberry Pi and the camera act

as smart CCTV camera to do face recognition. Furthermore, the third objective is also link to the second objective to complete the system. The face recognition is tested in Raspberry Pi camera and verify the recognition by checking the accuracy of the face recognition on valid users. Finally, this prototype will send the notification with the unknown people images to the user if its detected unknown face in the face recognition camera.

The result of the project is success and complete all the objectives. It takes time and need a lot of research as developing new products is not easy as it seems. The strength of the product is it have high accuracy in face recognition and also can send real-time notifications to the user through the email. The weakness of the products is the camera where if the person is too far from camera its hard to do face recognition.

7.2 Project Contribution

The project contribution that has been mentioned before in Chapter 1 are smart CCTV home camera with face recognition, the high accuracy in recognizing valid user faces by using Raspberry Pi camera and able to send notification to the user through email if unknown faces are detected in the face recognition.

All the project contribution that has been mentioned is for individual purposes and of course for users that interested to use this product and recognize and prevent from any theft or burglar cases.

7.3 Project Limitation

The project limitation during doing this project are time and limited resources. In this project, the prototype lack of clear and far view which will make the prototype more flexible and efficient. Due to the project needs to be completed as soon as possible, the low density of camera is applied to the prototype.

The low-density camera consists of less coverage that won't be able to capture or have a low clear view of people face when they are far from the camera. The camera also static which could not be able to move and only can recognize people when they get closer to the camera. The algorithm is already had high accuracy algorithm and able recognize people face easily but due to the camera density it reduces the efficiency of the face recognition system.

7.4 Future Works

For the future works, the suggestion that can be made is to build the camera able to zoom in and out automatically to detect people face from far or close and can be able to rotate 360 degrees to detect people surrounding of the area. After that, the prototype can upgrade by adding camera that have high density coverage to make the system more efficient.

Next, the algorithm accuracy is higher than the other algorithms, this method can be use in others camera. Which is the camera no need to be Raspberry Pi camera to do the face recognition. The system can upgrade by build the camera that able to automatically zoom in to get clear view of people face that not clear enough. The camera also must be able to detect people that far away from the camera. The camera also should be able to move around 360 degrees to fill the view around the house area. It will allow the surrounding area more secure from thief or burglar. It will be more efficient smart home CCTV camera to be a new project It just took a longer time to be execute for this project. The resources to help also limited due to the cost of the devices for this project.

7.5 Conclusion

In conclusion, the projects met the objectives that have been stated before and achieved the goals. However, to make this project perfects a lot of hard work and sacrifice is needed. The problem statement that has been stated have been solved.

From the chapter until last chapter, there is a lot thing that happened and fortunately all the problems can be solved. Chapter 1 until 4 the documentation part is easy to do as the product is still developing and not have problems yet. When the product has been developed, there is problems occur like how to implement the coding and error when debug the coding.

A lot of research needs to do before developing the product as this product is a new product and there is limited resource to find the similar product that can be refer. However, with a lot of reading and research the product can be developed. It takes time to complete the development of the product.

Last but not least, for the future of this product, it can be used by companies for more secure rather than using it in home only. The density of the camera can be improved, and the user can receive a lot of benefits from the product. The product can

benefit everyone and organizations in the future by developing this product which is Smart Home CCTV camera with face recognition by only using Raspberry Pi board with combination of Raspberry Pi camera.



REFERENCES

- Goudail, F., Lange, E., Iwamoto, T., Kyuma, K., & Otsu, N. (2017, Oct). Face recognition system using local autocorrelations and multiscale integration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1024 - 1028. doi:10.1109/34.541411
- Hteik Htar Lwin, A. S. (2018). Automatic Door Access System Using Face. *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY*, 125 - 299. doi:10.1109/CCST.2015.8176828
- Huang, K., & Trivedi, M. (2018). Robust real-time detection, tracking, and pose estimation of faces in video streams. *International Conference on Pattern Recognition*, 256 - 456. doi:10.1109/ICPR.2004.1334689
- Jain, A., Basantwani, S., Kazi, O., & Bang, Y. (2017). Smart surveillance monitoring system. *International Conference on Data Management, Analytics and Innovation*, 723 -726. doi:10.1109/ICDMAI.2017.8073523
- Kraus, K., & Reda, R. (2018). Ad-hoc connections of miscellaneous sensors in a CCTV system. *International Symposium on Computer and Information Sciences*, 246 - 326. doi:10.1109/ISCIS.2008.4717969
- Mulla, M. R., Patil, R. P., & Shah, S. K. (2018). Facial image based security system using PCA. *International Conference on Information Processing (ICIP)*, 156 - 200. doi:10.1109/INFOP.2015.7489445
- Nico Surantha, W. R. (2018). Procedia Computer Science. *Design of Smart Home Security System using Object Recognition and PIR Sensor*, 465-472. doi:doi.org/10.1016/j.procs.2018.08.198
- Nwalozie G. C1, A. A. (2015, June 6). Enhancing Home Security Using SMS-based. *International Journal of Computer Science and Mobile Computing*, 1177 – 1184. doi:10.1108/34.541411
- Sahani, M., Nanda, C., Sahu, A. K., & Pattnaik, B. (2016). Web-based online embedded door access control and home security system based on face recognition. *International Conference on Circuits Power and Computing Technologies*, 233 - 255. doi:10.1109/ICCPCT.2015.7159473
- Vel'as, A., Kutaj, M., & Ďurovec, M. (2017). Influence of changing the parameters of the camera system on video-based motion detection. *International Carnahan Conference on Security Technology (ICCST)*, 521 - 603. doi:10.1109/CCST.2017.8167829

APPENDIX A

1) Coding for HOG algorithm

```

# Create arrays of known face encodings and their names
known_face_encodings = [
    tony_face_encoding,
    elon_face_encoding,
    bella_face_encoding,
    arasu_face_encoding
]
known_face_names = [
    "Tony",
    "Elon",
    "Bella",
    "Arasu"
]

print("[Info] Done Training Module")
# Initialize some variables
face_locations = []
face_encodings = []
face_names = []
process_this_frame = True
#linear_val = 0
print("[Info] Initiating Camera...")
#bar.finish()

```

2) Accuracy calculation coding

```

if matches[best_match_index]:
    name = known_face_names[best_match_index]

#calling accuracy
#face_distance_to_conf(face_distances, face_match_threshold=0.6)
    face_match_threshold=0.6

    if face_encoding[1] > face_match_threshold:
        range = (1.0 - face_match_threshold)
        linear_val = (1.0 - face_encoding[1]) / (range * 2.0)
#return linear_val
        print(linear_val)

    else:
        range = face_match_threshold
        linear_val = 1.0 - (face_encoding[1] / (range * 2.0))
        linear_val = linear_val + ((1.0 - linear_val) * math.pow((linear_val - 0.5) * 2, 0.2))
#return linear_val + ((1.0 - linear_val) * math.pow((linear_val - 0.5) * 2, 0.2))
        print(linear_val)

```

APPENDIX B

User manual for Smart Home CCTV camera with face recognition:

1. Set a WIFI connection with name and password in the Raspberry Pi board for remote access.
2. Plug in the Raspberry Pi on power supply and make sure light blinking in the Raspberry Pi board.
3. Make sure the camera is connected to the Raspberry Pi board.
4. Run the main command to let the algorithm train and learn images in profile and the face recognition will be activated.
5. Before running the main command, make sure valid user images are added into profile.
6. User now can recognize valid user or invalid user that in front of camera.
7. Open browser in Raspberry Pi to view the video frame that being captured by camera.
8. User will receive email when invalid user is in front of the camera to alert user.